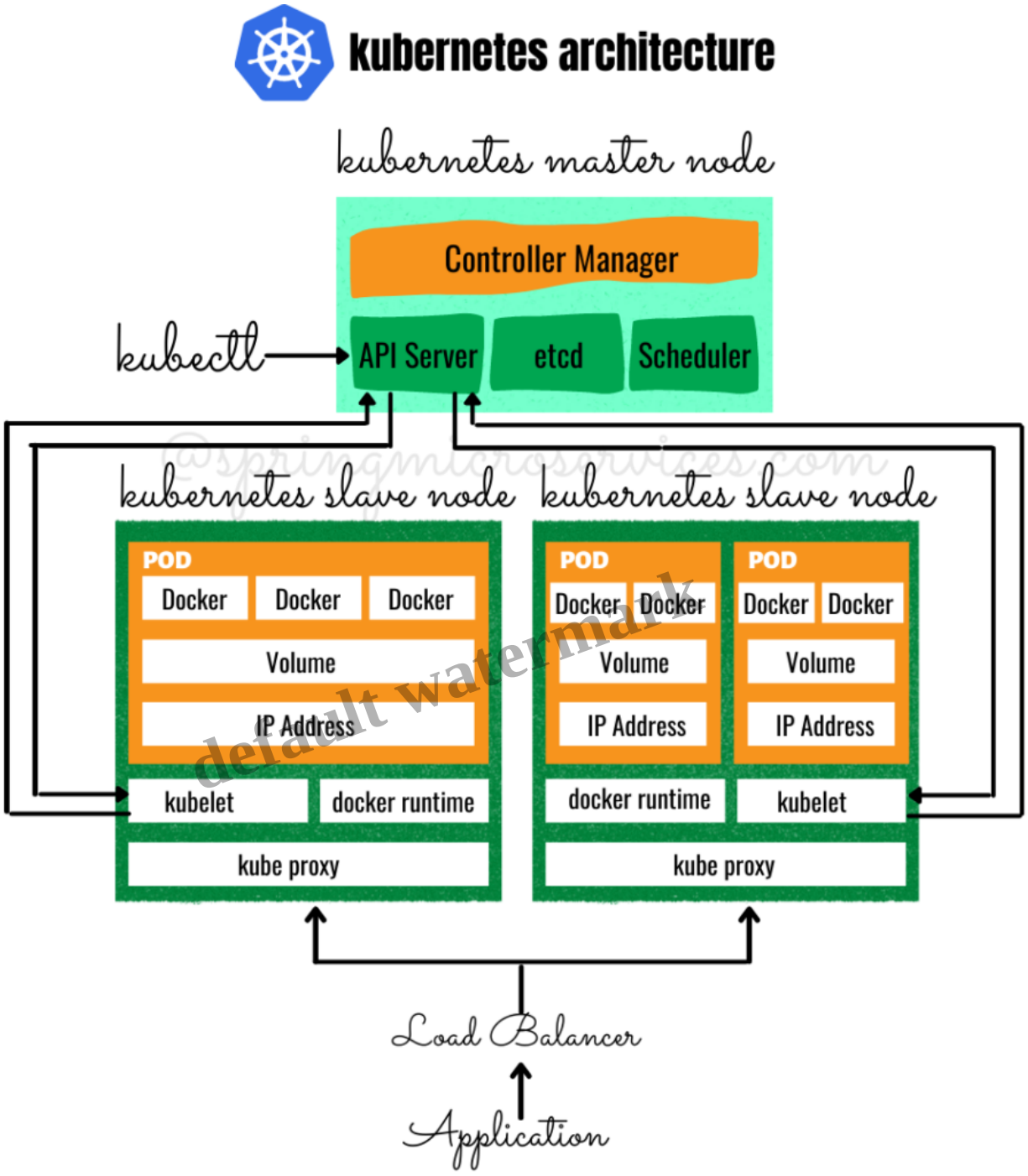
What is Kubernetes Architecture

Description

In this post we are going to have a look at the kubernetes architecture and its components. If you haven't read about my previous post on what is kubernetes, then I would recommend you to have a quick look at it. To sum up in a single line, "*Kubernetes is an open source container management tool which is used to automate container deployment, It can also scale up, scale down and load balance containers inside the ecosystem.*"

Kubernetes follows the master slave architecture. In the below diagram there are 3 servers, 1 is the kubernetes master and the 2 are the kubernetes slave nodes. A cluster is made up of multiple machines, each machine is called a node. The machine that is responsible for managing the cluster is called the master node and the machine where the bulk or most of the work is done are called worker nodes or slave nodes



Lets look at each component.

Kubernetes master node

Kubernetes master node manages the entire cluster. We can have a single or multiple master nodes depending on the type of environment we are operating on. For example Development environmentcan have a single master node and production environment can have multiple master nodes. Letsexplore each component.

API server

API server is the heart of the entire cluster. The **API Server** exposes RESTful APIs so that we can interact with the master node. The way we can fire the calls is using **kubectl**. It is basically a command line utility tool which takes yml or json files that we provide, and it in turn hands it over to the API server, the API server will take the request, validate and process by performing the task against the worker nodes. **kubectl** is just the wrapper around the curl command which makes the http call.

Scheduler

Scheduler will schedules the pods on the worker nodes. It has all the information about these worker nodes and the pods in them. Scheduler is responsible for launching the pods inside these worker nodes whenever required.

Control manager

Control manager runs in the background. The Control manager is responsible for making sure the cluster is in the desired state. For example, If we want 5 pods to be up and running, and within those 5 pods we always need to have 2 containers in each of those pods, then it is the responsibility of the control manager to make sure that the cluster is in the desired state as requested.

etcd

etcd is a name value storage, information about all the objects like pods and container is stored as name value pair in etcd. Since we can have multiple master nodes, etcd will be stored in each of those master nodes so that even if 1 master node goes down we have the state of the cluster in another master node.

Kubernetes Slave node

Kubernetes slave node is also called as a worker node where most of the work is done. They are also responsible for launching the pods or creating the containers. Kubernetes master node controls the slave nodes.

kubelet

There is a bidirectly communication between API server and the kubelet. Both can interact with each other. API server can interact with the kubelet to setup the number of pods no of containers etc. We never interact with the kubelet directly, it will always be through the API server. The communication flow is like – kubectl interacts with the API server and the API server will interact with the kubelet to get

all the setup done.

kubectl – API server – kubelet

kube proxy

Kube proxy is like an abstraction layer which acts as a load balancer. When the external applications want to communicate with the containers that are running, then they have to go through the kube proxy and that is how you can talk with the containers running inside the nodes..

load balancer

This is the external load balancer which will distribute the load across the worker nodes instead of all requests going to a single worker node.

Docker Runtime

Docker runtime is required to host docker containers inside a node.

Pods

You can learn in depth about pods in our separate post **what are pods in Kubernetes**.

Volumes

You can learn in depth about pods in our separate post **what are volumes in Kubernetes**.

Category

1. Cloud
2. Kubernetes

Date Created
January 2, 2022
Author
kk-ravi144gmail-com