# Kubernetes

- Born in Google
- Donated to CNCF in 2014
- Open source (Apache 2.0)
- v1.0 July 2015
- Written in Go/Golang
- https://github.com/kubernetes/kubernetes
- IRC, @kubernetesio, slack.k8s.io, Meetups...
- DNA from Borg and Omega
- Often shortened to **k8s**

Kubernetes

K8s

Greek for "Helmsman" < the person who steers a ship

Containers bring scalability challenges!

We're starting to view the data center as a computer!

We're starting to view the data center as a computer!

We're starting to view the data center as a computer!

**Kubernetes** can manage it

# Pets vs. Cattle

- Standard package format
- Manifest

Job done!

- Standard package format
- Manifest

Job done!

It's early days (but not super early)

Kubernetes is strongly positioned

Very platform agnostic

Lets you target deployments

Stick with it!

- Standard package format
- Manifest

Job done!

# Kubernetes Architecture

Team

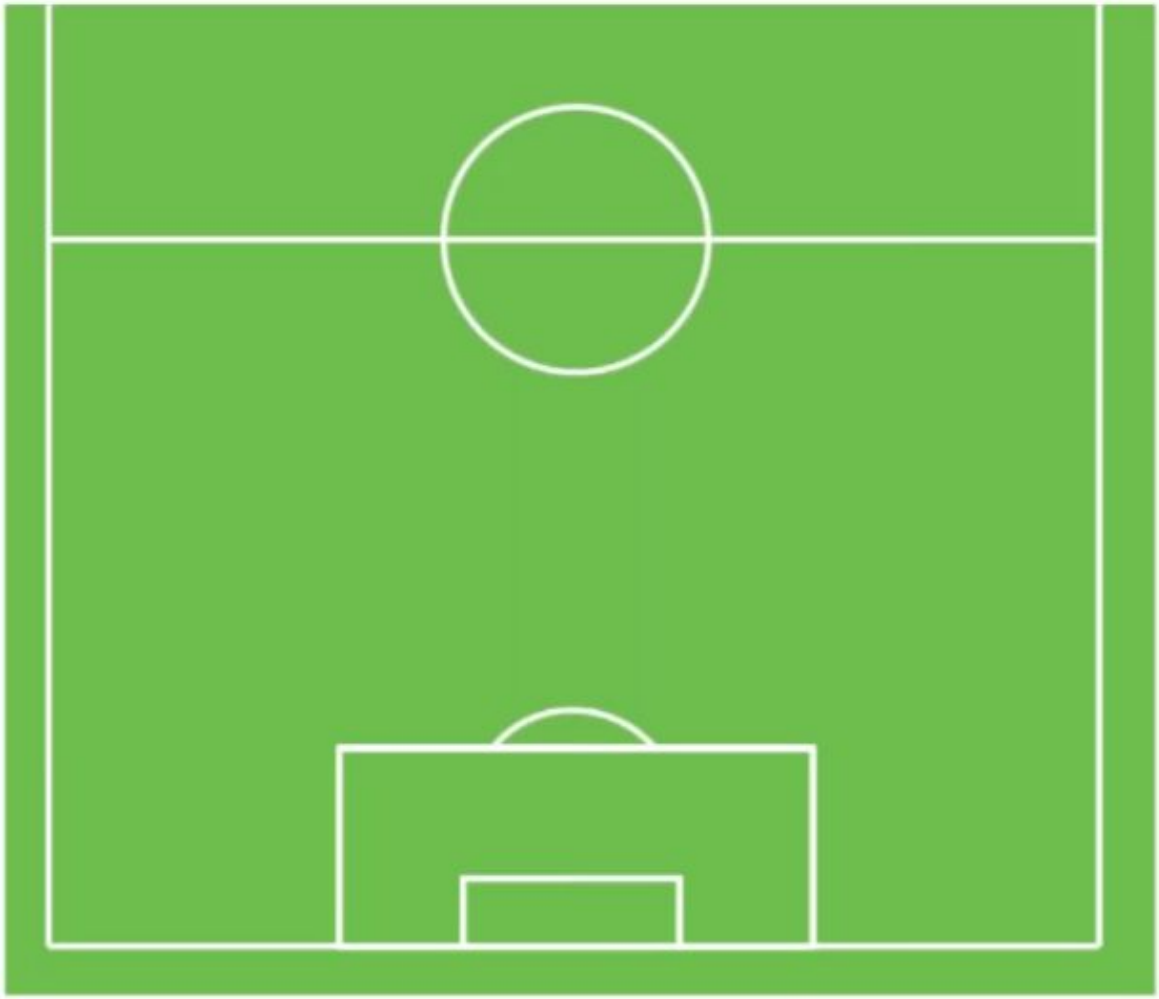
Manager
(coach)

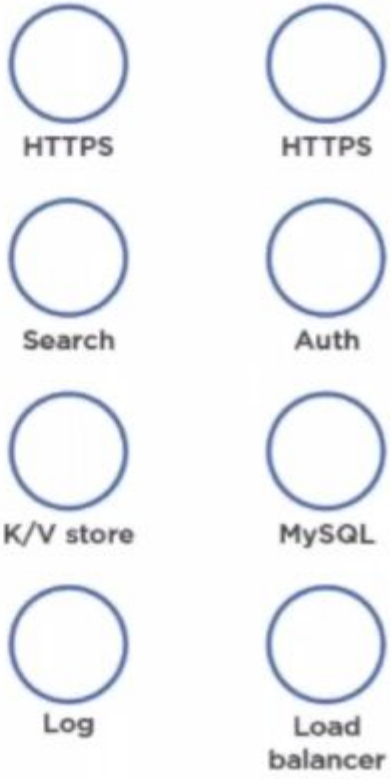
S1

S2 S3

RW S CH HM LW RB CH AM LB S GK

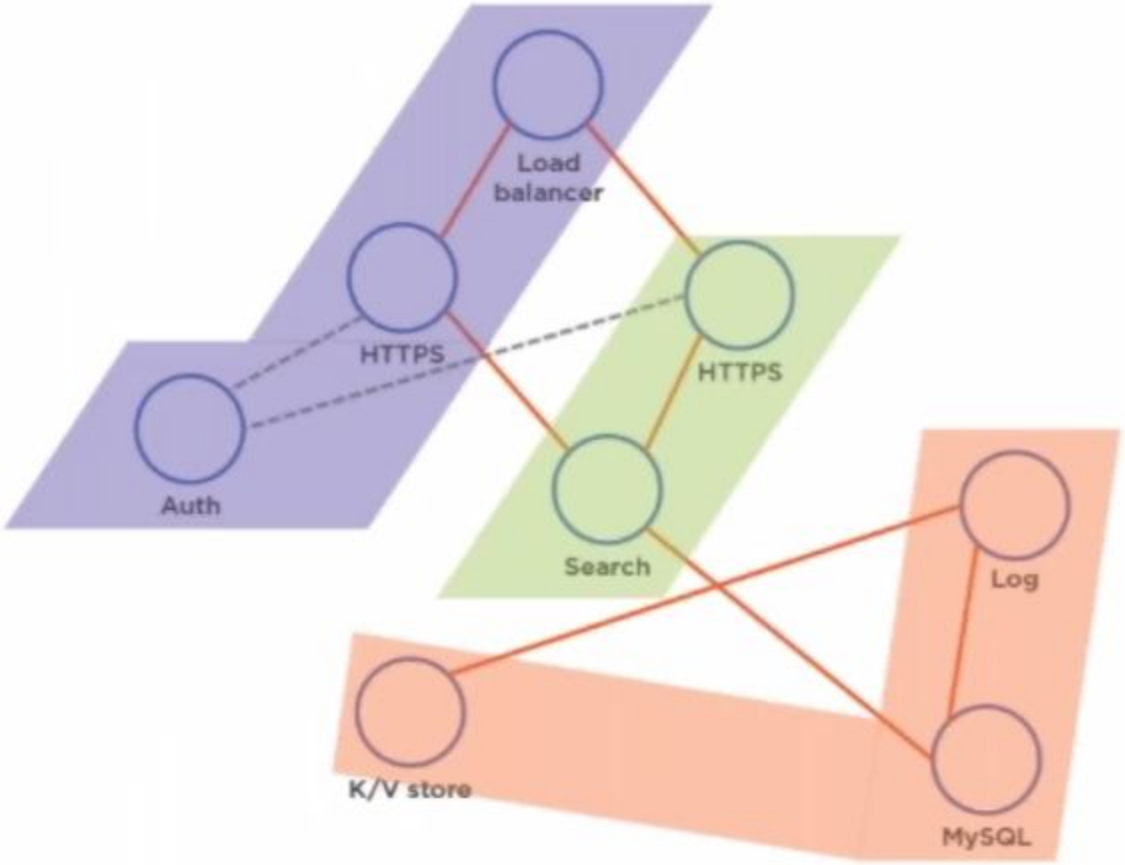
Team

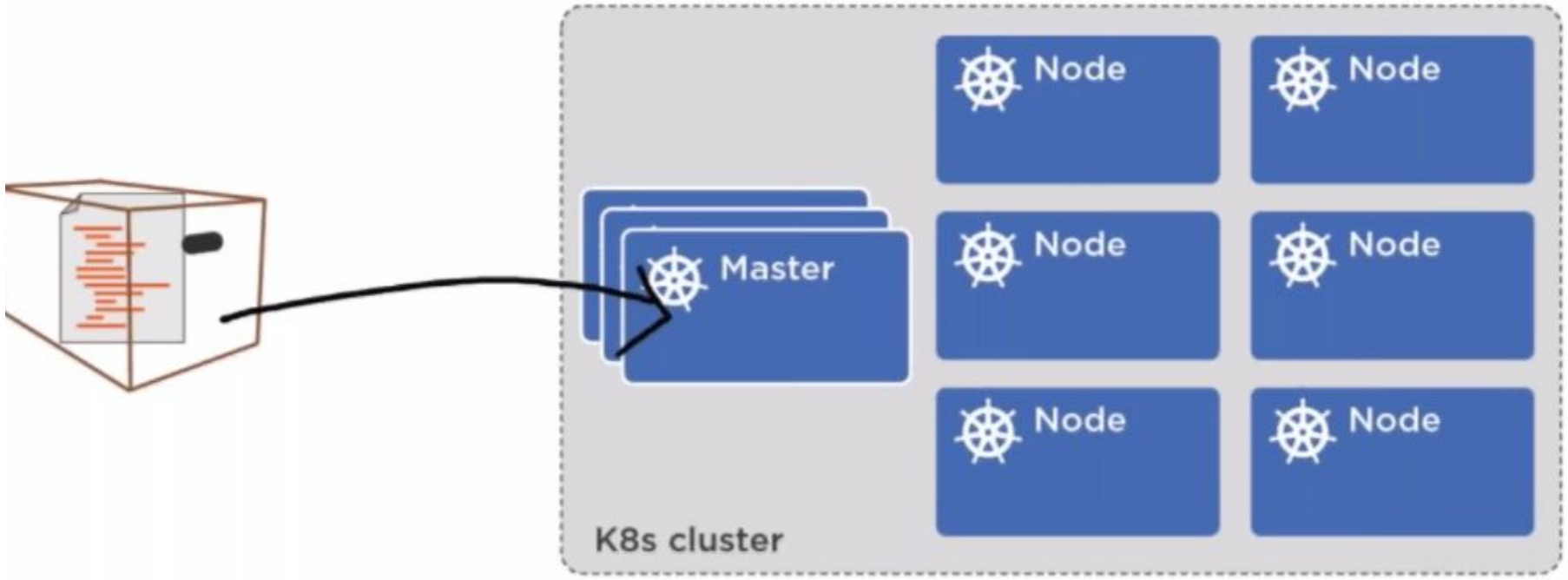


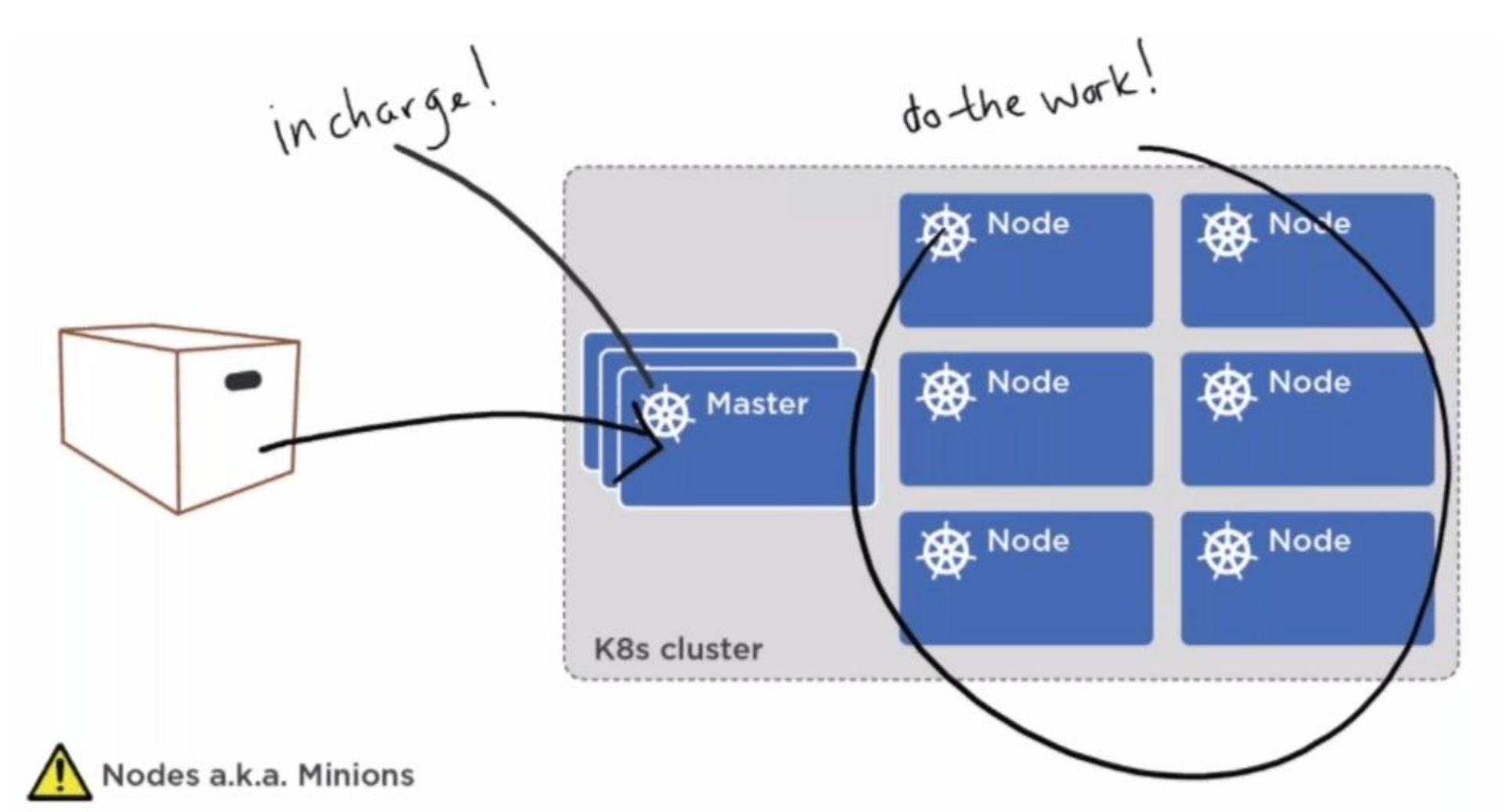

HTTPS
HTTPS
Search
Auth
K/V store
MySQL
Log
Load balancer

# Kubernetes
# Big Picture View

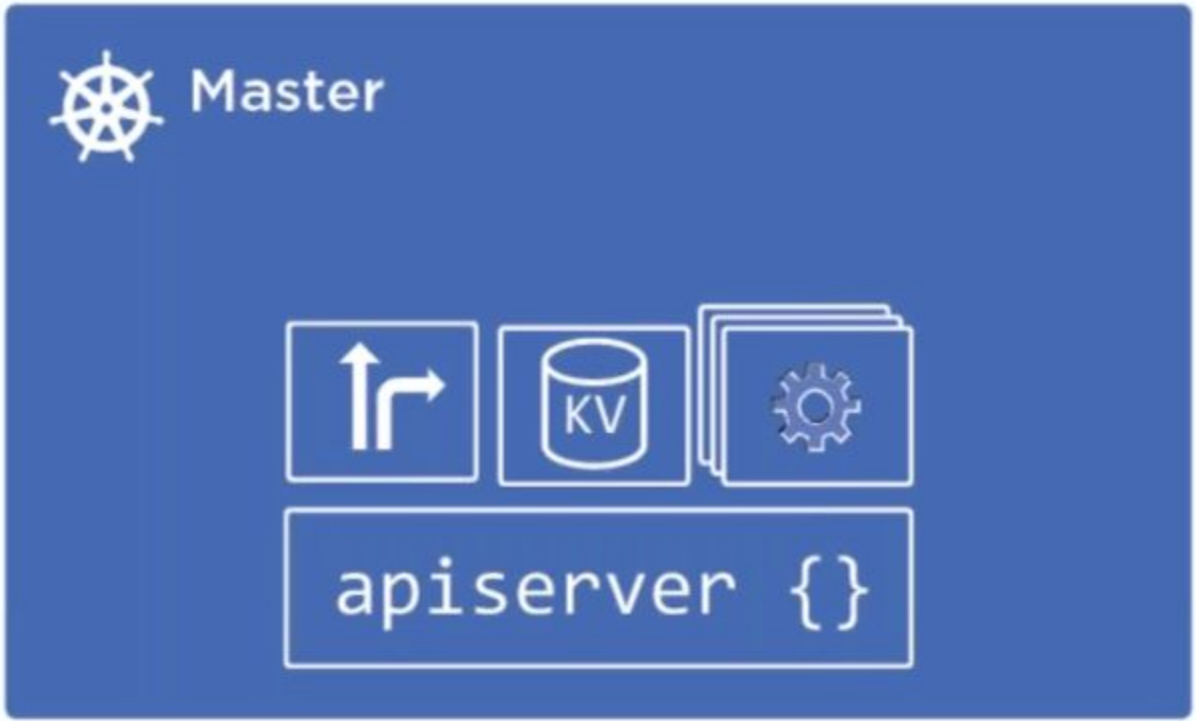

Node 1
Node 2
Node 3

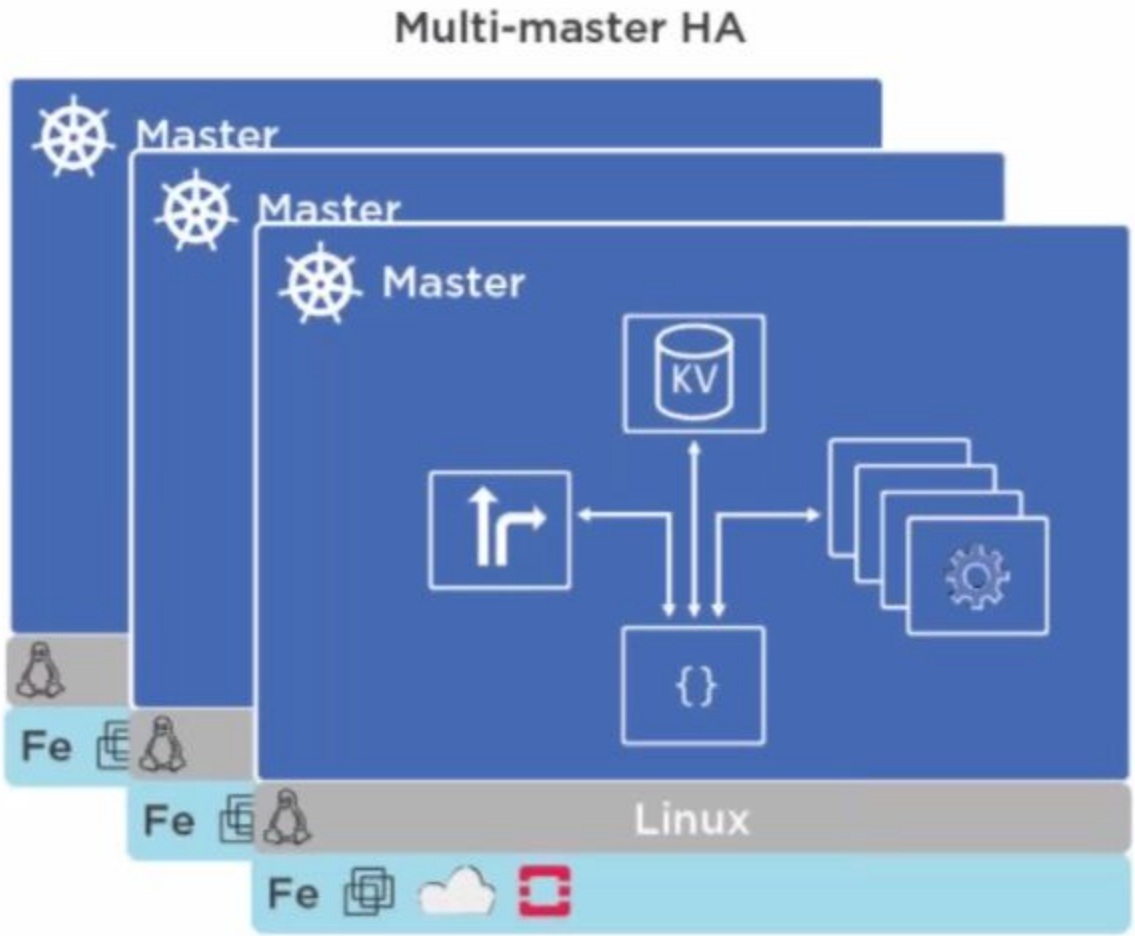

Load
balancer
HTTPS
HTTPS
Auth
Search
Log
K/V store
MySQL


Node
Node
Node
Node
Node
Node
Master
K8s cluster




Nodes a.k.a. Minions


Master
KV
apiserver {}


Multi-master HA
Master
Master
Master
KV
{}
Fe
Fe
Fe
Linux

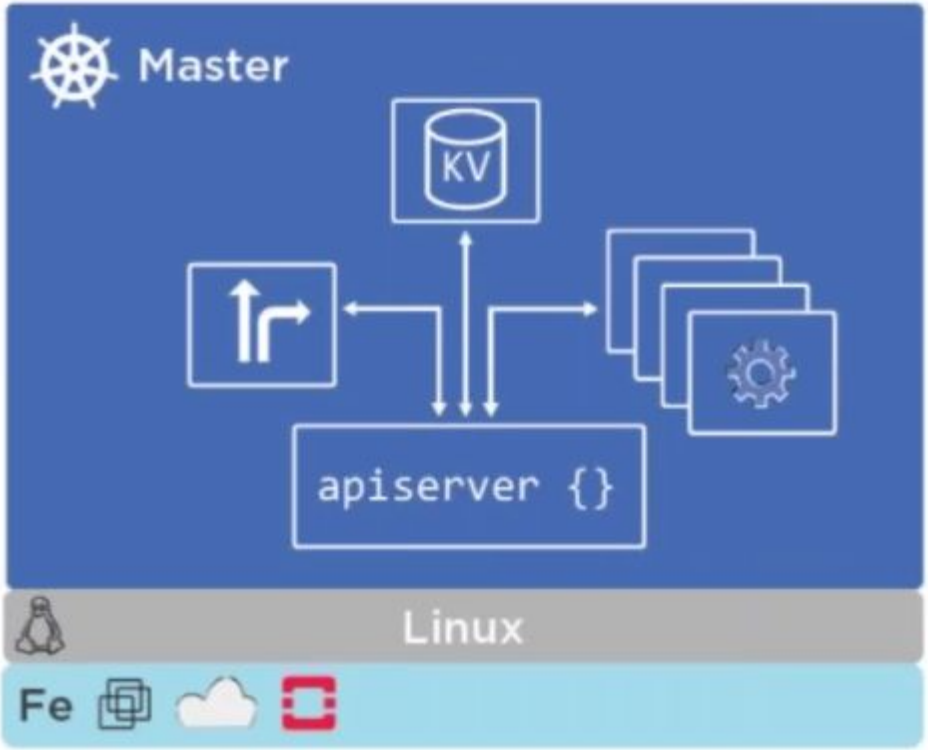
# kube-apiserver

Front-end to the control plane

Exposes the API (REST)

Consumes JSON

(via manifest files)



# Cluster store

Persistent storage

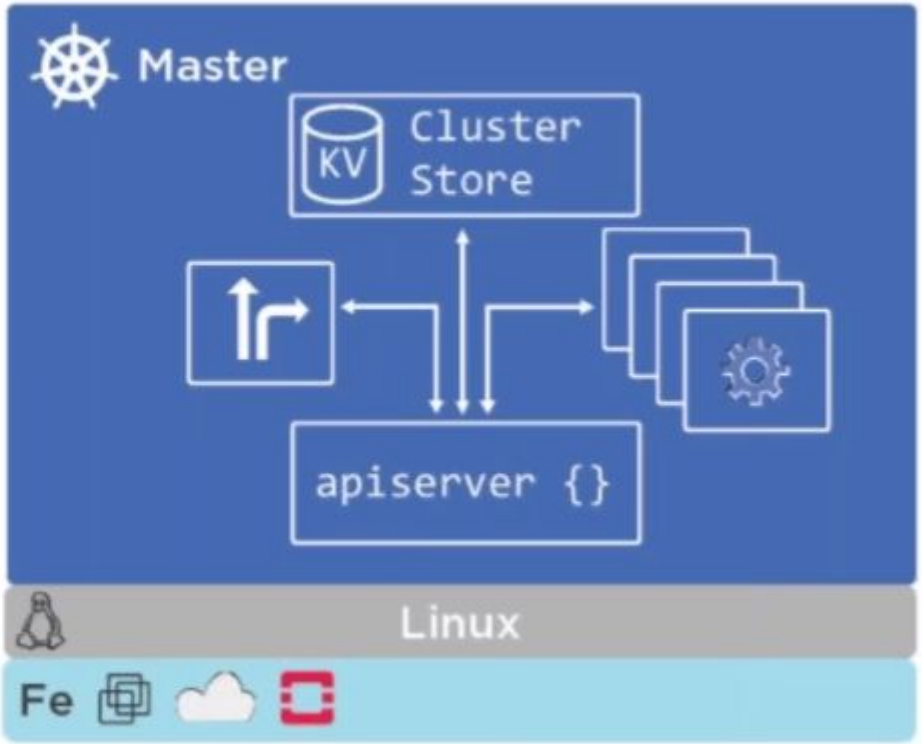
Cluster state and config

Uses **etcd**

Distributed, consistent, watchable...

The *"source of truth"* for the cluster

Have a backup plan for it!



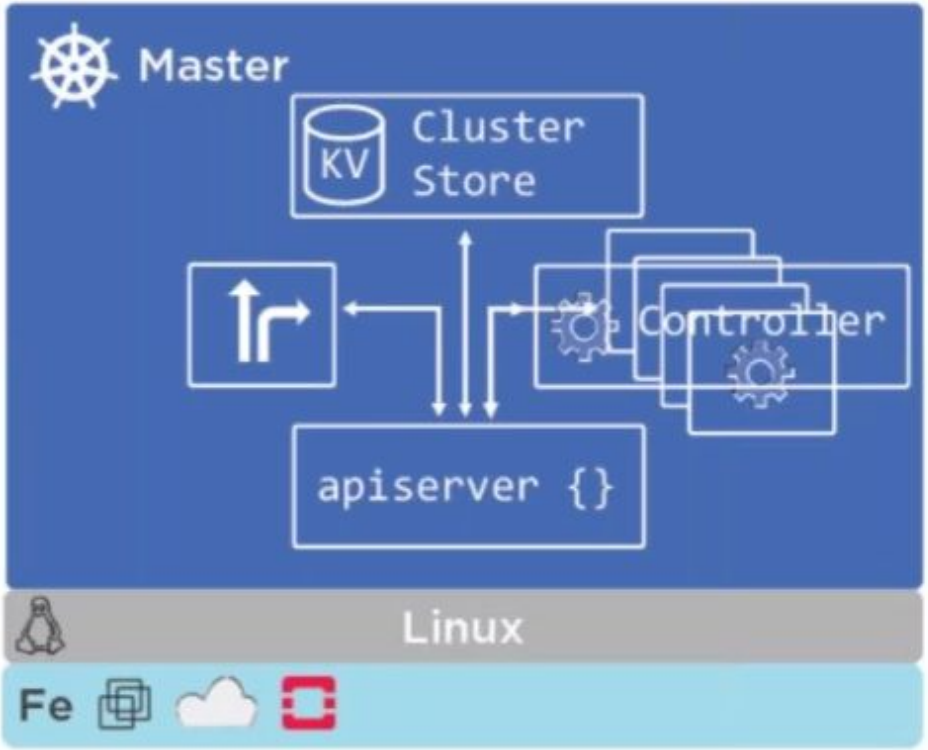
# kube-controller-manager

Controller of controllers

- Node controller
- Endpoints controller
- Namespace controller
- ...

Watches for changes

Helps maintain *desired state*

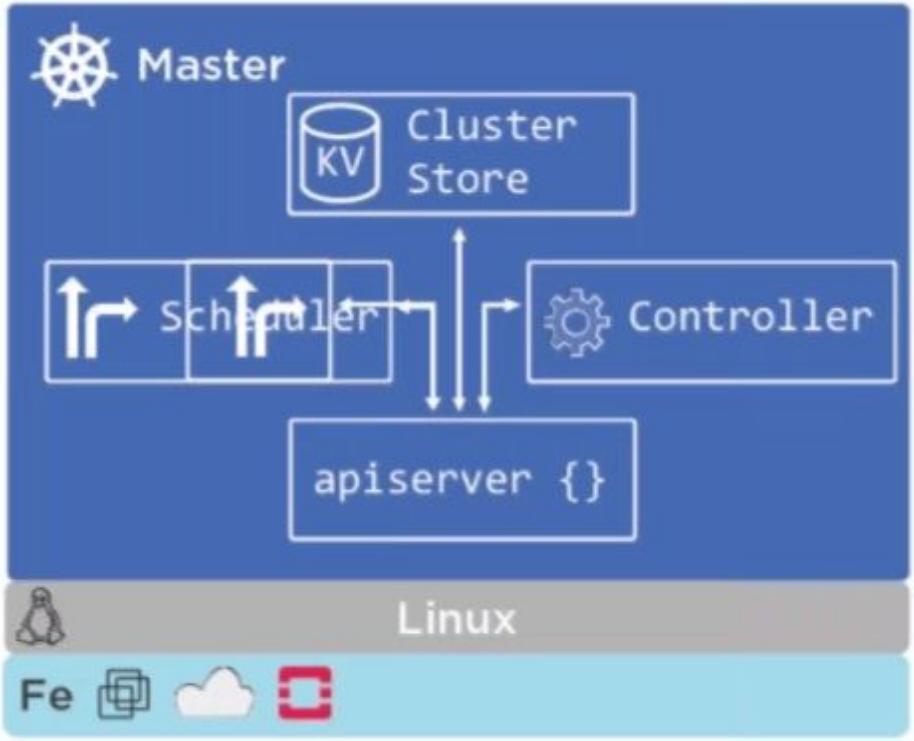


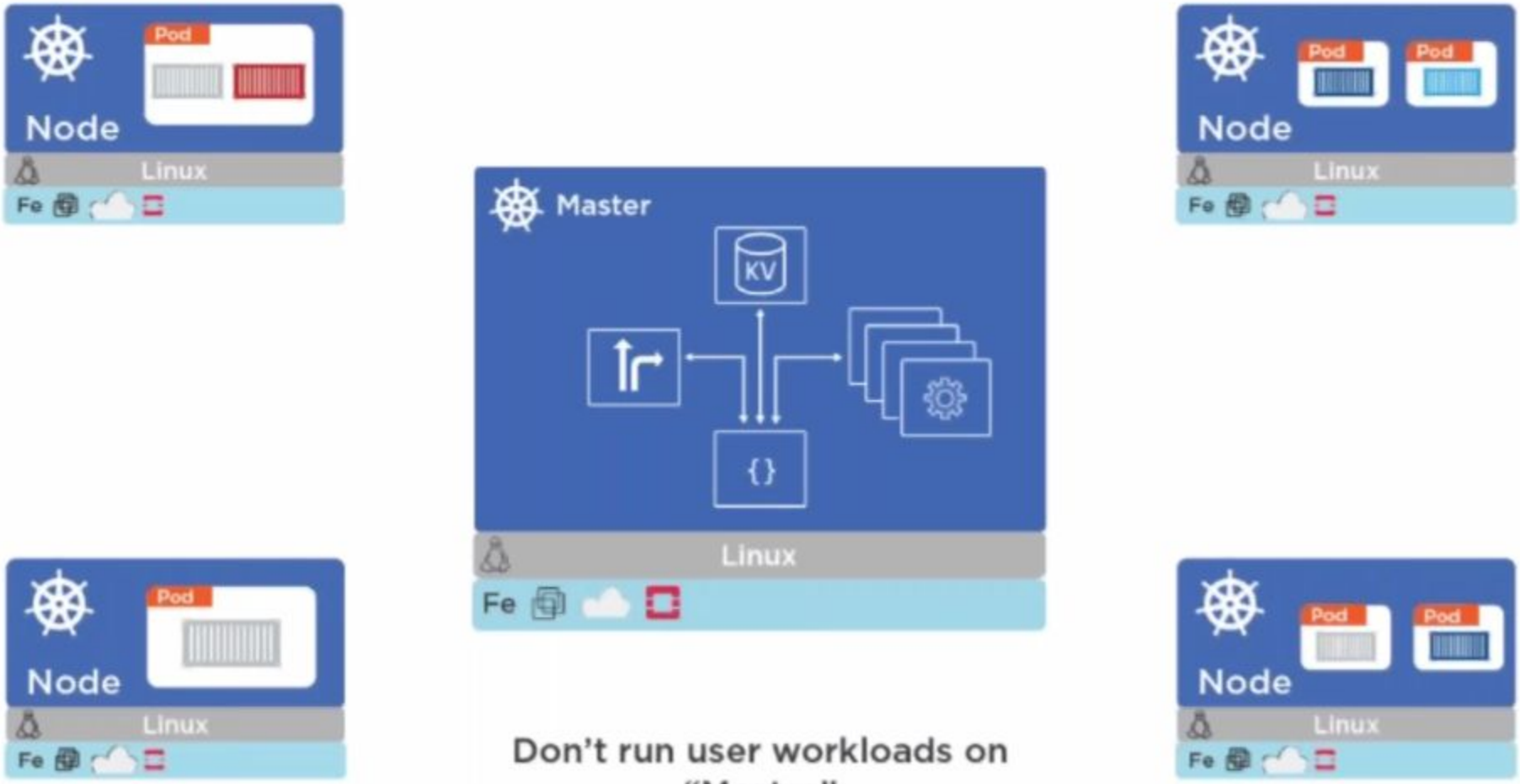
# kube-scheduler

Watches **apiserver** for new pods

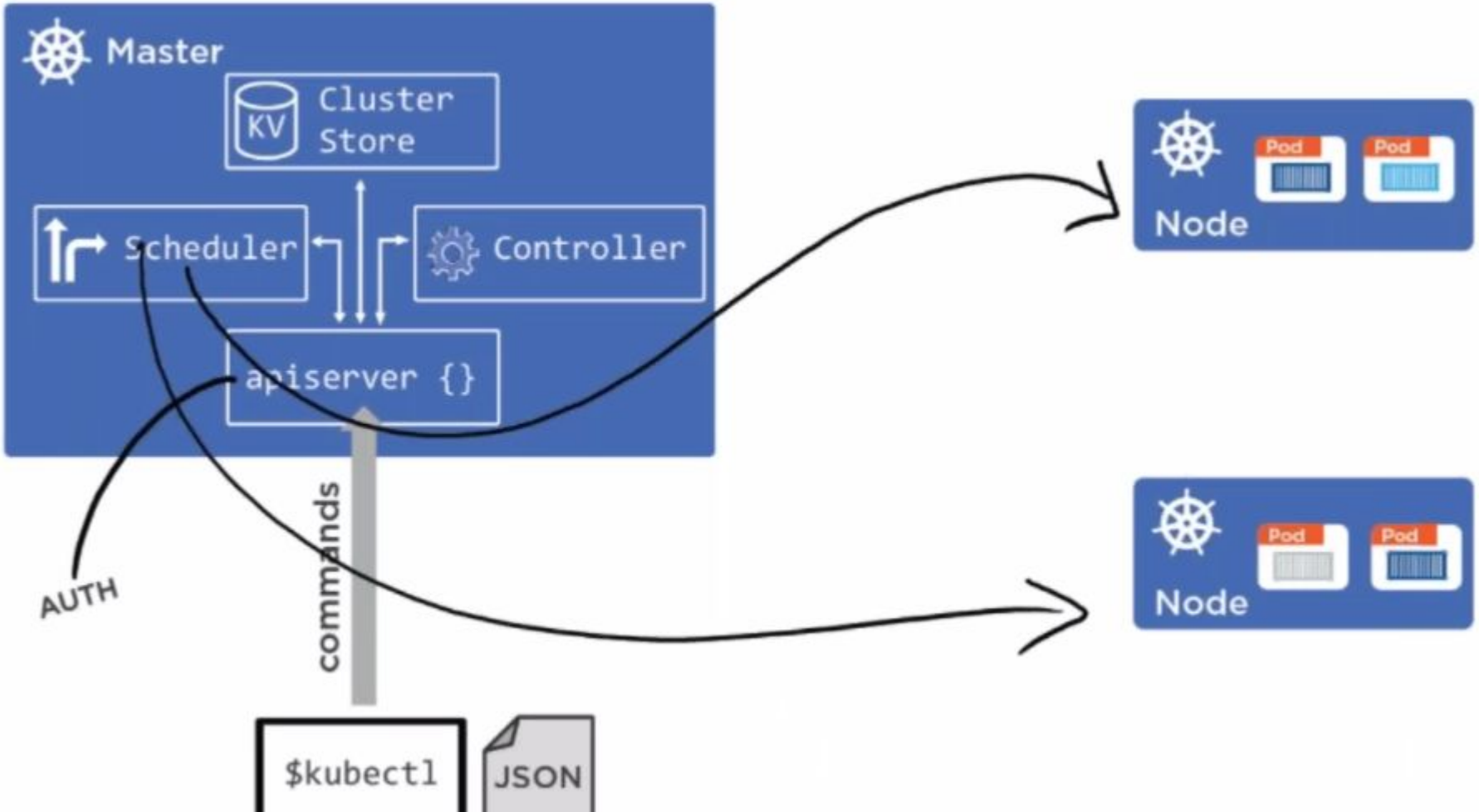
Assigns work to **nodes**

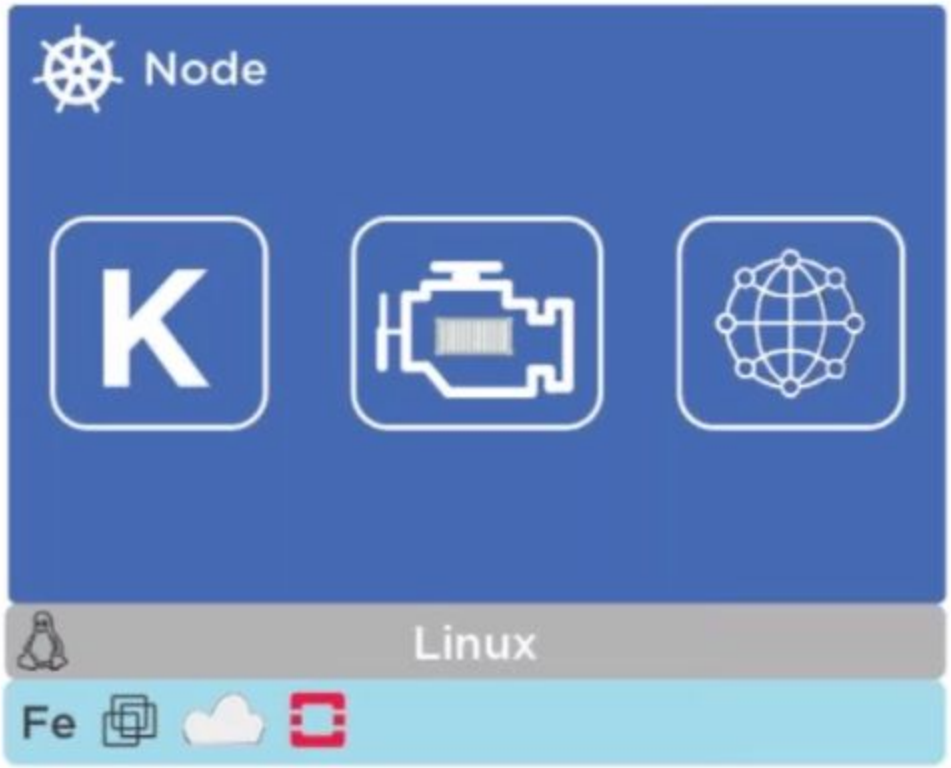
- affinity/anti-affinity
- constraints
- resources
- ...






Don't run user workloads on "Master"


Master
KV
Cluster
Store
Scheduler
Controller
apiserver {}
AUTH
commands
$kubectl
JSON
Pod
Pod
Node
Pod
Pod
Node


Node
K
Linux
Fe

## Kubelet

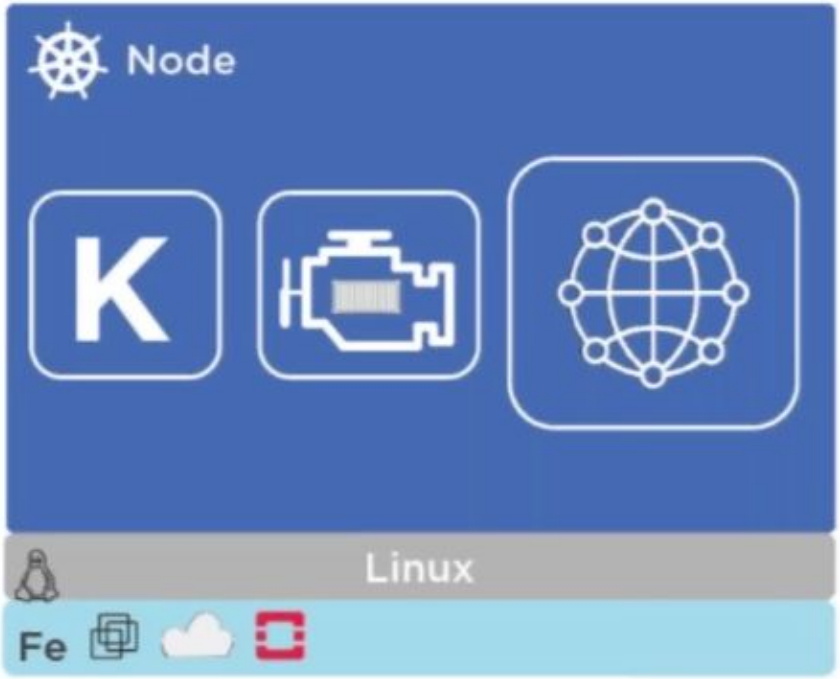
Main Kubernetes agent

## Container engine

Docker or rkt

## kube-proxy

Kubernetes networking



# Kubelet

- The **main** Kubernetes agent
- Registers node with cluster
- Watches **apiserver**
- Instantiates **pods**
- Reports back to **master**
- Exposes endpoint on :10255

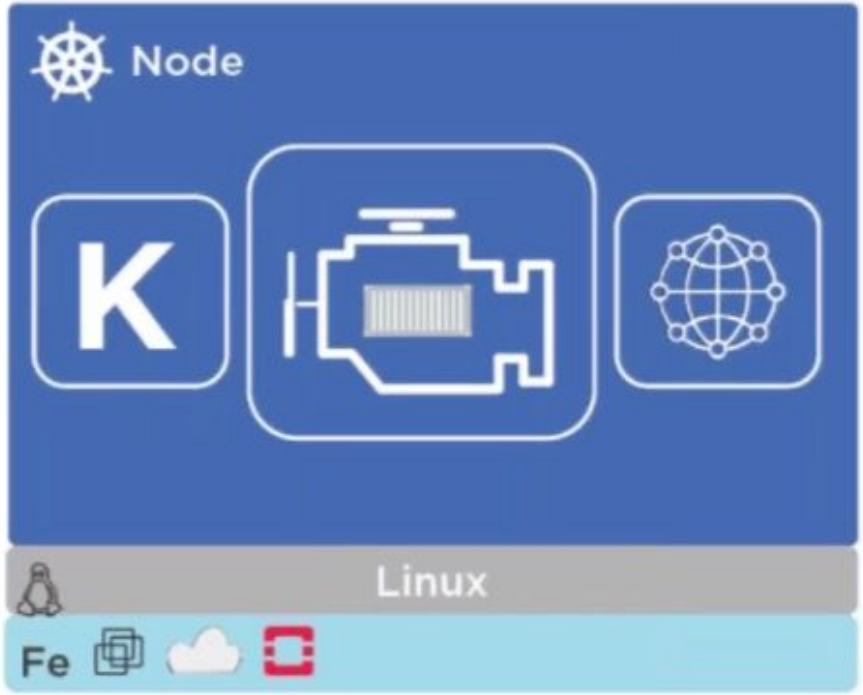
# Container Engine

Does container management:

- Pulling images
- Starting/stopping containers
- ...

Pluggable:

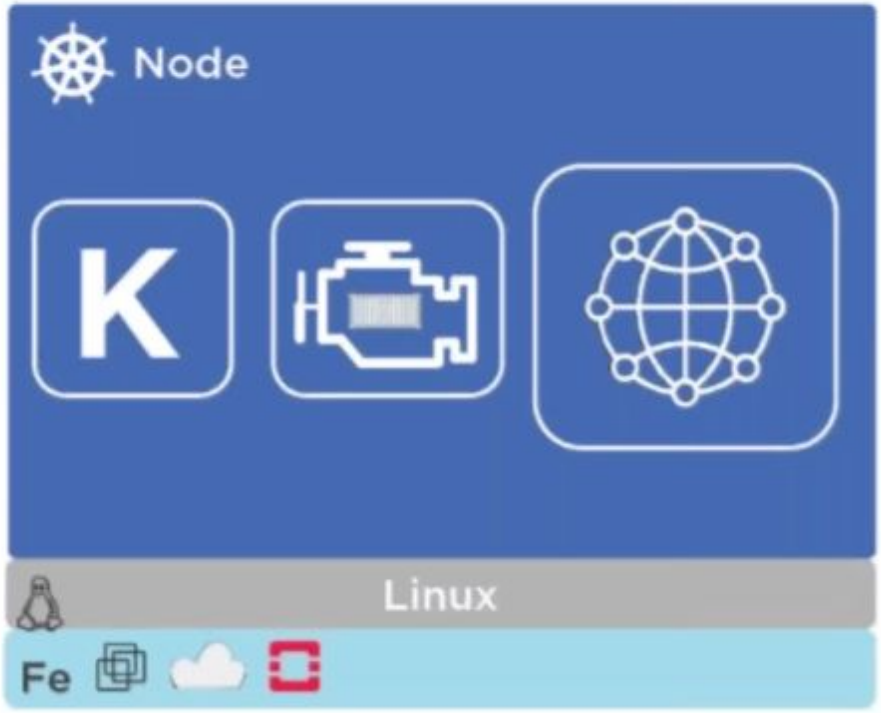
- Usually **Docker**
- Can be **rkt**



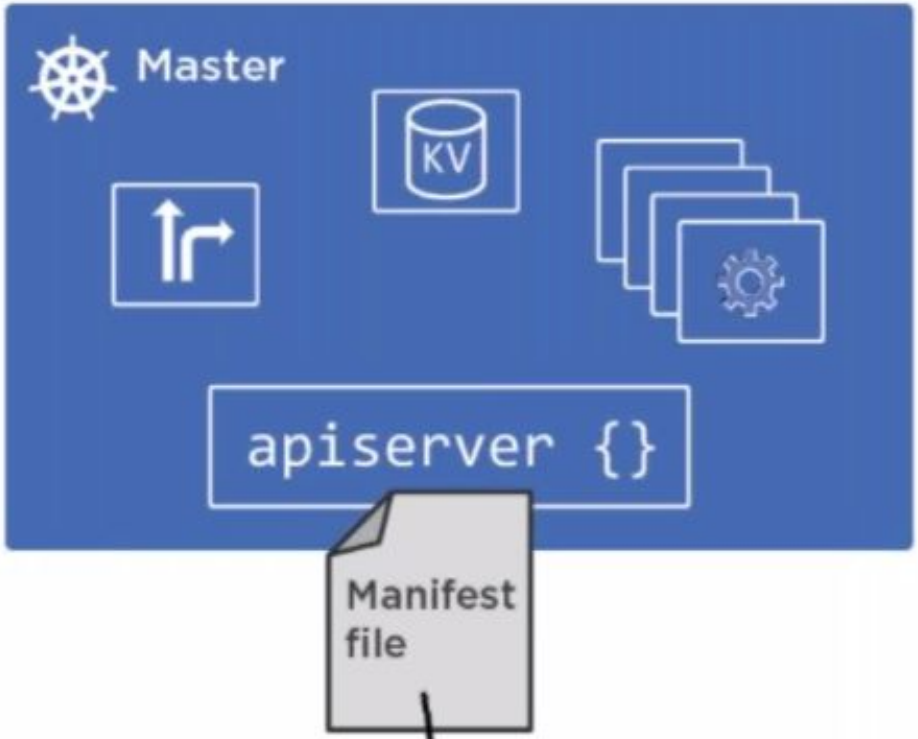
# kube-proxy

Kubernetes networking:

- Pod IP addresses
  - All containers in a pod share a single IP
- Load balances across all pods in a **service**

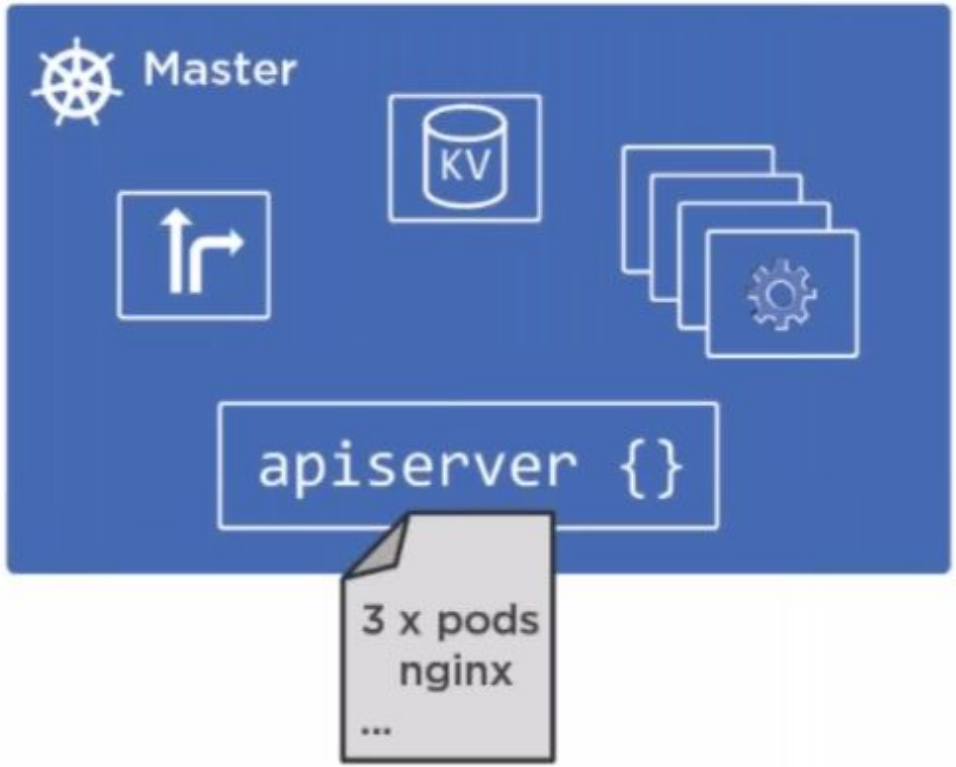


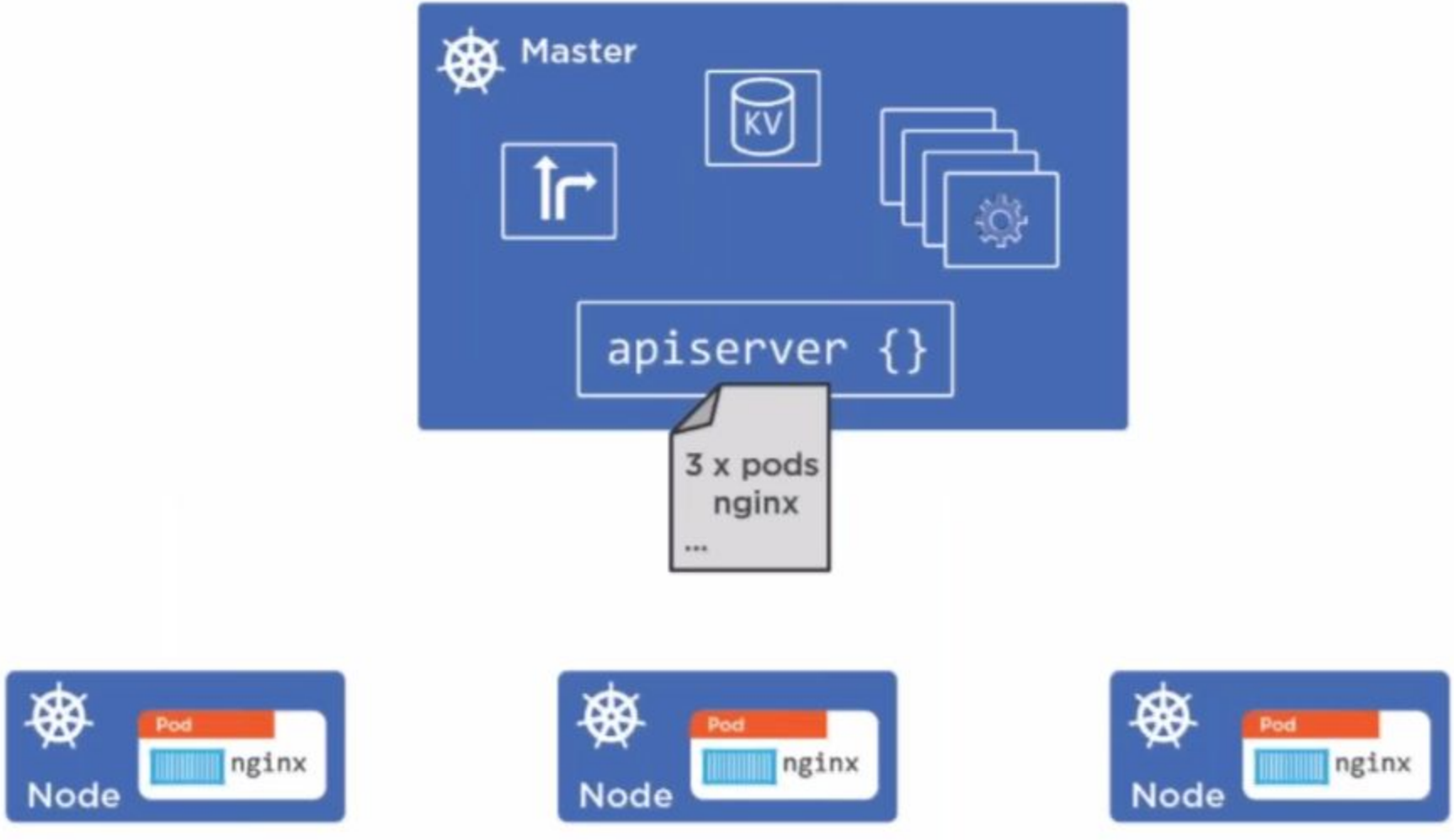
# Declarative Model
&
# Desired State

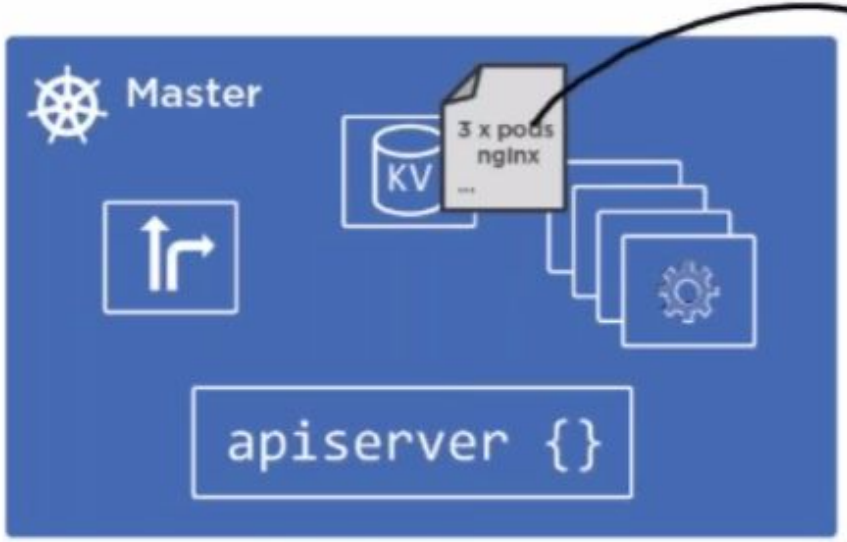



YAML or JSON

Describe **desired**


Master
KV
apiserver {}
3 x pods
nginx
...


Master
KV
apiserver {}
3 x pods
nginx
...
Node
Pod
nginx
Node
Pod
nginx
Node
Pod
nginx



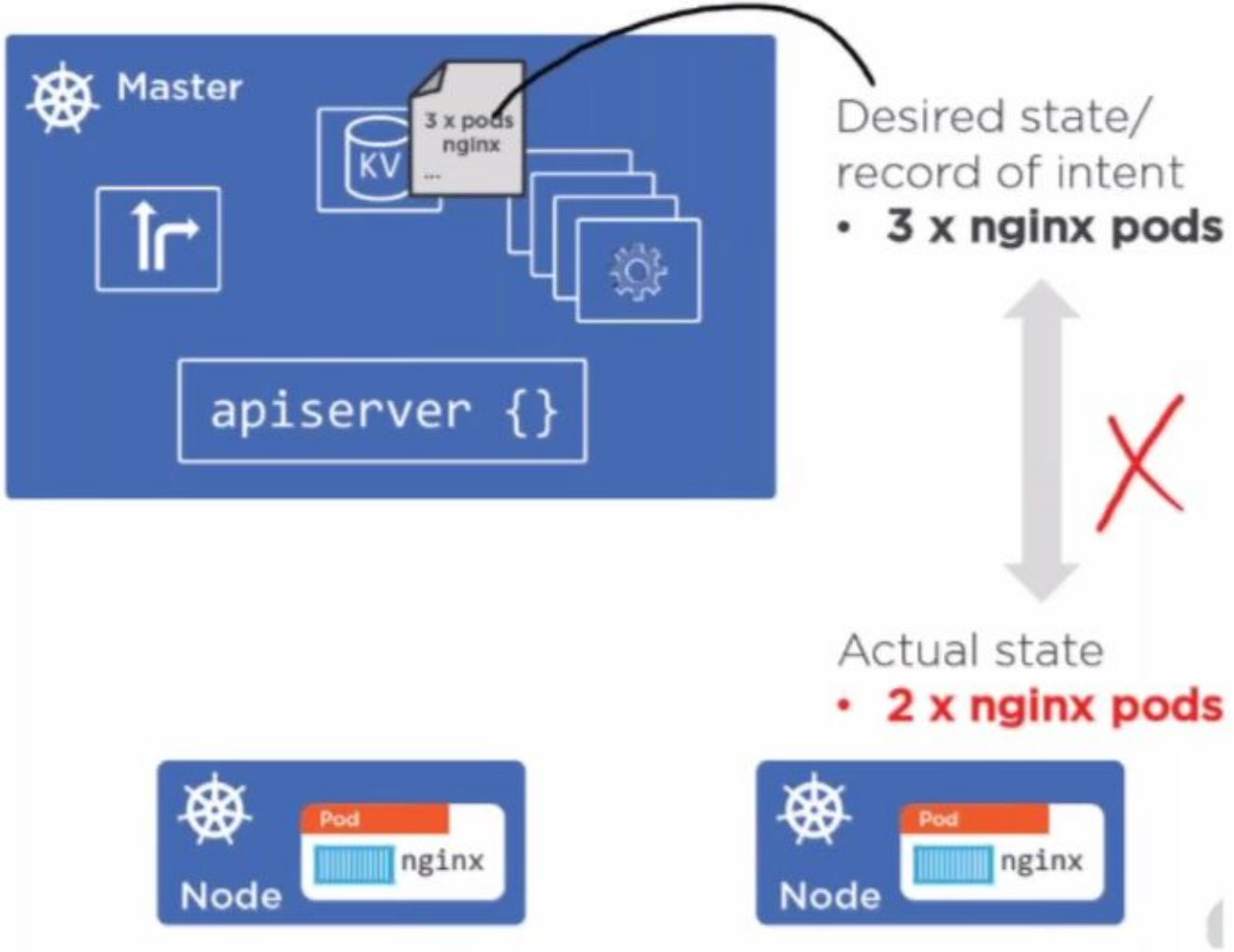

Desired state/
record of intent

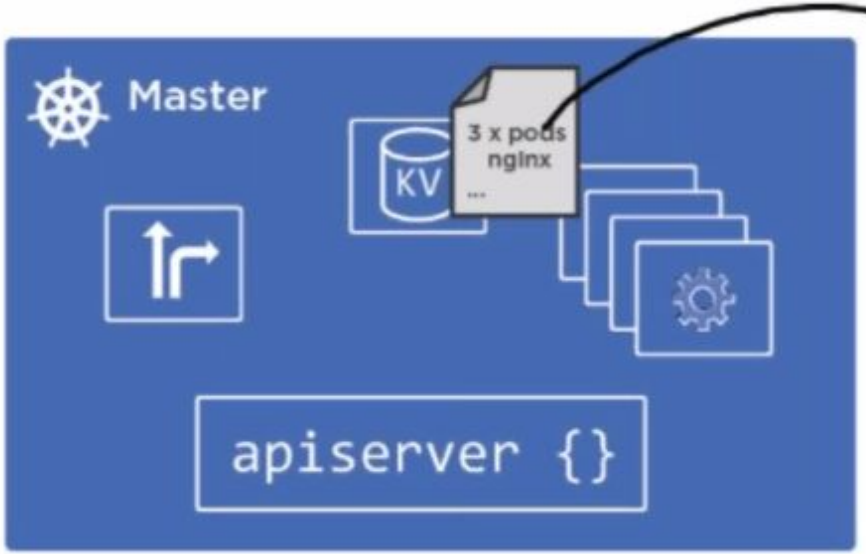
- **3 x nginx pods**

Actual state

- **3 x nginx pods**




Master
KV
3 x pods
nginx
...
apiserver {}
Desired state/
record of intent
• 3 x nginx pods
Actual state
• 2 x nginx pods
Node
Pod
nginx
Node
Pod
nginx
Node
Pod
nginx

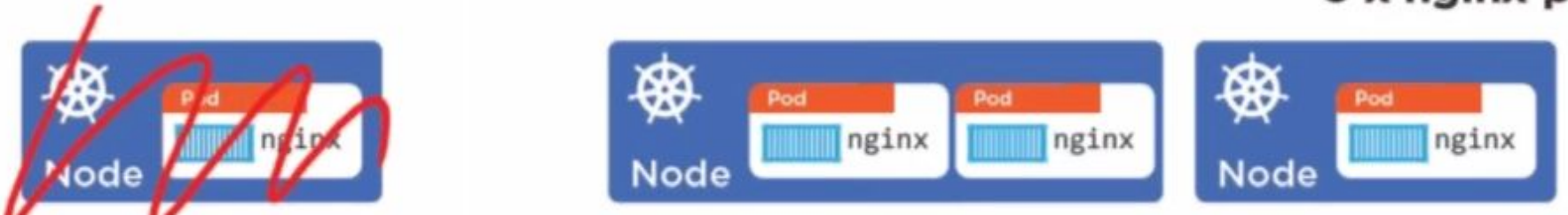



Desired state/
record of intent

- **3 x nginx pods**

Actual state

- **3 x nginx pods**




Pods


VM

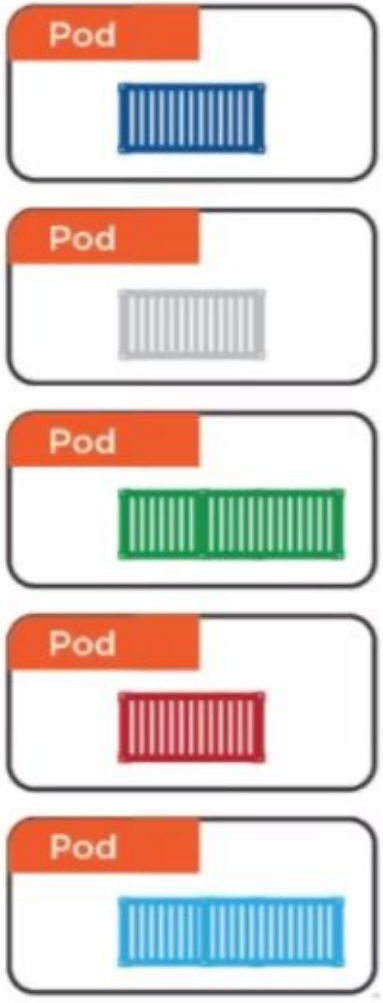
Container

Pod

Atomic units of scheduling

Containers always run inside of pods

Pods can have multiple containers
(advanced use-case)

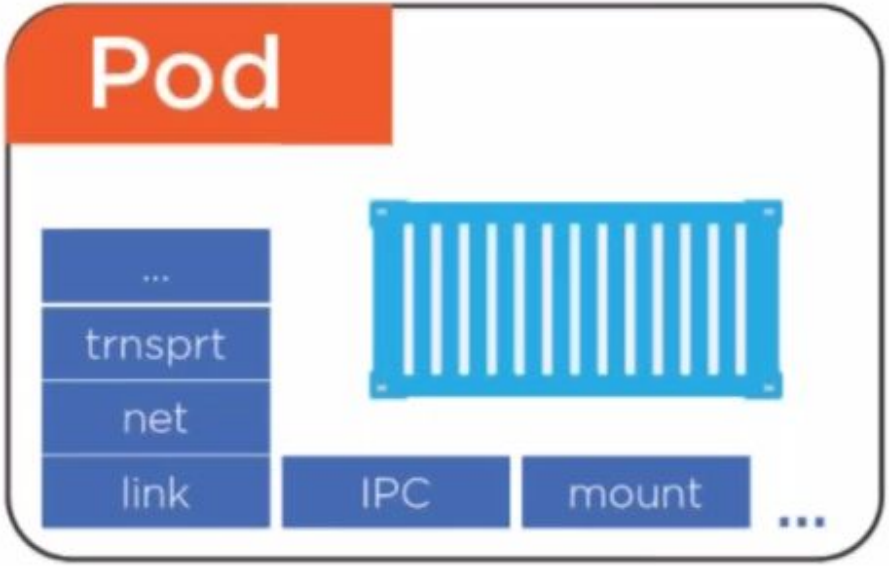


Ring-fenced environment

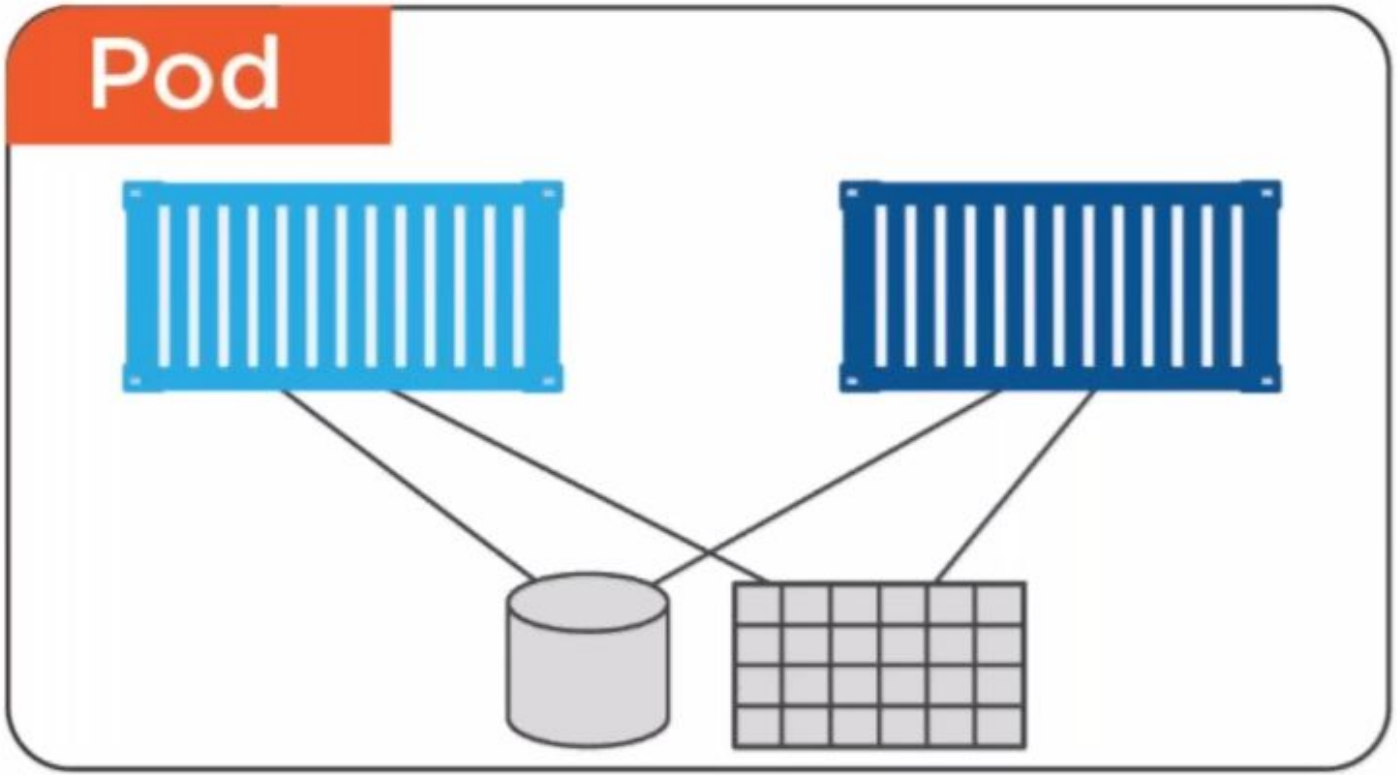
- Network stack
- Kernel namespaces
- ...

*n* containers

<u>All containers in pod share the pod environment</u>

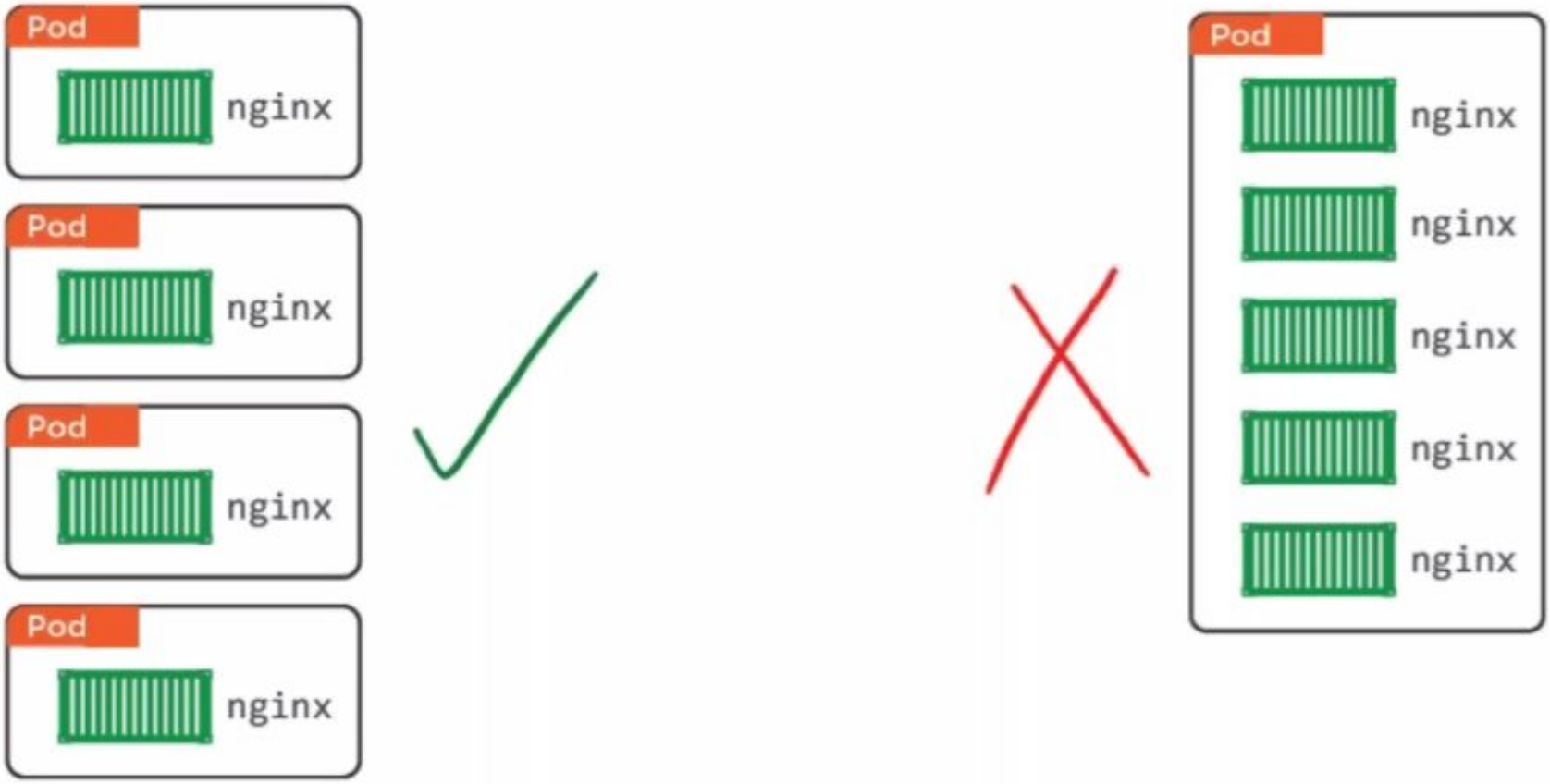


# Tight Coupling

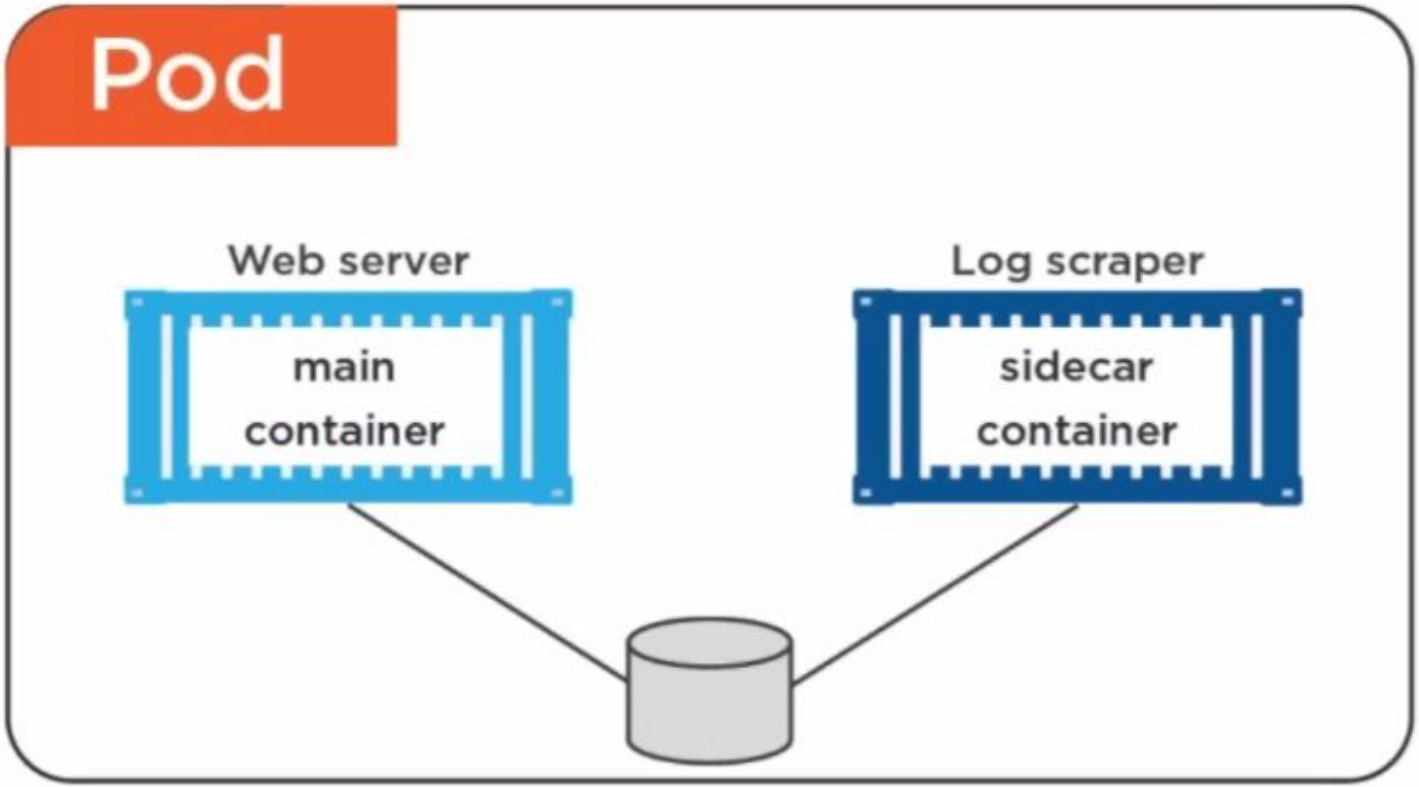


# Loose Coupling

# Pods and Scaling



# Multi-container Pods



# Pod Lifecycle


Phase: pending

Phase: running

Phase: succeeded/failed