Kubernetes

Session 5

30th June 2018

Rama Krishna Bhupathi

Kubernetes

It is a platform to orchestrate the deployment, scaling, and management of container-based applications. Kubernetes came out of research from Google.

- Kubernetes is a platform that encompasses a huge number of services and capabilities that keep growing. Its core functionality is its ability to schedule workloads in containers across your infrastructure.
- Supports several container engines such as Dockers ,Rkt (from CoreOS)
- Container Runtime Interface (CRI) provides various integration levels to other container engines.

Kubernetes

Kubernetes provides the following benefits:

- **Microservices, by breaking an application into smaller, manageable, scalable components.**
- **Fault-tolerant cluster in which if a single Pod replica fails (due to node failure, for example), another is started automatically.**
- **Horizontal scaling in which additional or fewer replicas of a Pod could be run by just modifying the "replicas" setting in the Replication Controller**
- **Higher resource utilization and efficiency.**
- **Separation of concerns. The Service development team does not need to interface with the cluster infrastructure team.**
- **Replication and Auto-scaling**
- **Fault Tolerance**

Kubernetes

Kubernetes concepts include Pod, Service, and Replication controller and nodes

What is a node?

A node or a worker is a machine (physical or virtual) running Kubernetes onto which Pods may be scheduled. The node could be the master node or one of the worker nodes . Also referred to as “minions”

What Is a Cluster?

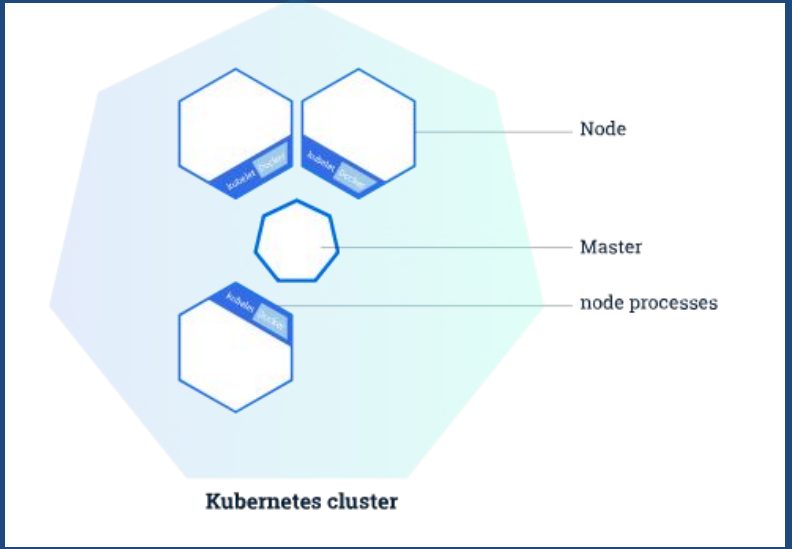
A cluster is a collection of nodes including other resources such as storage to run Kubernetes applications. A cluster has a single Kubernetes master node and zero or more worker nodes. A highly available cluster consists of multiple masters or master nodes.

Kubernetes

What is a Master node?

The Master is responsible for managing the cluster. The master coordinates all activities in your cluster, such as scheduling applications, maintaining applications' desired state, scaling applications, and rolling out new updates.

Each node has a Kubelet, which is an agent for managing the node and communicating with the Kubernetes master.



Kubernetes

What is a Pod ?

A Pod is the smallest deployable unit that can be created, scheduled, and managed. It's a logical collection of containers that belong to an application. Multiple applications may be run within a Pod typically different containers are for different applications. A Pod is a higher level abstraction for managing a group of containers with shared volumes and network namespace. All the applications (containers) in a Pod share the same filesystem and IP address with the port on which each application is exposed being different. Pods are run on nodes.

What Is a Label?

A Label is a key-value pair identifying a resource such as a Pod, Service, or Replication Controller: most commonly a Pod. Labels are used to identify a group or subset of resources for tasks such as assigning them to a Service. Ex: "app = helloApp"

Kubernetes

REPLICA SETS

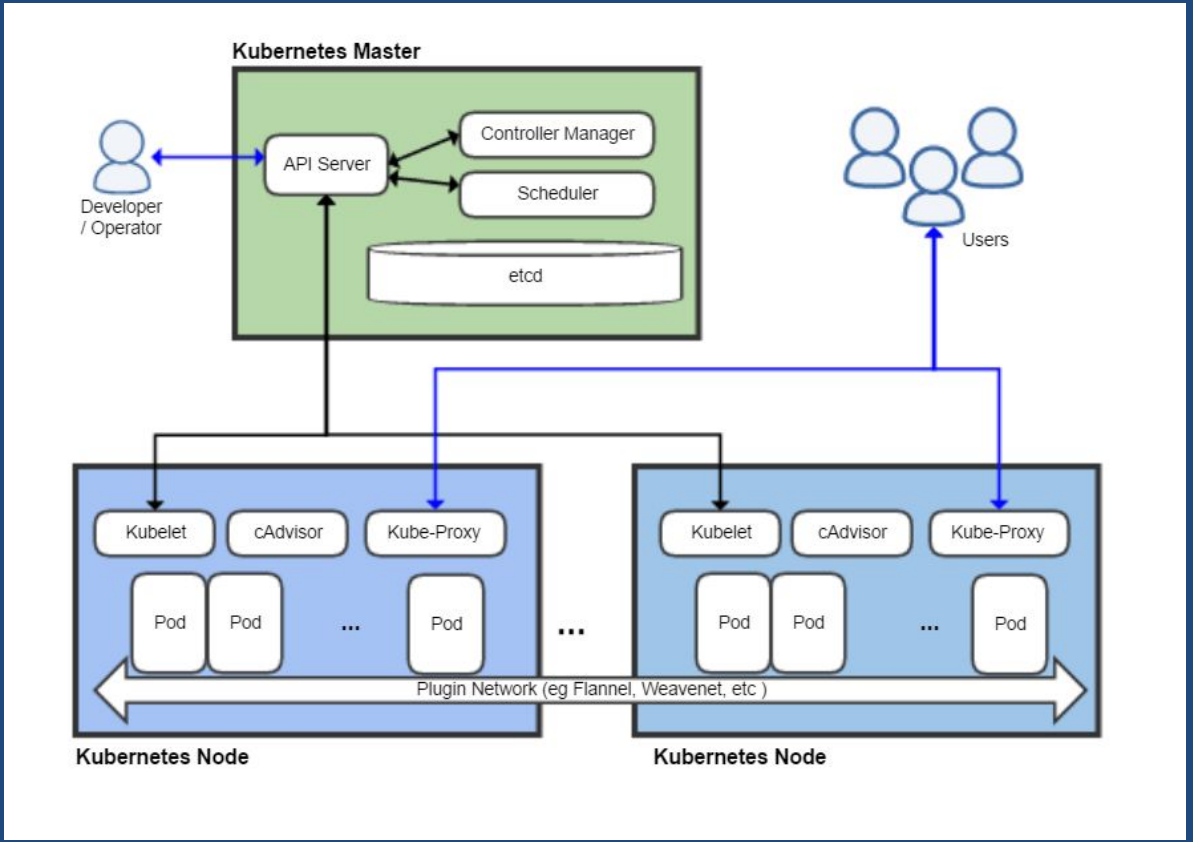
A replica set ensures that a specified number of pod replicas are running on worker nodes at any one time. It allows both up- and down-scaling the number of replicas. It also ensures recreation of a pod when the worker node reboots or otherwise fails.

Kubernetes Architecture

Key components

On Master (Control Plane)
Kube-API Server
Kube-Controller-Manager
Kube Scheduler
Etcd (deamon)

On Nodes
Kubelet
CAdvisor
Kube-Proxy



Kubernetes Architecture Components

The Kubernetes Master is the main controlling unit of the cluster that manages its workload and directs communication across the system.

etcd : A persistent lightweight distributed key-value data store that reliably stores the configuration data of the cluster, representing the overall state of the cluster.

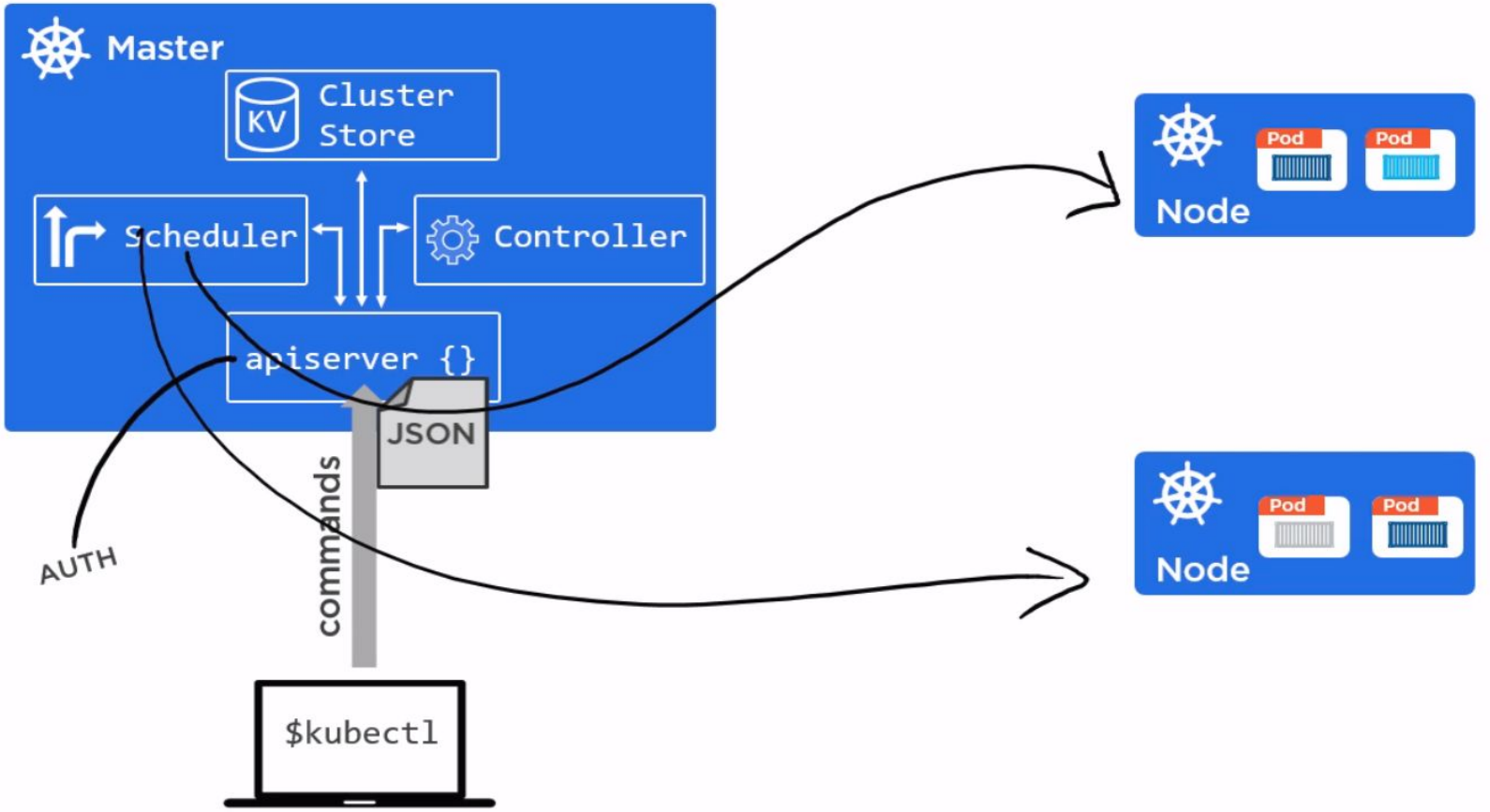
API Server: The API server is a key component and serves the Kubernetes API using JSON (via manifest files) over HTTP. Provides both the internal and external interface to Kubernetes. Front-end to the control plane. This is the only component the user directly deals with.

Kubernetes Architecture Components

Controller Manager : The Controller communicate with the API server to create, update and delete the resources they manage (pods, service endpoints, etc.) Assigns work to nodes.

Scheduler : The scheduler is the pluggable component that selects which node an unscheduled pod (the basic entity managed by the scheduler) should run on based on resource availability.

Kubernetes Interaction



Kubernetes Node

The Node, also known as Worker or Minion, is a machine where containers (workloads) are deployed. Every node in the cluster must run a container runtime such as **Docker**, as well as the below-mentioned components, for communication with master for network configuration of these containers.

Kubelet is responsible for the running state of each node, ensuring that all containers on the node are healthy. It takes care of starting, stopping, and maintaining application containers organized into pods as directed by the control plane. Kubelet monitors the state of a pod and if not in the desired state, the pod will be redeployed to the same node. The node status is relayed every few seconds via heartbeat messages to the master. Once the master detects a node failure, the Replication Controller observes this state change and launches pods on other healthy nodes. Registers node with the cluster.

Kubernetes Node

A **Container** resides inside a Pod. The container is the lowest level of a micro-service which holds the running application, the libraries and their dependencies. Containers can be exposed to the world through an external IP address.

The **Kube-proxy** is an implementation of a network proxy and a load balancer, and it supports the service abstraction along with other networking operation.It is responsible for routing traffic to the appropriate container based on IP and port number of the incoming request.

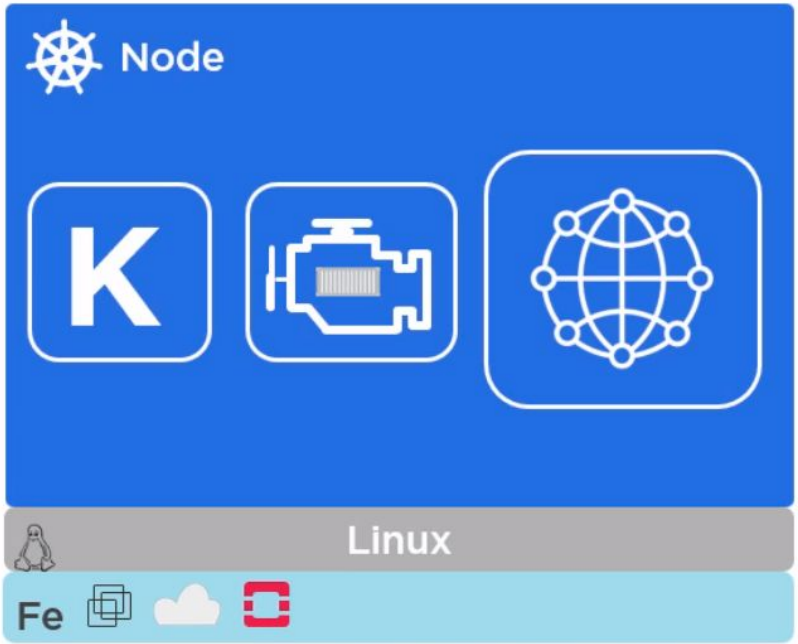
cAdvisor is an agent that monitors and gathers resource usage and performance metrics such as CPU, memory, file and network usage of containers on each node.

Kubernetes Node

Kubelet
Main Kubernetes agent

Container engine
Docker or rkt

kube-proxy
Kubernetes networking



Kubernetes Minicube

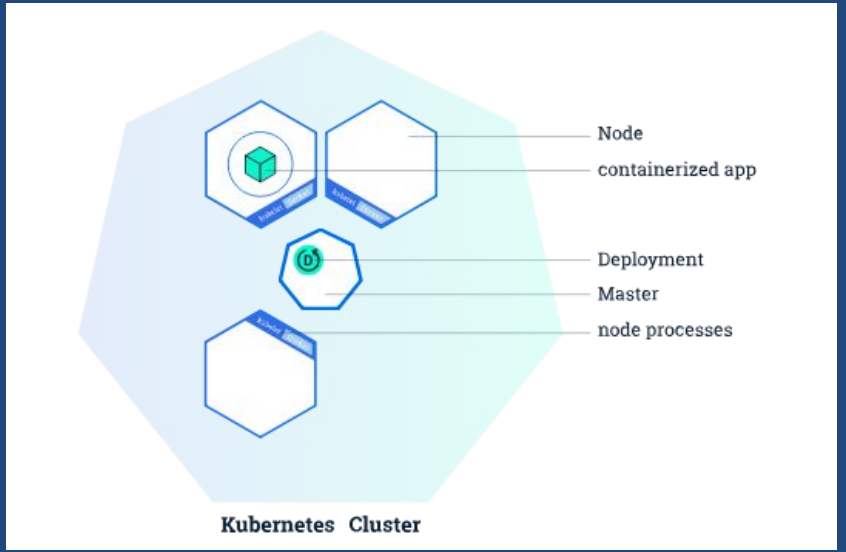
Minikube is a tool that makes it easy to run Kubernetes locally.
Minikube runs a single-node Kubernetes cluster inside a VM on your laptop for users looking to try out Kubernetes or develop with it day-to-day.

Tutorial
https://kubernetes.io/docs/tutorials/kubernetes-basics/create-cluster/cluster-interactive/

Kubectl (Deploying App)

kubectl is a command line interface for running commands against Kubernetes clusters. This overview covers kubectl syntax, describes the command operations, and provides common examples.
For details about each command, including all the supported flags and subcommands, see the kubectl reference documentation.

We will use kubectl for our tutorials



Kubernetes (Deploying App)

Let us deploy an app.

https://kubernetes.io/docs/tutorials/kubernetes-basics/deploy-app/deploy-interactive/