Introduction

CLARUSWAY©
WAY TO REINVENT YOURSELF

Table of Contents

Table of Contents

1

What is Kubernetes?

What is Kubernetes?

- Born in Google
- Donated to CNCF in 2014
- Open source (Apache 2.0)
- v1.0 July 2015
- Written in Go/Golang
- Often shortened to k8s

~~Kubernetes~~ s

K8 s

CNCF: Cloud Native Computing Foundation

What is Kubernetes?

What is Kubernetes?

- Kubernetes is **Open Source Orchestration** system for Containerized Applications.
- Kubernetes is a platform that **eliminates the manual processes** involved in **deploying** containerized applications.
- Kubernetes is used to manage the **State of Containers**.
    - Start Containers on Specific Nodes.
    - Restart Containers when gets Killed.
    - Move containers from one Node to Another.

2

Why you need Kubernetes?

Why you need Kubernetes?

Containers are a perfect way to get the applications **packaged** and **run**. **In production** environment, you should manage the containers that run the applications and **ensure no downtime**.

Why you need Kubernetes?

Kubernetes supplies you with:

- Service discovery and load balancing
- Storage orchestration
- Automated rollouts and rollbacks
- Automatic bin packing
- Self-healing
- Secret and configuration management

Features of Kubernetes

➢ **Automated Scheduling :** Kubernetes provides advanced scheduler to launch container on cluster nodes **based on** their **resource requirements** and other constraints.

➢ **Healing Capabilities:** Kubernetes allows to **replace and reschedule** containers when nodes die. K8s doesn't allow Containers to use a node until they get ready.

➢ **Auto Upgrade and RollBack :** Kubernetes **rolls out changes** to the application. K8s **doesn't kill** the containers **all at once**. If something goes wrong, you can **rollback the change**.

Features of Kubernetes

➢ **Horizontal Scaling :** K8s can **scale up** and **scale down** the application as per the requirements **with a simple command**, using a UI, **or automatically** based on CPU usage.

➢ **Storage Orchestration :** With K8s, you can mount the storage system of your choice. You can either opt for **local storage**, or choose a public **cloud provider**.

➢ **Secret & Configuration Management :** K8s can help you **deploy and update secrets** and application **configuration without rebuilding your image** and without exposing secrets in your stack configuration.

Features of Kubernetes

You can Run Kubernetes Anywhere:

- On-Premise (Own DataCenter)
- Public Cloud (Google, AWS, Azure, DigitalOcean…)
- Hybrid Cloud

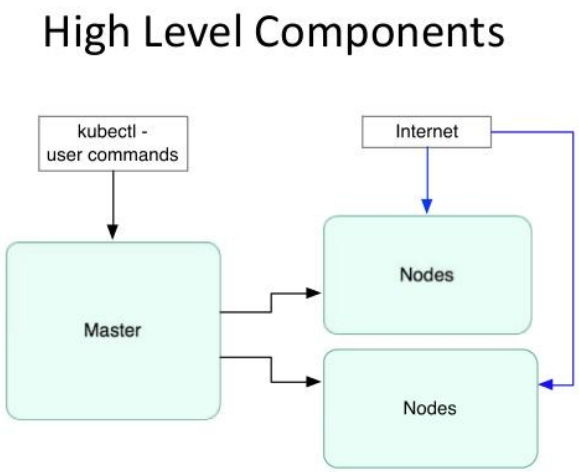
Docker Swarm vs Kubernetes

The differences of Kubernetes and Docker Swarm

Docker Swarm	Kubernetes
1 No Auto Scaling	1 Auto Scaling
2 Good community	2 Great active community
3 Easy to start a cluster	3 Difficult to start a cluster
4 Limited to the Docker API's capabilities	4 Can overcome constraints of Docker and Docker API
5 Does not have as much experience with production deployments at scale	5 Deployed at scale more often among organizations

4

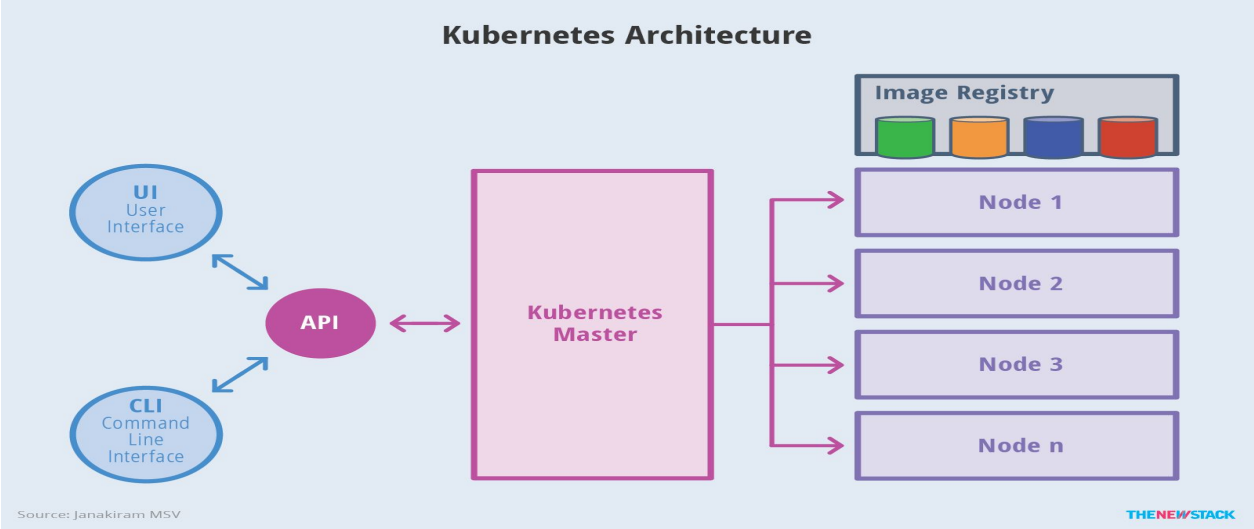
Kubernetes Components



Kubernetes Components

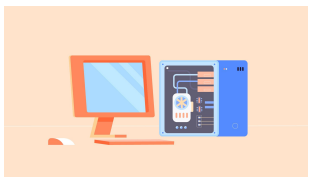
Kubernetes has the following main components:

- One or more master nodes
- One or more worker nodes.



Kubernetes Nodes

- A **node** is a **worker** machine in Kubernetes.
- A node may be a **VM or physical machine**, depending on the cluster.
- Each node contains the **services** necessary to run pods.
- Nodes Primarily managed by **Kubernetes Master**.
- Single Master Can manage **~5000 Worker Nodes**.



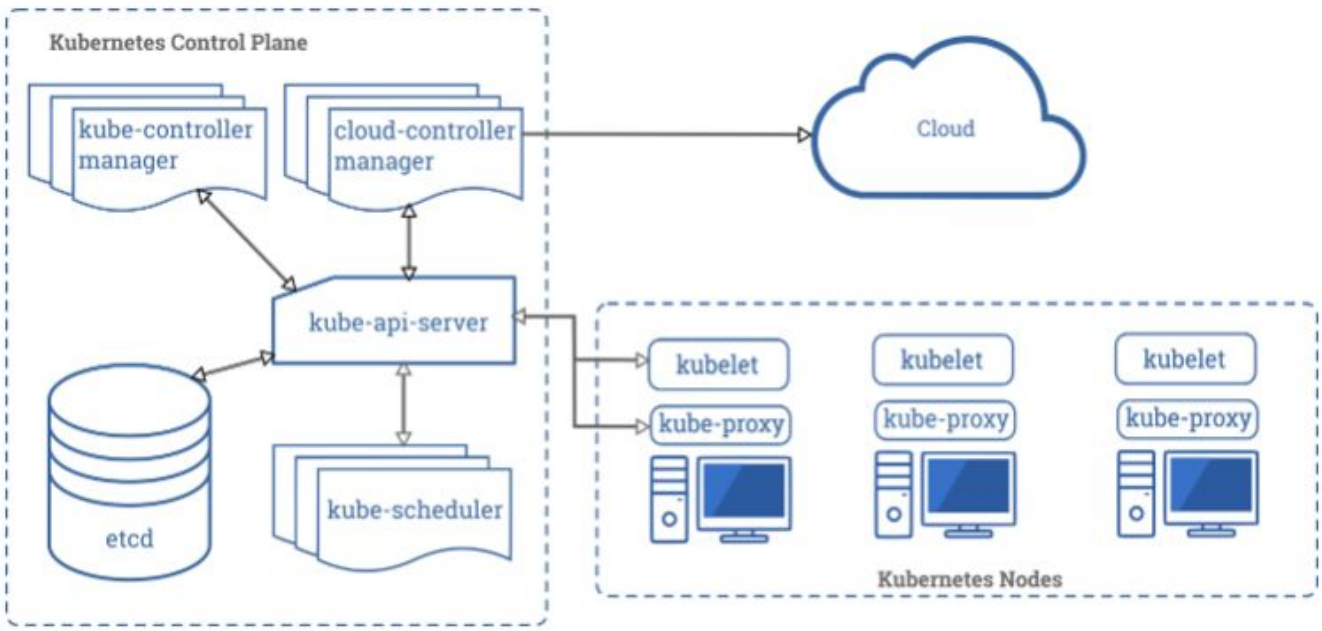
Kubernetes Master Component

➢ To understand the Kubernetes Administration, we should understand the **Architecture of Master** and the **workflow of Master Components**.

➢ Kubernetes master runs the **Scheduler**, **Controller Manager**, **API Server** and **etcd** components and is responsible for managing the Kubernetes cluster.

➢ You can say the **master** is the **brain** of the Kubernetes Cluster.

Control Plane Components



Control Plane Components

kube-apiserver:

- Provides a REST interface into the kubernetes control plane and datastore.
- Clients and applications interact with k8s strictly through the API Server.
- Acts as the gatekeeper to the cluster by handling authentication and authorization, request validation in addition to being the front-end to the backing datastore.

Control Plane Components

etcd:

- etcd acts as the cluster datastore.
- Aims to provide a strong, consistent and highly available key-value store for persisting cluster state.
- Stores objects and config information.

Control Plane Components

kube-controller-manager:

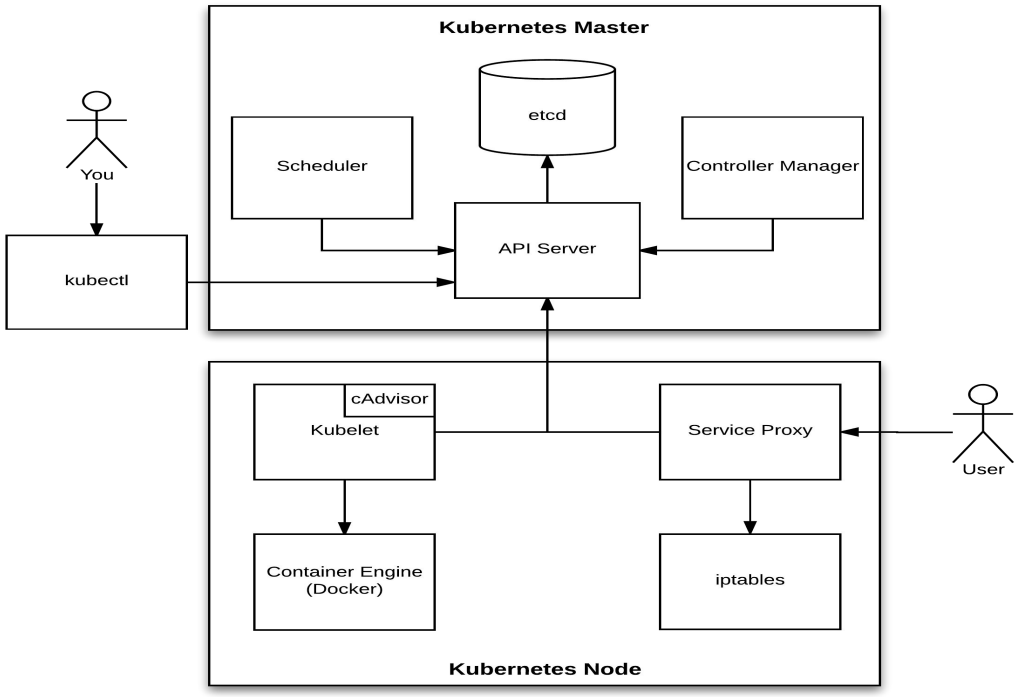
- serves as the primary daemon that manages all core component control loops.
- monitors the cluster state via the apiserver and steers the cluster towards the desired state

Control Plane Components

kube-scheduler:

- is the engine that evaluates workload requirements and attempts to place it on a matching resource.
- is the component that uses bin packing.
- Workload Requirements can include: general hardware requirements, affinity/anti-affinity, labels, and other various custom resource requirements.

Node Components



Node Components

kubelet:

- Acts as the node agent responsible for managing the lifecycle of every pod on its host.
- Understands YAML container manifests that it can read from several sources:
  - API Server
  - File Path
  - HTTP Endpoint
  - HTTP Server mode accepting container manifests over a simple API.

Node Components

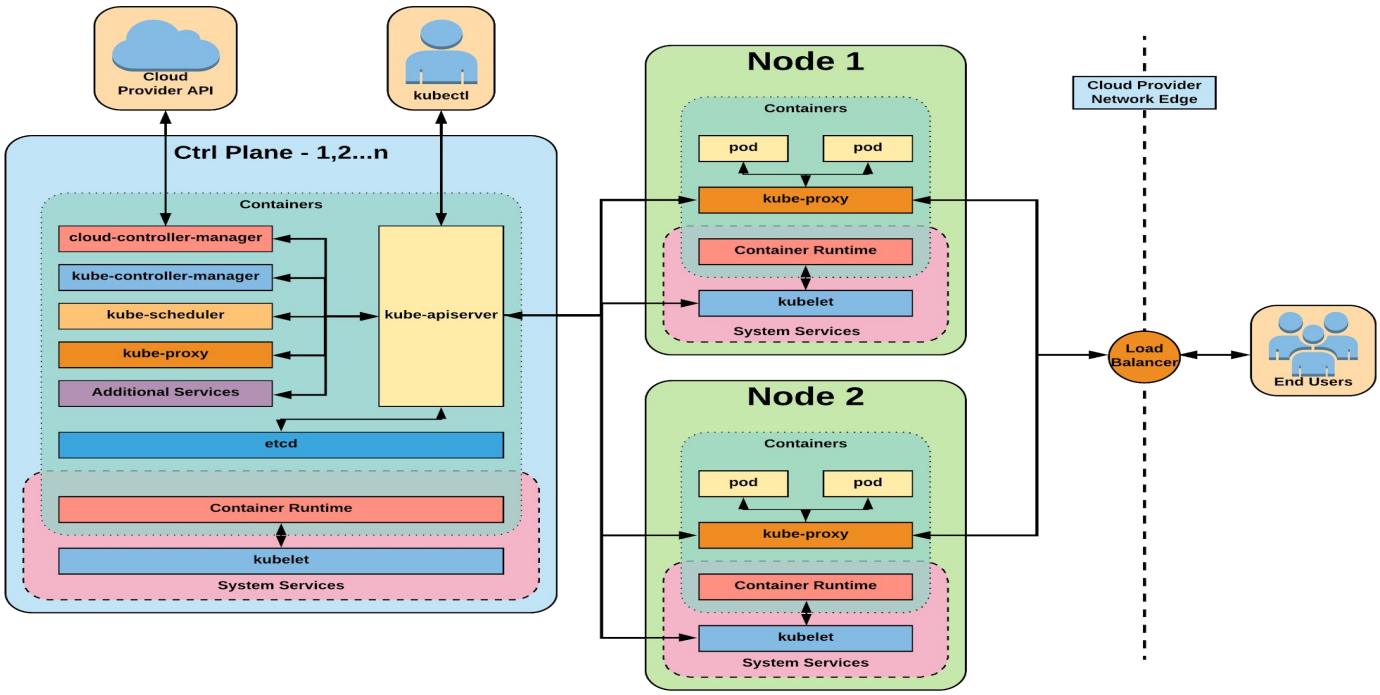
kube-proxy:

- Manages the network rules on each node.
- Performs connection forwarding or load balancing for Kubernetes cluster services.
- Available Proxy Modes:
  - user space
  - iptables (default)
  - ipvs

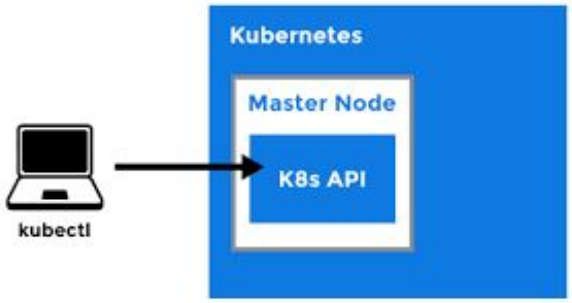
Node Components

Container Runtime Engine:

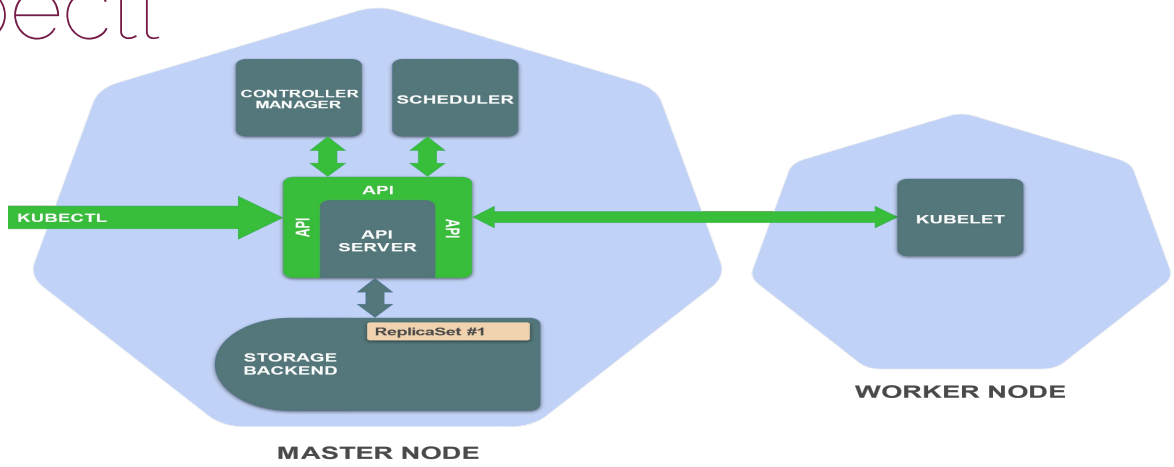
- A container runtime is a CRI (Container Runtime Interface) compatible application that executes and manages containers.
  - Containerd (docker uses this)
  - Cri-o
  - Rkt
  - Kata (formerly clear and hyper)
  - Virtlet (VM CRI compatible runtime)


Cloud Provider API
kubectl
Ctrl Plane - 1,2...n
Containers
cloud-controller-manager
kube-controller-manager
kube-scheduler
kube-proxy
Additional Services
kube-apiserver
etcd
Container Runtime
kubelet
System Services
Node 1
Containers
pod
pod
kube-proxy
Container Runtime
kubelet
System Services
Node 2
Containers
pod
pod
kube-proxy
Container Runtime
kubelet
System Services
Cloud Provider Network Edge
Load Balancer
End Users

5 kubectl


Kubernetes
Master Node
K8s API
kubectl

kubectl



- **kubectl** is (almost) the only tool we'll need to talk to Kubernetes
- It is a rich CLI tool around the Kubernetes API
- Everything you can do with kubectl, you can do directly with the API
- kubectl can be pronounced "Cube C T L", "Cube cuttle", "Cube cuddle".

THANKS!

Any questions?

- tyler@clarusway.com
- serdar@clarusway.com