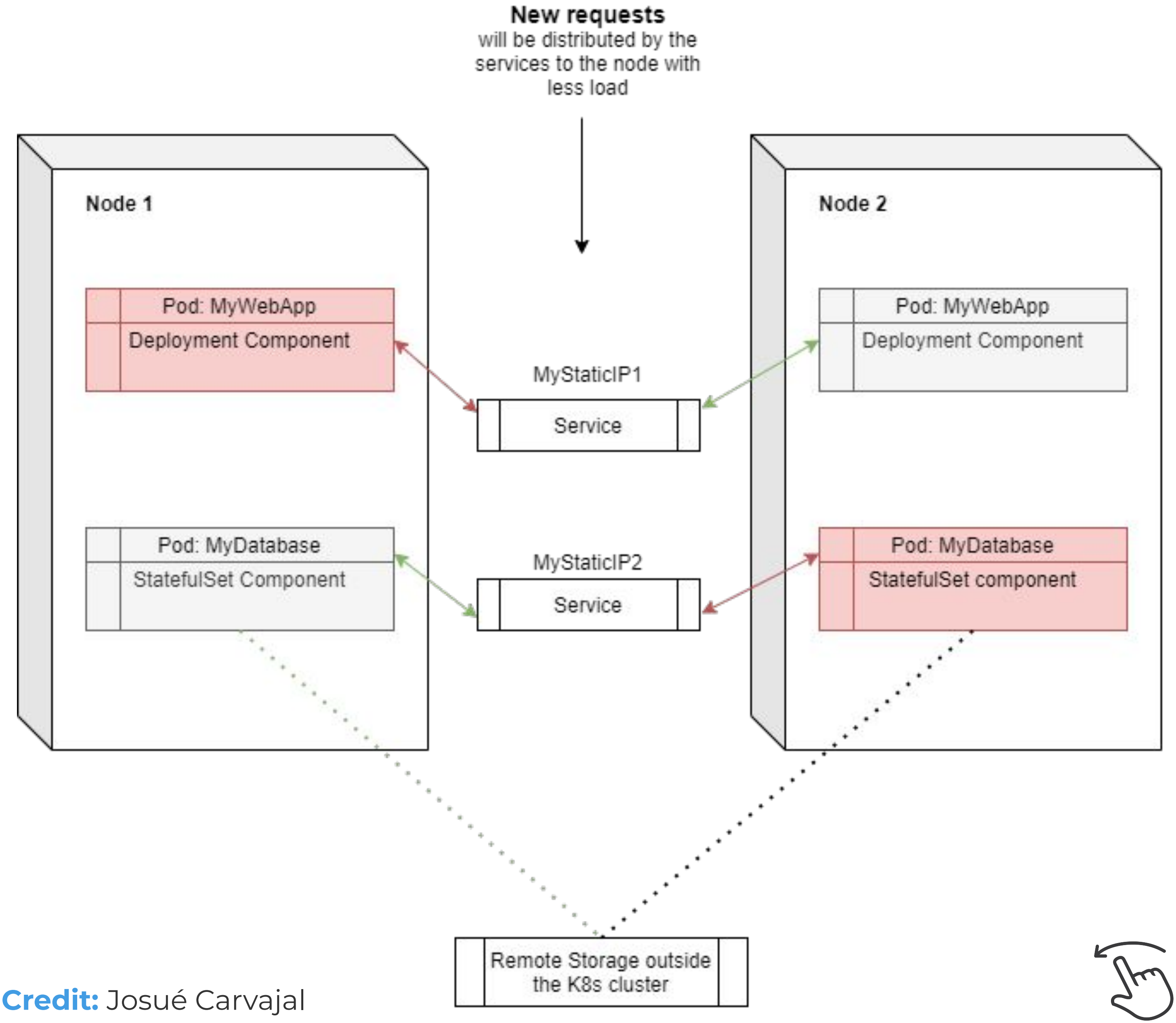
Introduction to Kubernetes Components and K8s Architecture



New requests will be distributed by the services to the node with less load

Node 1

Pod: MyWebApp

Deployment Component

Pod: MyDatabase

StatefulSet Component

MyStaticIP1

Service

MyStaticIP2

Service

Node 2

Pod: MyWebApp

Deployment Component

Pod: MyDatabase

StatefulSet component

Remote Storage outside the K8s cluster

Credit: Josué Carvajal

Node

A node will contain an **N number of pods**. Usually, it is a physical server, virtual machine, or any server.

Pod’s

A Pod is the smallest unit of Kubernetes, which is an abstraction of what we knew as a container, it will run only one application on each Pod. One important thing to point out is that **each pod has its own IP address and a new IP address will be added per creation** of a pod after it dies, this is usually something that we do not want and can be solved with Kubernetes Service.

Service

A service allows communication between pods by adding a permanent IP to the pod, this means if the pod dies **it will conserve its IP**. It also acts as a load balancer by locating the less loaded POD and send those requests to it.

Ingress

The ingress route the traffic into the cluster, this is usually the first service that is reached, **can be used for WEB servers that interact** with the end-user.

Config Map

It is an external configuration file to our application that we use to configure our Pods therefore all the pods can read this configuration file **without the need of recreating or rebuilding** the pods.

Config Map — Secrets

The secret is a configuration file that is meant to store sensitive information such as passwords. It uses **base64 encoding format**.

Volumes

Each time a node is restarted the data on it is deleted, to avoid this Kubernetes has the Volumes component, **which is meant to get persistence**.

Pod Blueprint

Defines the pod and its replicating mechanisms.

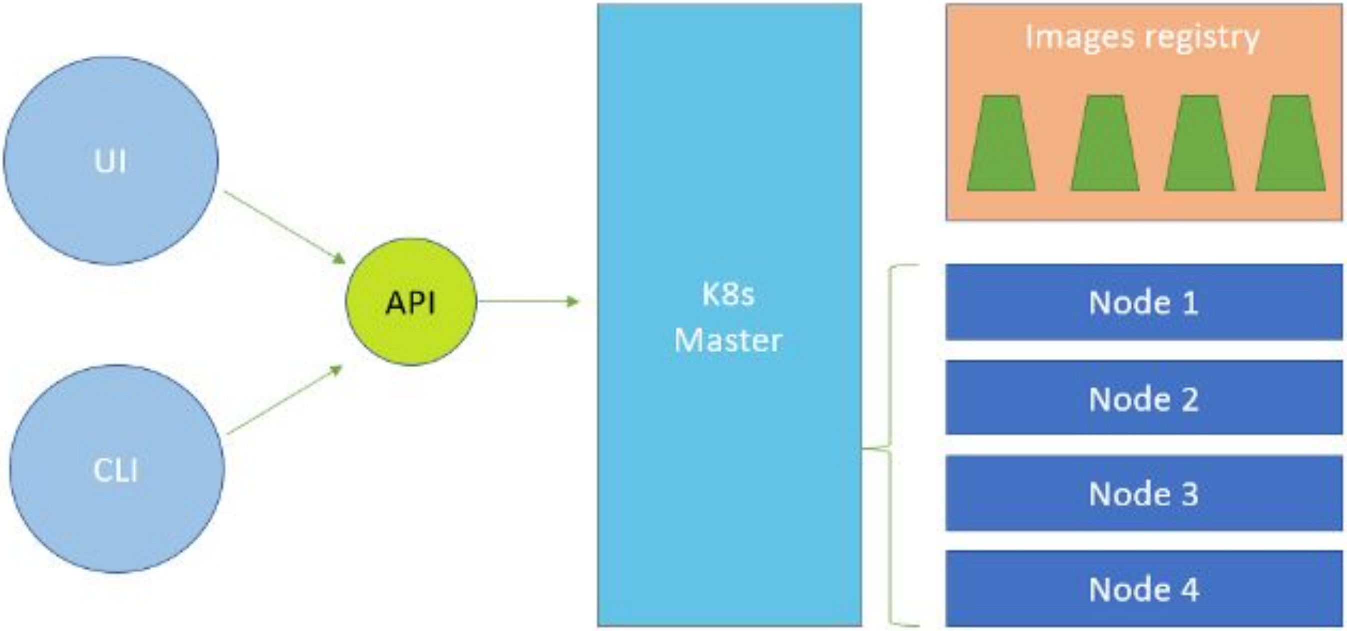
Deployment

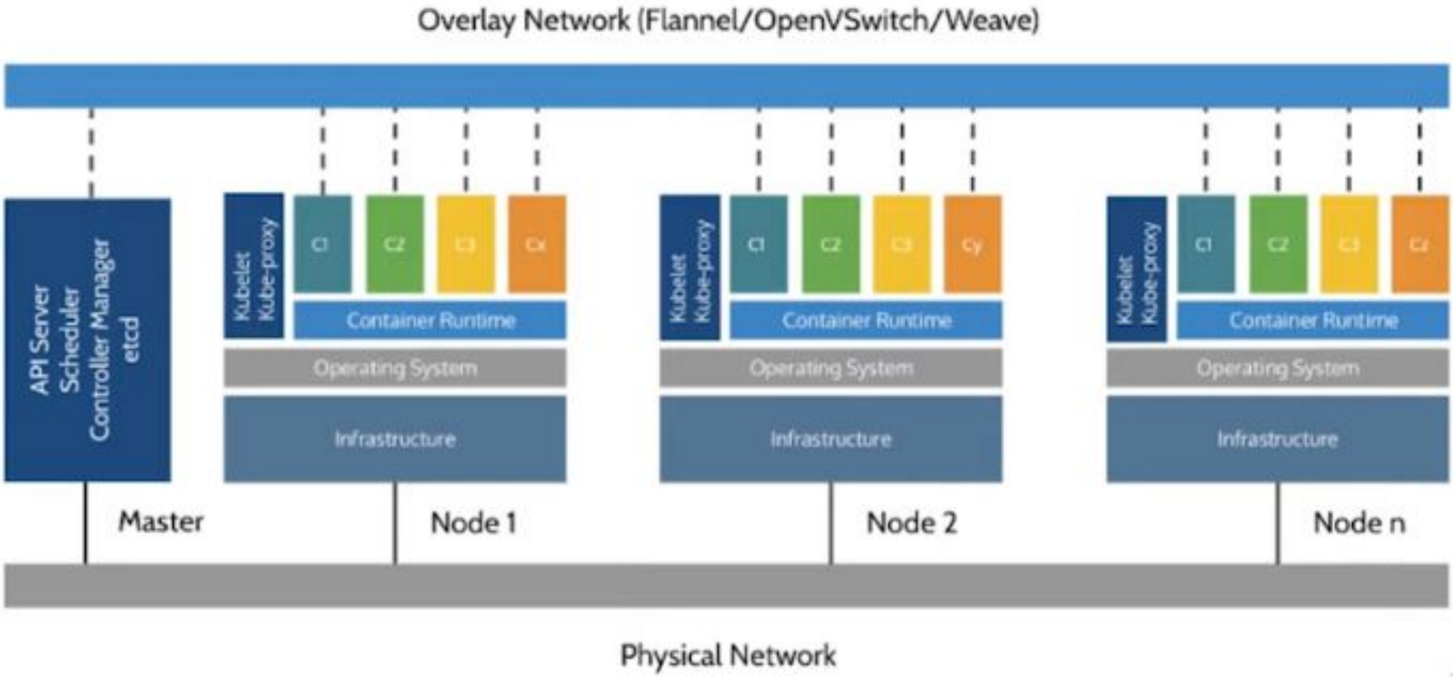
This is how we deploy new pods, it is meant for stateless applications (**applications that do not store data**).

Stateful

This is meant for stateful applications **such as databases**, this must be used in this way since it avoids data inconsistency when two replicated nodes sharing a database pod are accessing the same database, how do Kubernetes can ensure there is data consistency, well by using stateful component and making sure who is using the storage at a given point of time.

Kubernetes architecture




Overlay Network (Flannel/OpenVSwitch/Weave)
API Server
Scheduler
Controller Manager
etcd
Master
Kubelet
Kube-proxy
C1
C2
C3
Cx
Container Runtime
Operating System
Infrastructure
Node 1
Kubelet
Kube-proxy
C1
C2
C3
Cy
Container Runtime
Operating System
Infrastructure
Node 2
Kubelet
Kube-proxy
C1
C2
C3
Cz
Container Runtime
Operating System
Infrastructure
Node n
Physical Network

API Server

Here is were both interfaces are connected (**UI/CLI**) and it uses the raft algorithm to select the next master if the master node dies. It is the brain of the cluster and what get lost when we lost the master

Scheduler

Orchestrates the scheduled jobs in the pods and tracks the resource consumption.

Controller manager

Uses control loops and it based on a declarative modes to reach out to a desired stated.

etcd

It is a highly available database based on key-value store.

Container Runtime

In the latest versions of kubernetes it uses dockerd which is the core of docker.

Kubelet

It is Kubernetes Agent that connects to the control plane (Master node), requests and monitor resources to run in the nodes and also **updates the control plane** with the state of the nodes.

Kube-proxy

Load balance in our container to send it to the less loaded node.