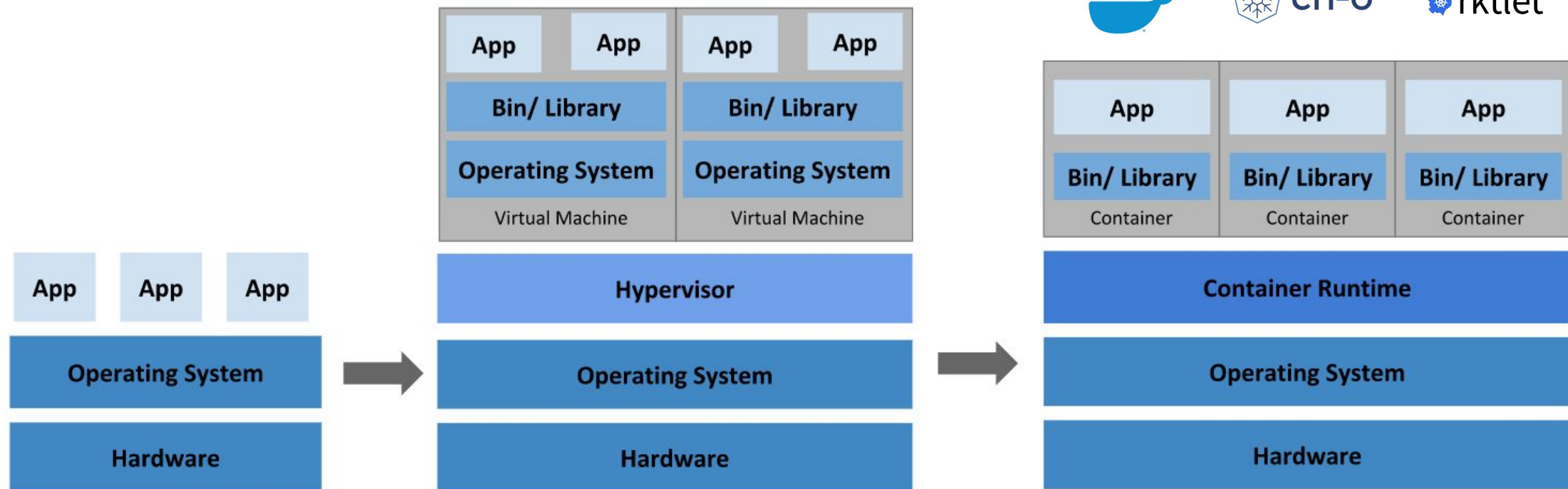


Kubernetes for Developers

John Bush
Solutions Engineer
VMware, Inc.

Back To The Future ...



Traditional Deployment

- Organizations ran applications on physical servers
- No way to define resource boundaries
- Not scalable
- Problematic in many ways

Virtualized Deployment

- Run multiple Virtual Machines (VMs) on a single physical server's CPU
- Allows applications to be isolated between VMs
- Need Hypervisor to run the VM
- Provides a level of security as the information of one application cannot be freely accessed by another application.
- Each VM is a full machine running all the components, including its own operating system, on top of the virtualized hardware - Heavyweight (GB in size)

Container Deployment

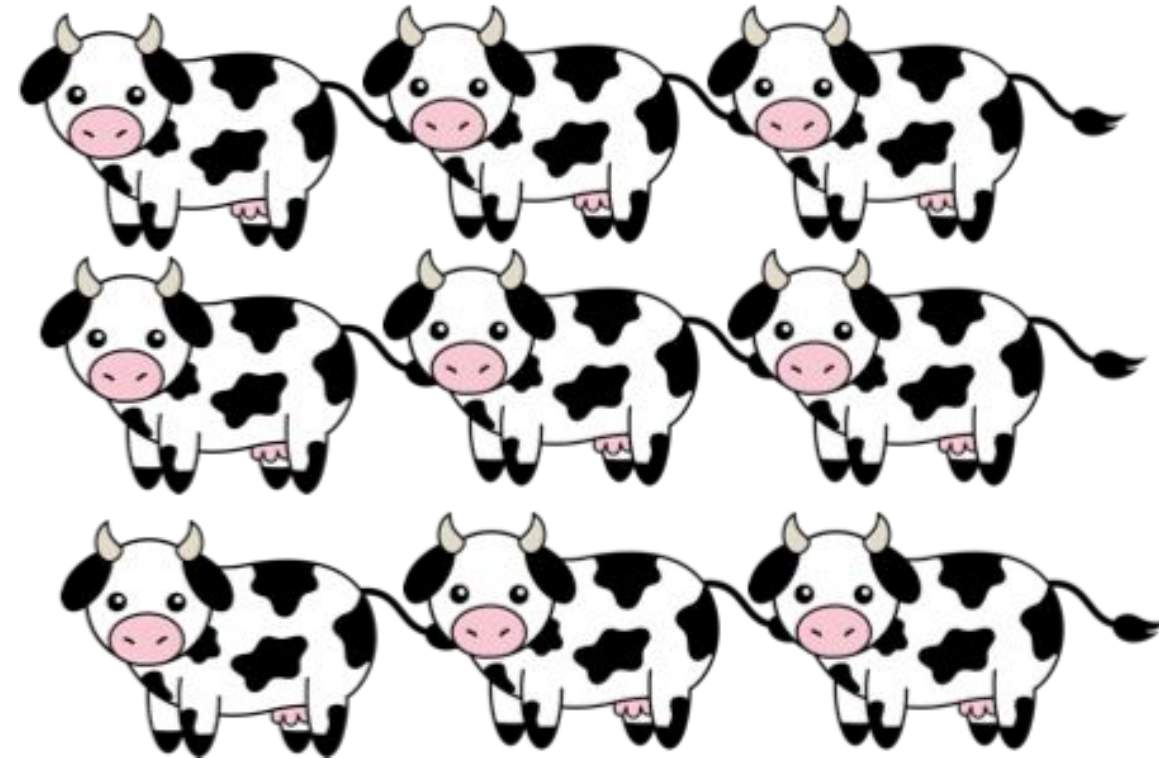
- Similar to VMs, but they have relaxed isolation properties to **share** the Operating System (OS) among the applications - lightweight (MB in size)
- Has its own filesystem, CPU, memory, process space, and more.
- Portable across clouds and OS distributions.
- Increased ease and efficiency of container image creation compared to VM image use.
- **Raises the level of abstraction from running an OS on virtual hardware TO running an application on an OS**
- Great way to bundle and run applications

Think Of VMs vs. Containers As “Pets Vs. Cattle”



Virtual Machine

Long lived
You name them
When they get ill, you nurse them back to health



Container

Ephemeral
You brand them with #'s
When they get ill, you get another one

Things Container Technologies Can't (Or Won't) Do ...



- Solve Port Mapping Hell
- Monitor Running Containers
- Handle Dead Containers
- Move Containers to Improve Utilization
- Autoscale Container Instances To Handle Load

kubernetes (n.) - *greek word for pilot or helmsman*



Also called “K8s”



Back To The Cattle vs. Pets Thing ...



And you can never get away from
“Pets”

UNLESS

You have an environment to
manage the “Cattle” ...



So what IS kubernetes?



A Production-Grade **Container Orchestration** System

Originally designed by **Google** (2014) and **donated** to the Cloud Native Computing Foundation (2015).

Portable and **extensible** open-source platform for managing containerized workloads.

It aims to provide a platform for **automating deployment, scaling, and operations** of application containers across **clusters** of hosts.

Principle: Manage your applications like **Cattle** instead of like Pets

From Google Trends - Searches On “Kubernetes”

● **kubernetes**
Search term

+ Compare

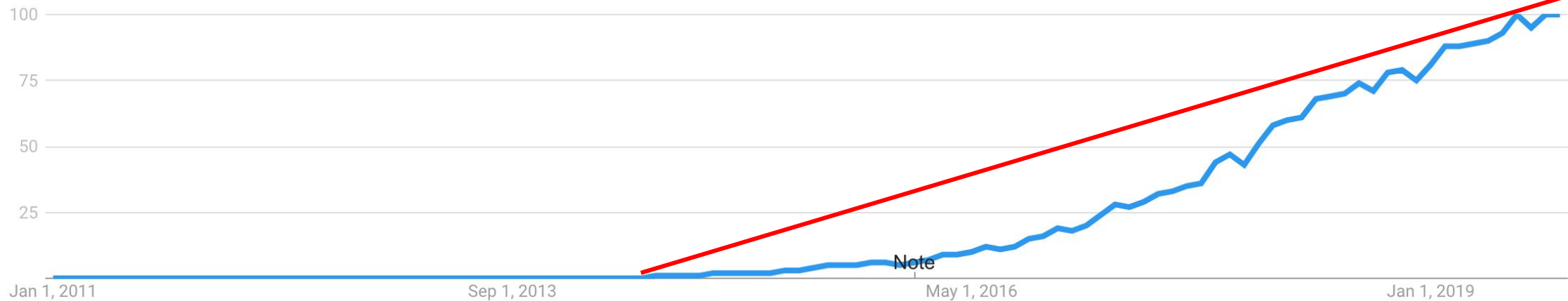
Worldwide ▼

1/1/11 - 11/7/19 ▼

All categories ▼

Web Search ▼

Interest over time ⓘ



So Really Why Is Kubernetes So Popular?

It's so much more than just enabling a containerized application to scale.

Kubernetes features:

- It's vendor agnostic

- Updates with near-zero downtime

- Version rollback

- Clusters that can 'self-heal' when there is a problem

- Load balancing, auto-scaling and SSL can easily be implemented.

- Helm, a plugin for Kubernetes, is a package manager that makes deployment of apps and its dependencies very easy

- Remember the "Things that Container technologies can't / won't do?" slide ?

And it does all of the above with speed, power and efficiency !!!



Every cloud supports Kubernetes



A re-architecture of the vSphere server virtualization platform that turns vSphere into a Kubernetes native platform.

Want To Install Kubernetes?

Simplest

Most
involved



Minikube



Google
Container
Engine
(GKE)



AWS
Provider



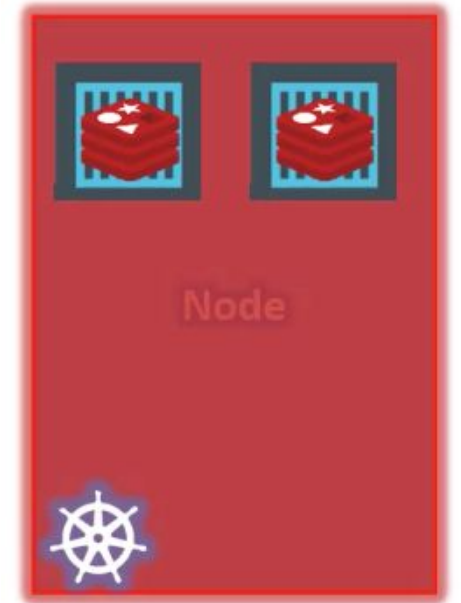
Manual
install

K8s Architecture Basics



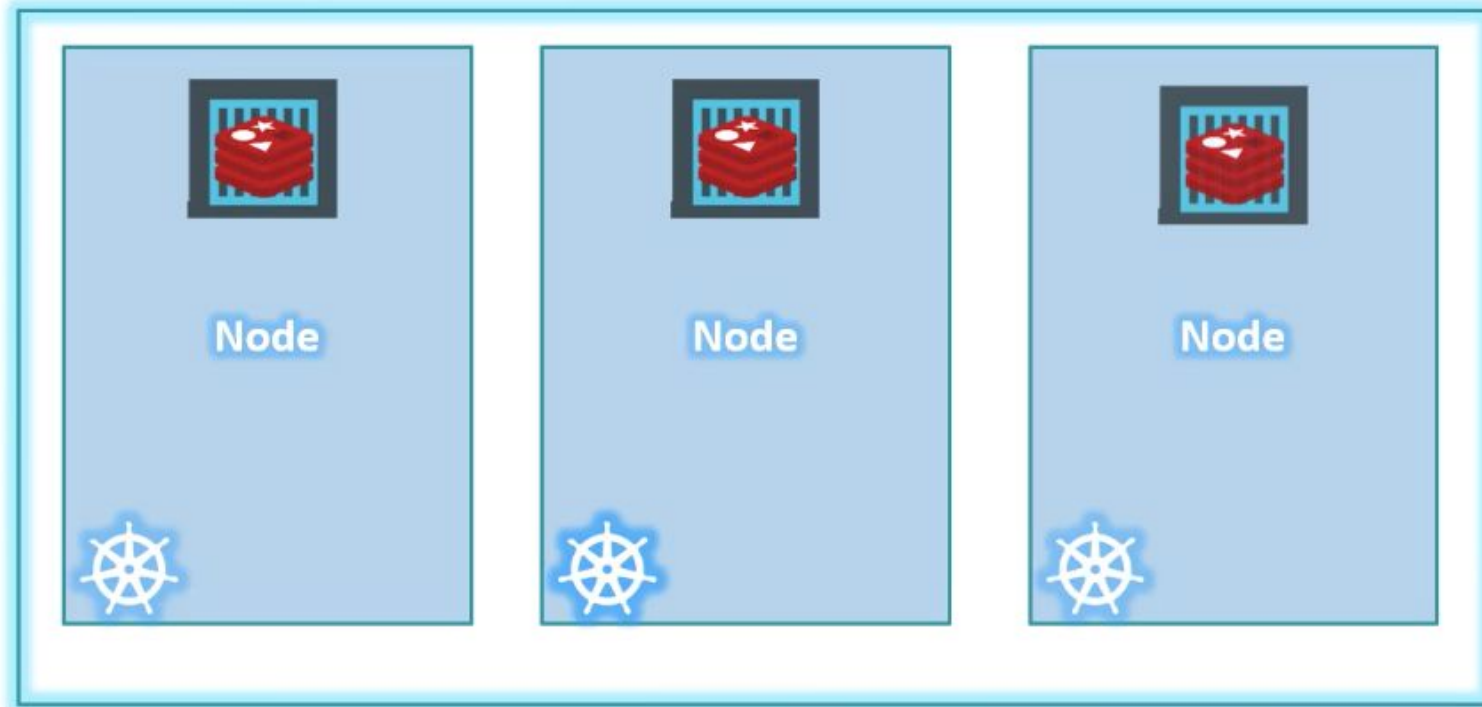
Nodes

- A Physical or Virtual Machine on which K8s is installed
- A node is a worker machine and where containers will be launched by K8s
- Also called a “minion” in the past
- Not advisable to have only one node



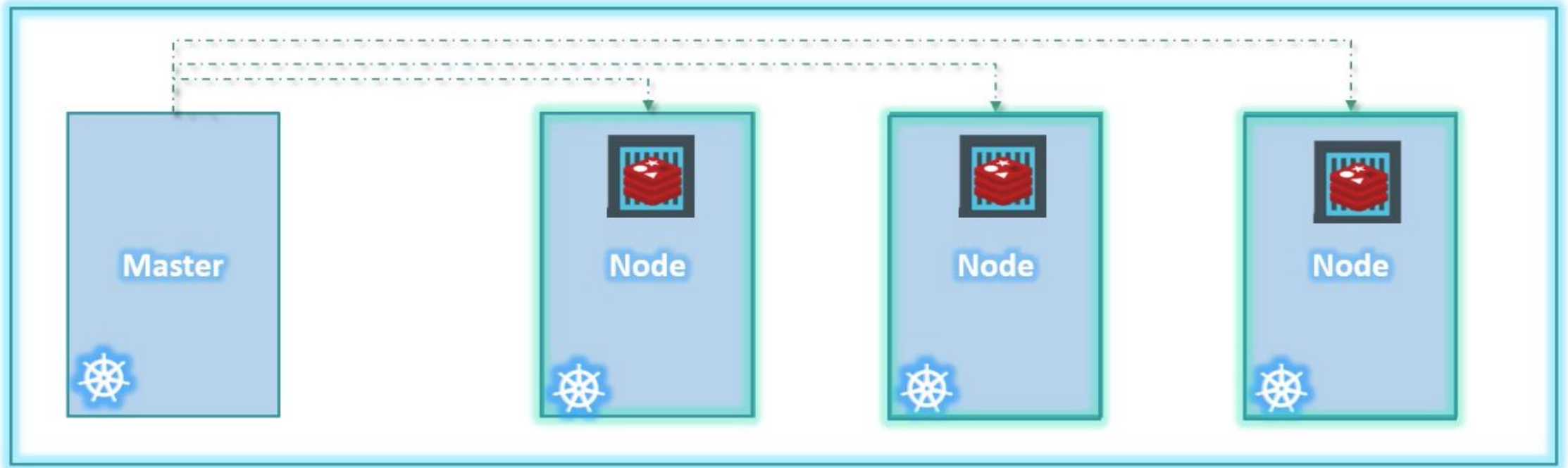


Cluster

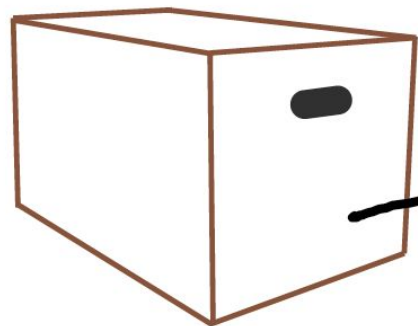


- Set of nodes grouped together
- Gives failover and ability to share the load

Master



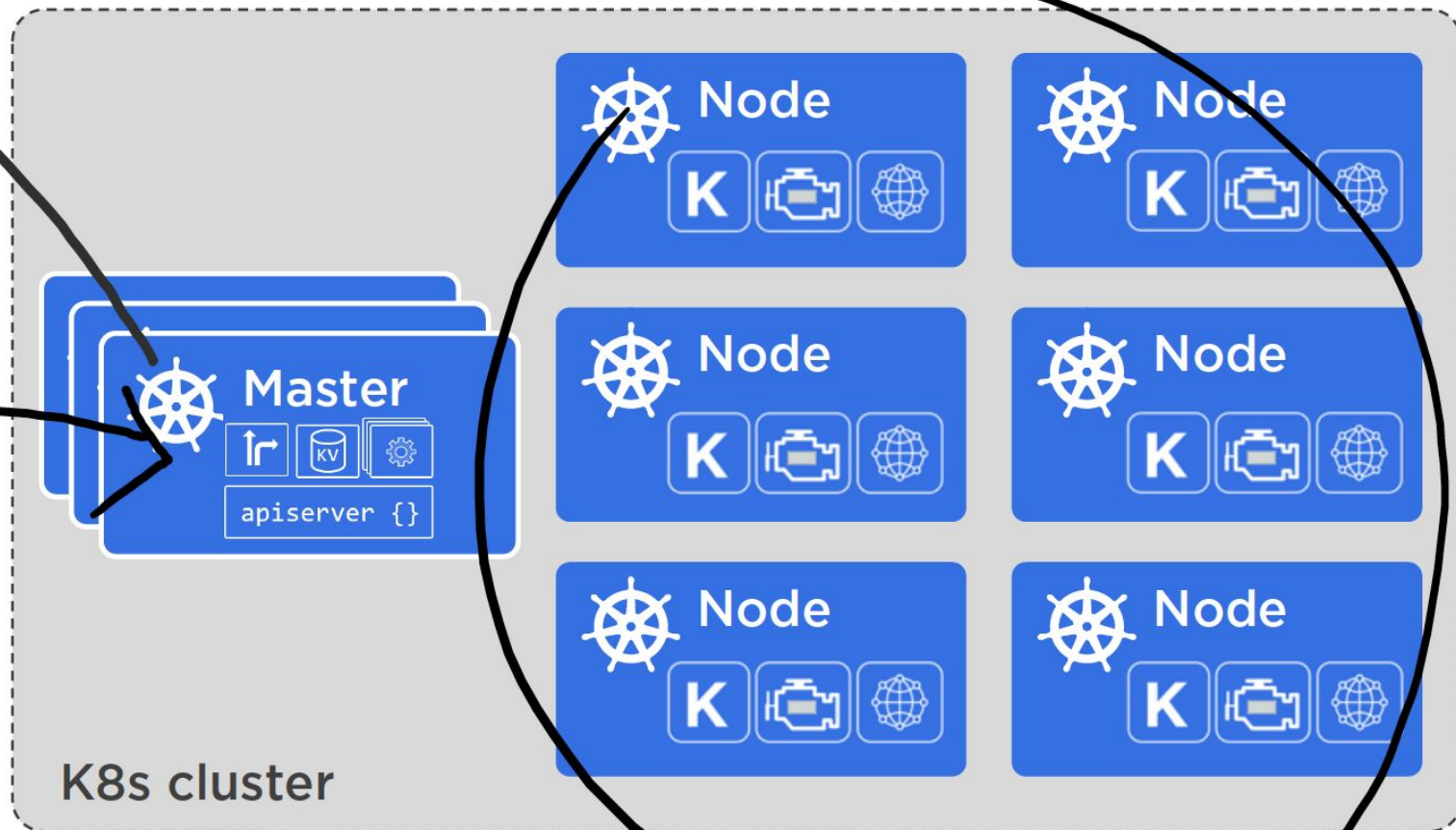
- Another node with K8s installed in it and configured to be a master node
- Master watches over the nodes in the cluster and is responsible for the actual orchestration of containers on the worker nodes
- Master Node:
 - Manages the cluster
 - Storage of the member cluster information
 - How are the nodes monitored?
 - When the node fails, how do you move the workload of the failed node to another worker node?



Deployment

in charge!

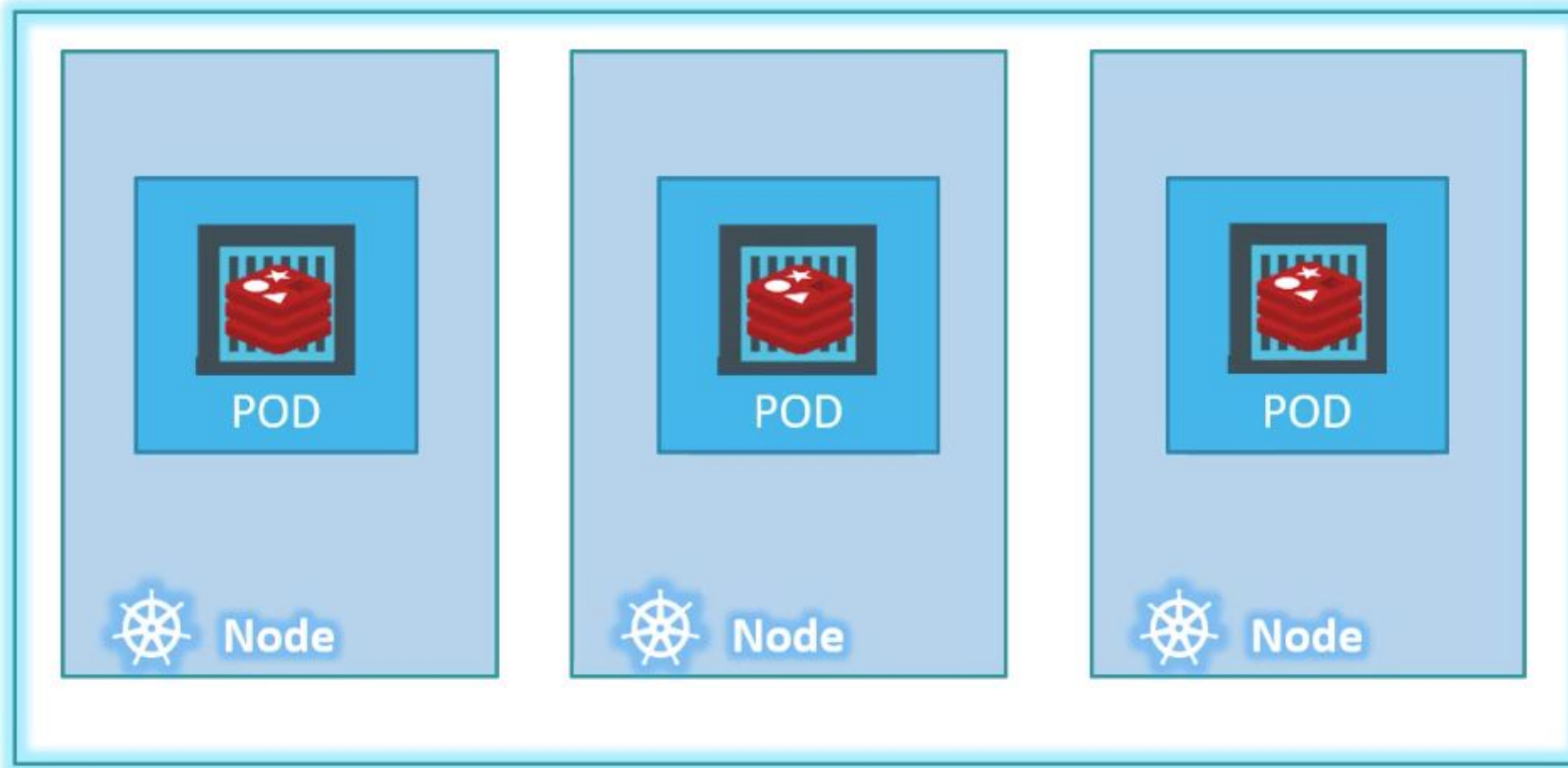
do the work!



Nodes a.k.a. Minions

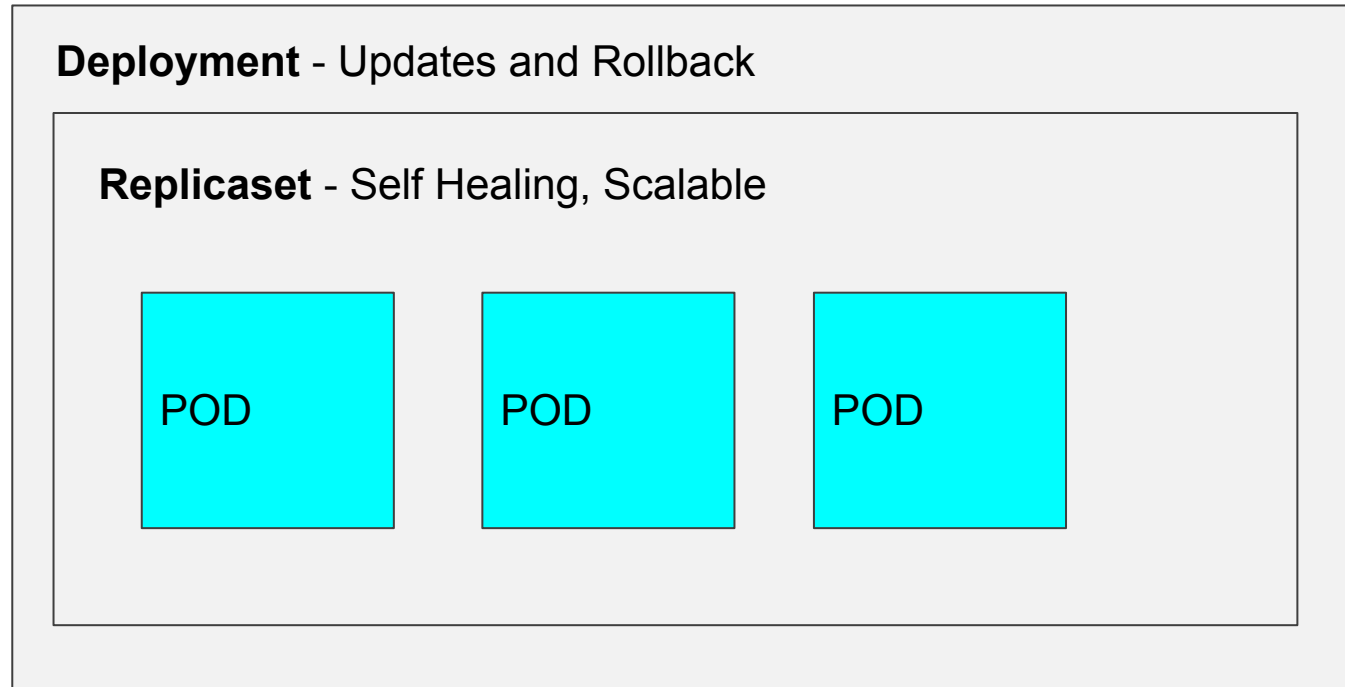


POD



- K8s does not run containers directly; instead it wraps one or more containers into a higher level structure called a “POD”.
- PODs almost always have a 1:1 relationship with containers running an application
- A POD is the smallest object you can create in Kubernetes
- Any containers in the same POD will share the same resources and local network
- Each POD has a unique IP Address

Replicaset, Deployments, & Services



Service - Enables network access to a set of pods

- ClusterIP
- NodePort
- LoadBalancer

Configuring Kubernetes Using YAML Files ...

- “Yet Another Markup Language”
- Easy way to configure and manage K8 cluster(s)
- Using YAML for K8s definitions gives you a number of advantages, including:
 - Convenience:** You’ll no longer have to add all of your parameters to the command line
 - Maintenance:** YAML files can be added to source control, so you can track changes
 - Flexibility:** You’ll be able to create much more complex structures using YAML than you can on the command line
- YAML is a superset of JSON, which means that any valid JSON file is also a valid YAML file

```
---
apiVersion: v1
kind: Pod
metadata:
  name: rss-site
  labels:
    app: web
spec:
  containers:
    - name: front-end
      image: nginx
      ports:
        - containerPort: 80
    - name: rss-reader
      image: nickchase/rss-php-nginx:v1
      ports:
        - containerPort: 88
```


Components

- * Front end for Kubernetes
- * Users, command line interfaces, management devices all talk to the API Server to interact with the K8 Cluster

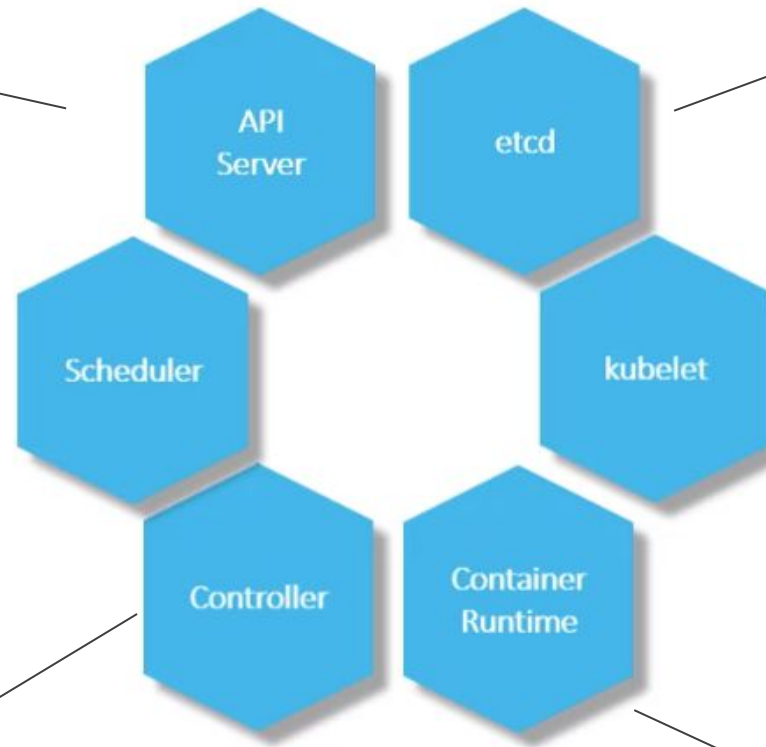
- * Looks for newly created Pods that have no Node assigned
- * For every Pod that the scheduler discovers, the scheduler becomes responsible for finding the best Node for the Pod to run on

- * The brains behind orchestration
- * Watches the state of the cluster, then makes or request changes where needed

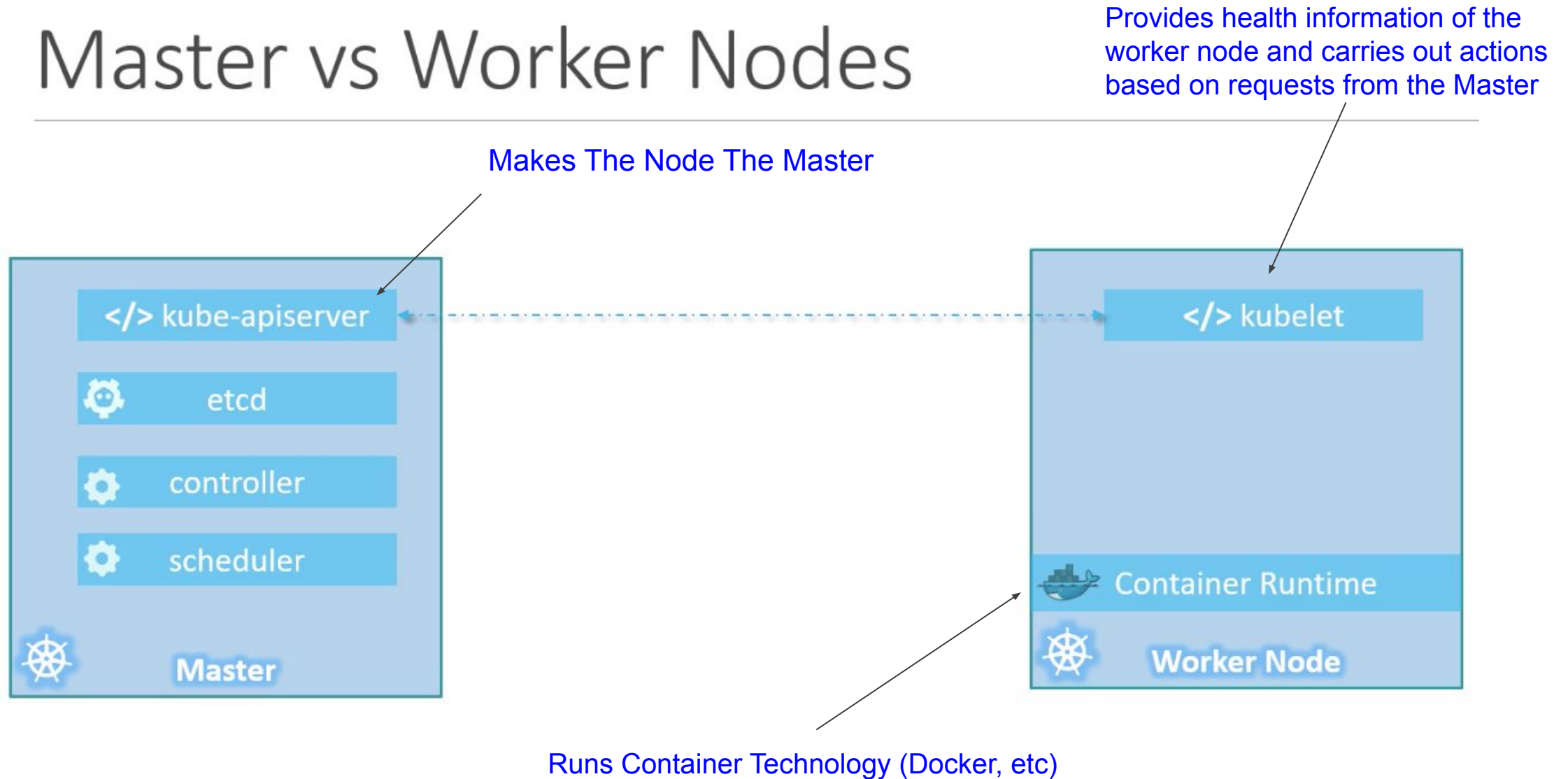
- * Distributed key / value store used by K8 to store all data about the cluster
- * K8s uses etcd to store all its data – its configuration data, its state, and its metadata

- * Primary “node agent” that runs on each node of the cluster
- * Agent makes sure that the containers running on the node are running as expected

- * Underlying software that is used to run the containers
- * In most cases, its Docker but there are other options



Master vs Worker Nodes



kubectl

- command line interface for running commands against Kubernetes clusters

```
kubectl run hello-minikube
```



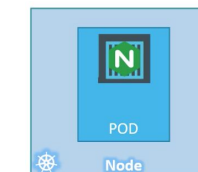
```
kubectl cluster-info
```



```
kubectl get nodes
```



```
kubectl run nginx --image nginx
```





Demo Time

Thank You

<https://github.com/fjb4/kubernetes-demo>