## Kubernetes Introduction

In modern web applications, customers often expect applications to be available at any time, and developers of applications often expect to deploy new versions of applications multiple times a day without affecting applications’ users. An very easy and fast way to release and update these applications is containerization. To ease the management and deployment of containerized applications, we can use Kubernetes, a production-ready and open source platform, for application management and deployment.

Kubernetes can coordinate a cluster of connected computers to work as a single unit. The high-level abstractions of Kubernets enable us to deploy containerized applications to a cluster without deploying them specifically to each individual computers. To make use of this novel type of deployment, applications need to be containerized, that is, they need to be packaged in a way that decouples them from individual hosts. In addition, Kubernetes can automate the distribution and scheduling of our application containers in a cluster in a more efficient way.

## Kubernetes Cluster Diagram

A Kubernetes cluster can be deployed on either physical or virtual machines. A Kubernetes cluster consists of two types of resources -- a Master and multiple Nodes.

The Master is responsible for managing the Kubernetes cluster. It coordinates all activities in the cluster, including scheduling applications, maintaining applications’ desired state, scaling applications and rolling out new updates. A node is a virtual machine or a physical computer that serves as a worker machine in a Kubernetes cluster. Each node has a Kubelet for managing this node and communicating with the Kubernetes cluster master. Each node also needs to contain some tools for handling container operations, like Docker. A minimum of three nodes are needed for a Kubernetes cluster to handle production traffic. A typical diagram of a Kubernetes cluster is shown below in Figure 1.

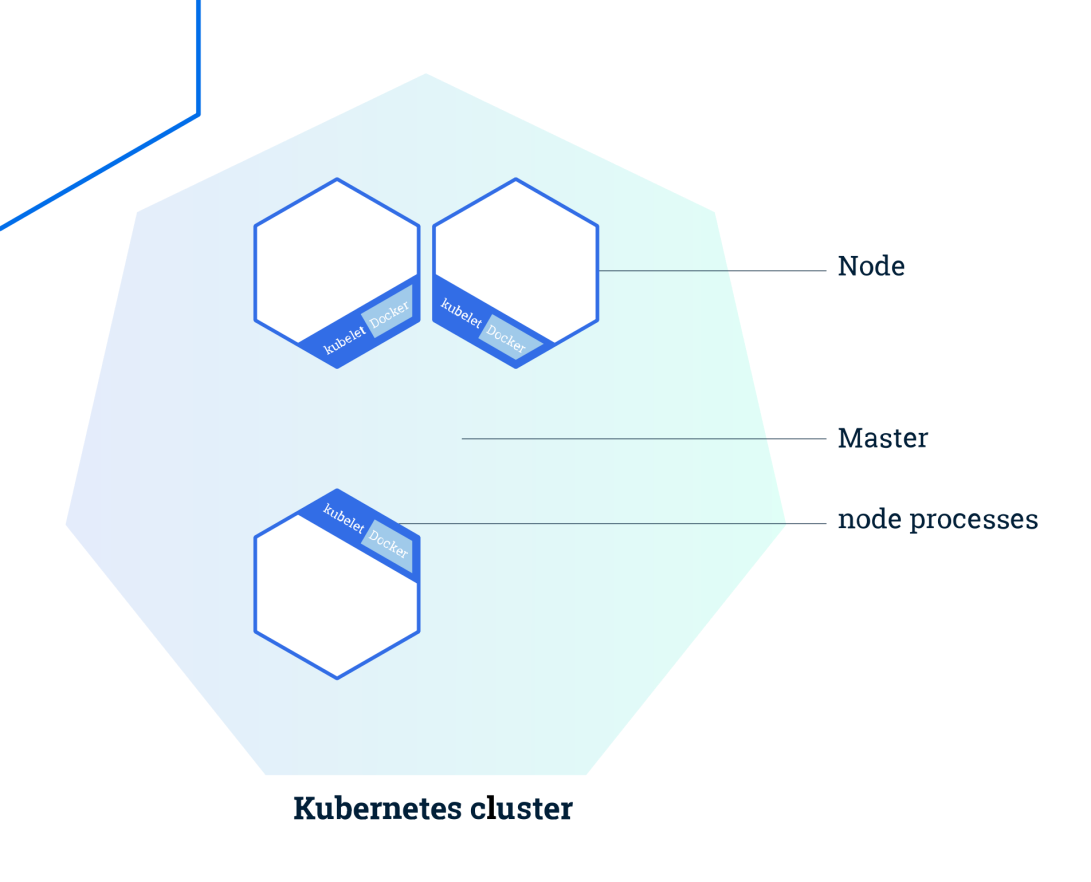


Figure 1. Diagram of a Kubernetes cluster

After we deploy the applications on Kubernetes, we tell the master to start the application containers. The master schedules the containers to run on the cluster’s nodes. The nodes communicate with the master using the Kubernetes API exposed by the master. We can also use the Kubernetes API directly to interact with the cluster.

## Kubernetes Deployment

Once we have a running Kubernetes cluster, we can deploy our containerized applications in this cluster. We need to create a Kubernetes Deployment configuration to do it. The Deployment instructs Kubernetes how to create and update instances of our application. After the Deployment is created, the Kubernetes master schedules multiple application instances onto different Nodes of the Kubernetes cluster.

A particular advantage of using Kubernetes is the self-healing mechanism. If a Node hosting an application instance goes down or is deleted in a running Kubernetes cluster, the Deployment controller will automatically replaces it with an instance on another Node. This function of recovery from machine failure has not been covered by most other application management systems.