An open-source system for automating deployment, scaling, and management of containerized applications. Kubernetes itself is not a tool for containerizing applications or packaging them into containers. Kubernetes is an open-source **container orchestration platform** that manages the deployment, scaling, and management of containerized applications. To containerize an application or package it into a container, you would typically use a containerization tool like Docker. Once you have containerized your application with Docker, you can then use Kubernetes to manage and orchestrate these containers.

As organizations are shifting from monolithic to microservice, need for the system to manage these thousands of containers arise.

**Advantages**

Automatic scaling for containers

High availability or nearly 0 downtime

Portability

**Disadvantages**

Complex and difficult

Cost

Architecture:

Control plane/master node

The control plane's components make decisions about the cluster and manage the overall cluster. Ex- scheduling pods and responding to cluster events.

A screenshot of a computer

Description automatically generated

**API server** is a component of the Kubernetes control plane that exposes the Kubernetes API. It is the front end. Also do the communications with nodes

**Etcd** is a key value store. It stores all cluster data, including configuration information, secrets, and the current state of all resources. If your Kubernetes cluster uses etcd as its backing store, make sure you have a backup plan for the data. It stores snapshots that can be used in case of failures.

**Controller** checks the health of nods whether they are up or down. Checks whether something needs to be repaired and creates pods for containers. Kubernetes controllers keep the cluster state matching the desired state.

**scheduler** watches for newly created [Pods](https://kubernetes.io/docs/concepts/workloads/pods/) with no assigned [node](https://kubernetes.io/docs/concepts/architecture/nodes/), and selects a node for them to run on. Selects the worker node based on the availability.

Worker node

Kubernetes runs your workload by placing containers into Pods to run on Nodes. A node may be a virtual or physical machine, depending on the cluster.

**Kubelet**: it is primary "node agent" that runs on each node of a cluster. It can register the node with the apiserver. It is a process by which nodes communicate with each other.

Pods : smallest deployable unit in node or k8s. it is abstraction over containers from nodes. Usually 1 app per pods but can deploy more than 1. Each pod has its own ip for communication with each other. Ip addresses change when a pod is recreated even for the same container.

Service is used to assign permanent ip address for a group of pods as it changes when a pod dies. It is exposing pods within a cluster to other pods. Ex (ip be like 127.45.67.8) . also act as a load balancer

Ingress is designed to handle HTTP and HTTPS traffic and can route requests based on URL paths and exposes pods outside cluster. Ingress in Kubernetes is a resource used to manage external access to services within the cluster. While service can do the same, but it works on ip not on url

Configmap can store some sort of info like env var. ex- db url for mongodb database. It is better to keep in configmap as change in url will result in new image formation and repull into pods. Can be shared with multiple pods or nodes.

Secret is similar to configmap but it stores confidential info in encrypted format. X- userid and pass for mongodb . Can be shared with multiple pods or nodes.

Volume(data persistence): physical storage(local/remote) attached to a pod to store the generated data to prevent loss in case of pod death. It can be outside of k8s cluster like on cloud.

A screenshot of a computer

Description automatically generated

2 replication mech

In replication, Deployments are suitable for stateless applications where pods can be treated as interchangeable, while StatefulSets are designed for stateful applications that require stable identities. Ex db