Kubernetes Architecture and Installation

Giri Kuncoro from **GOJEK**

Giri Kuncoro

Senior Software Engineer at GOJEK

- Active Kubernetes Org Member
- sig-cli, sig-service-catalog, sig-cluster-lifecycle
- Building Pivotal Container Service 1.0
- Building Kubernetes at GOJEK
- Top 3 K8s Contributors from VMware
- Talks at Kubecon China 2018

Learning Objectives

- Discuss Kubernetes
- Learn basic Kubernetes terminology
- Learn installation and configuration tools

Borg Heritage

Large-scale cluster management at Google with Borg

Abhishek Verma† Luis Pedrosa‡ Madhukar Korupolu
David Oppenheimer Eric Tune John Wilkes

Google Inc.

Abstract

Google's Borg system is a cluster manager that runs hundreds of thousands of jobs, from many thousands of different applications, across a number of clusters each with up to tens of thousands of machines.

It achieves high utilization by combining admission control, efficient task-packing, over-commitment, and machine sharing with process-level performance isolation. It supports high-availability applications with runtime features that minimize fault-recovery time, and scheduling policies that reduce the probability of correlated failures. Borg simplifies life for its users by offering a declarative job specification language, name service integration, real-time job monitoring, and tools to analyze and simulate system behavior.

We present a summary of the Borg system architecture and features, important design decisions, a quantitative analysis of some of its policy decisions, and a qualitative examination of lessons learned from a decade of operational experience with it.

1. Introduction

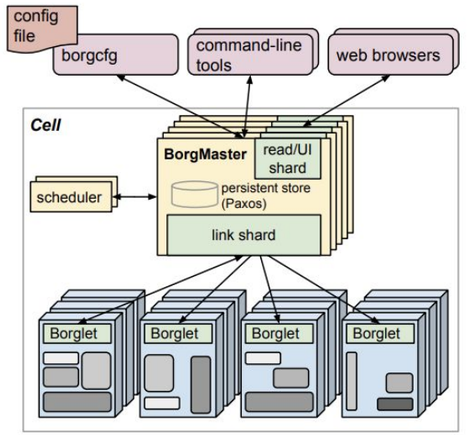


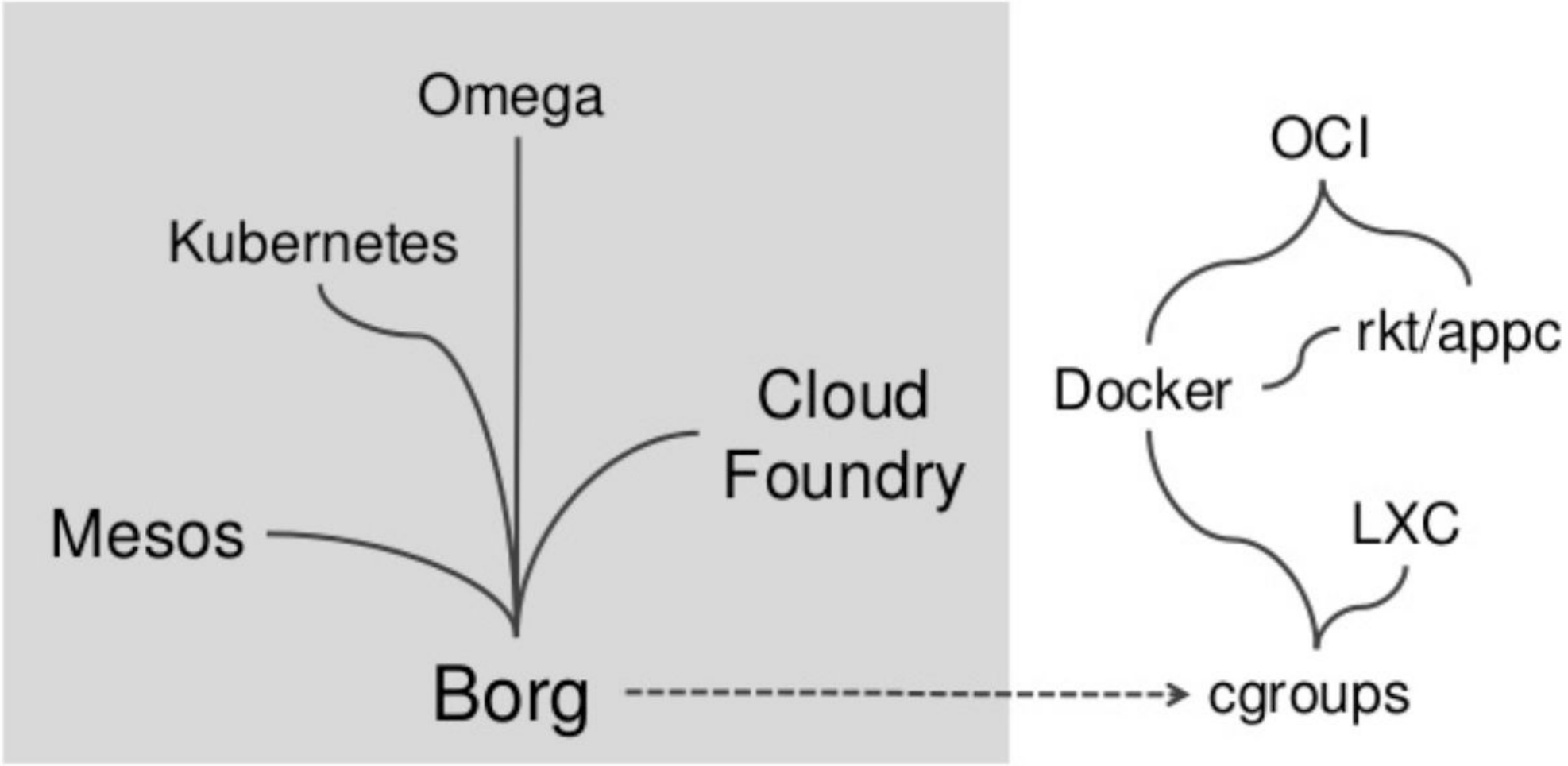
Figure 1: The high-level architecture of Borg. *Only a tiny fraction of the thousands of worker nodes are shown.*

cluding with a set of qualitative observations we have made from operating Borg in production for more than a decade.

Source: https://ai.google/research/pubs/pub43438

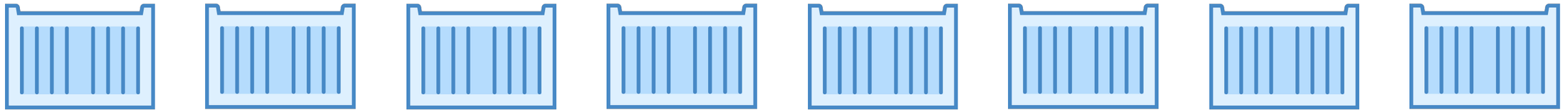
GO JEK

Borg Heritage



Source: Kubernetes Lineage

Application containers

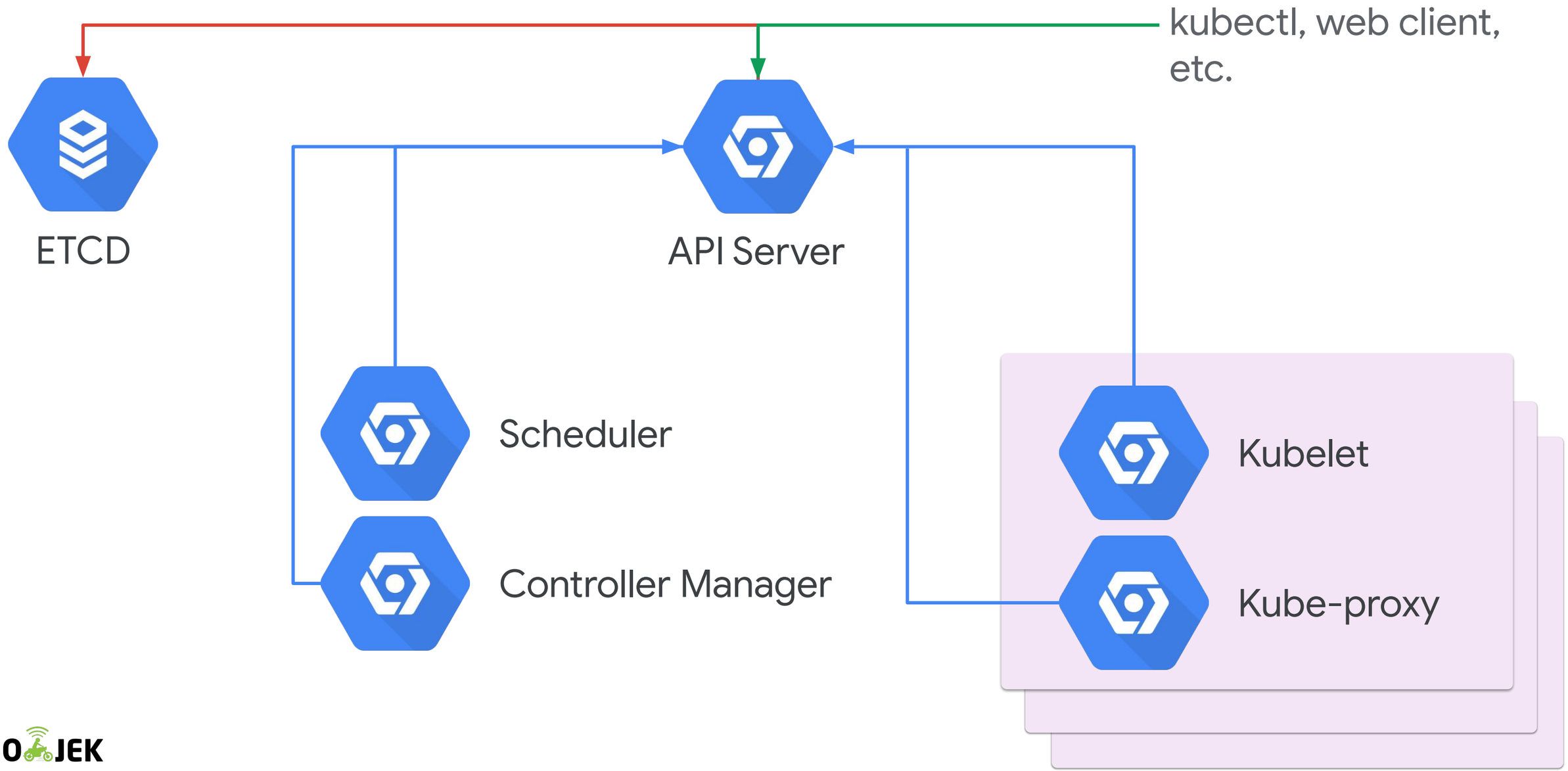


Kubernetes API: Unified Compute Substrate

Homogenous Machine Fleet (Virtual or Physical)

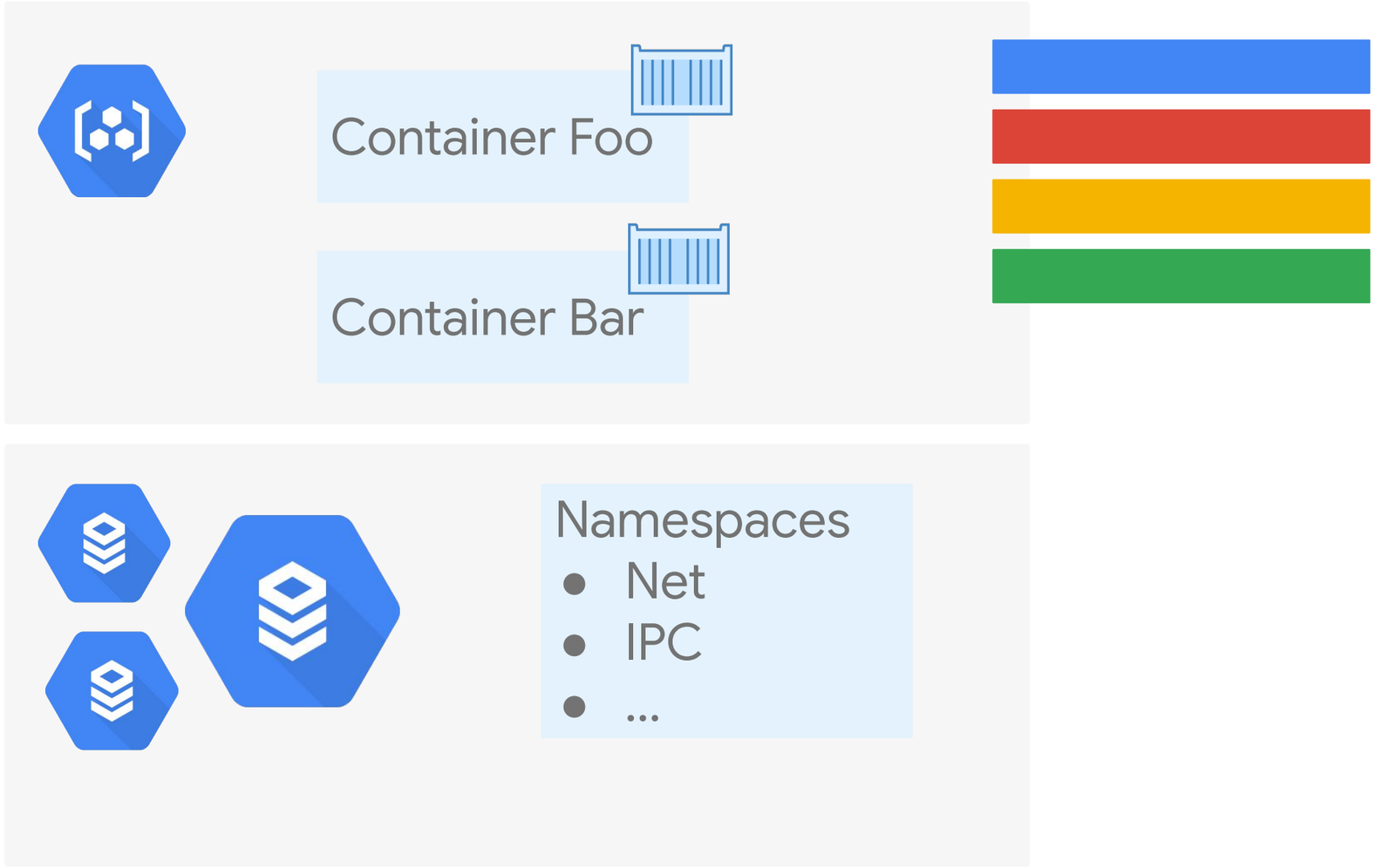
GO JEK

Kubernetes Architecture



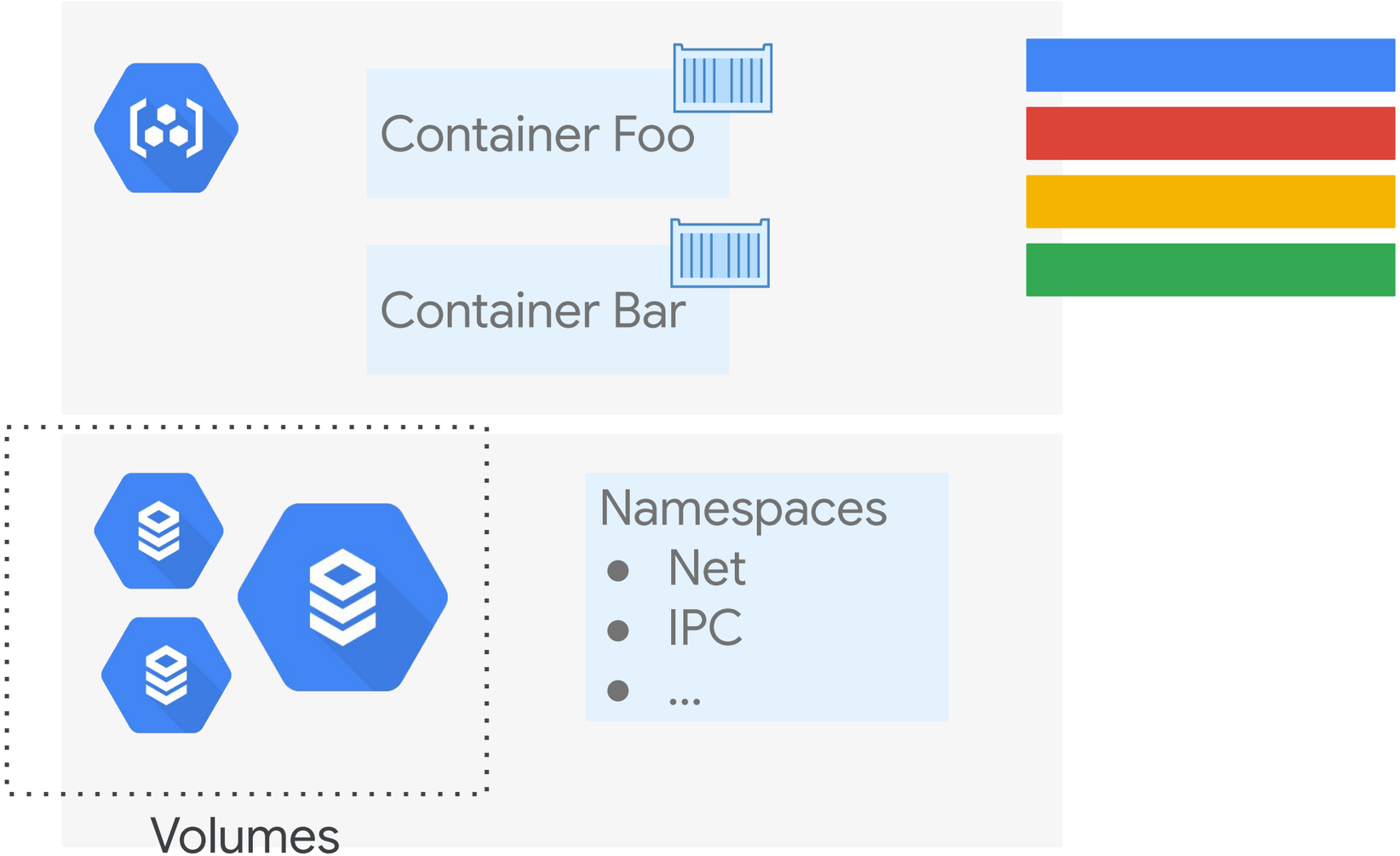
GO JEK

Kubernetes Concepts: Pods



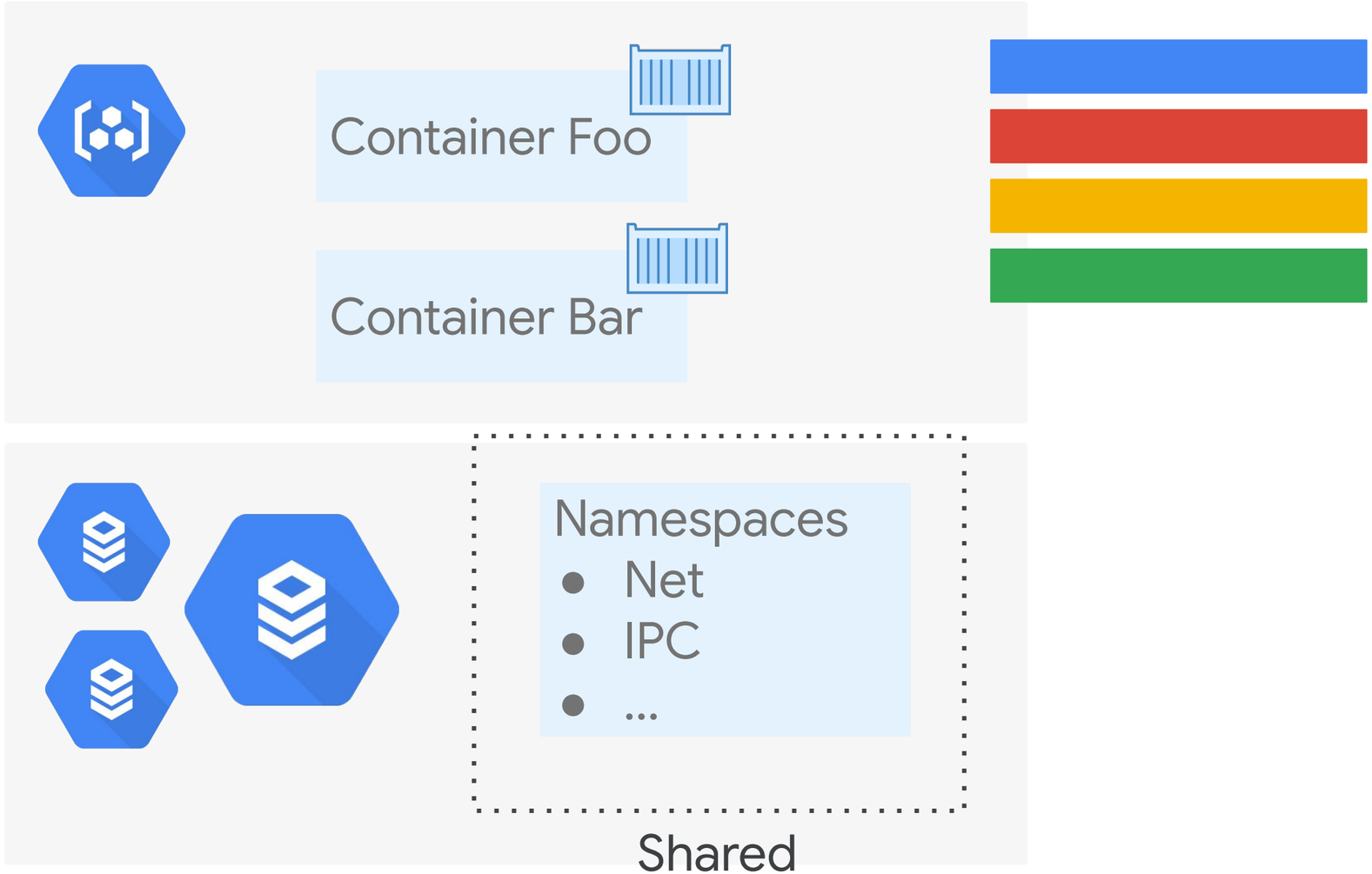
GO JEK

Kubernetes Concepts: Pods



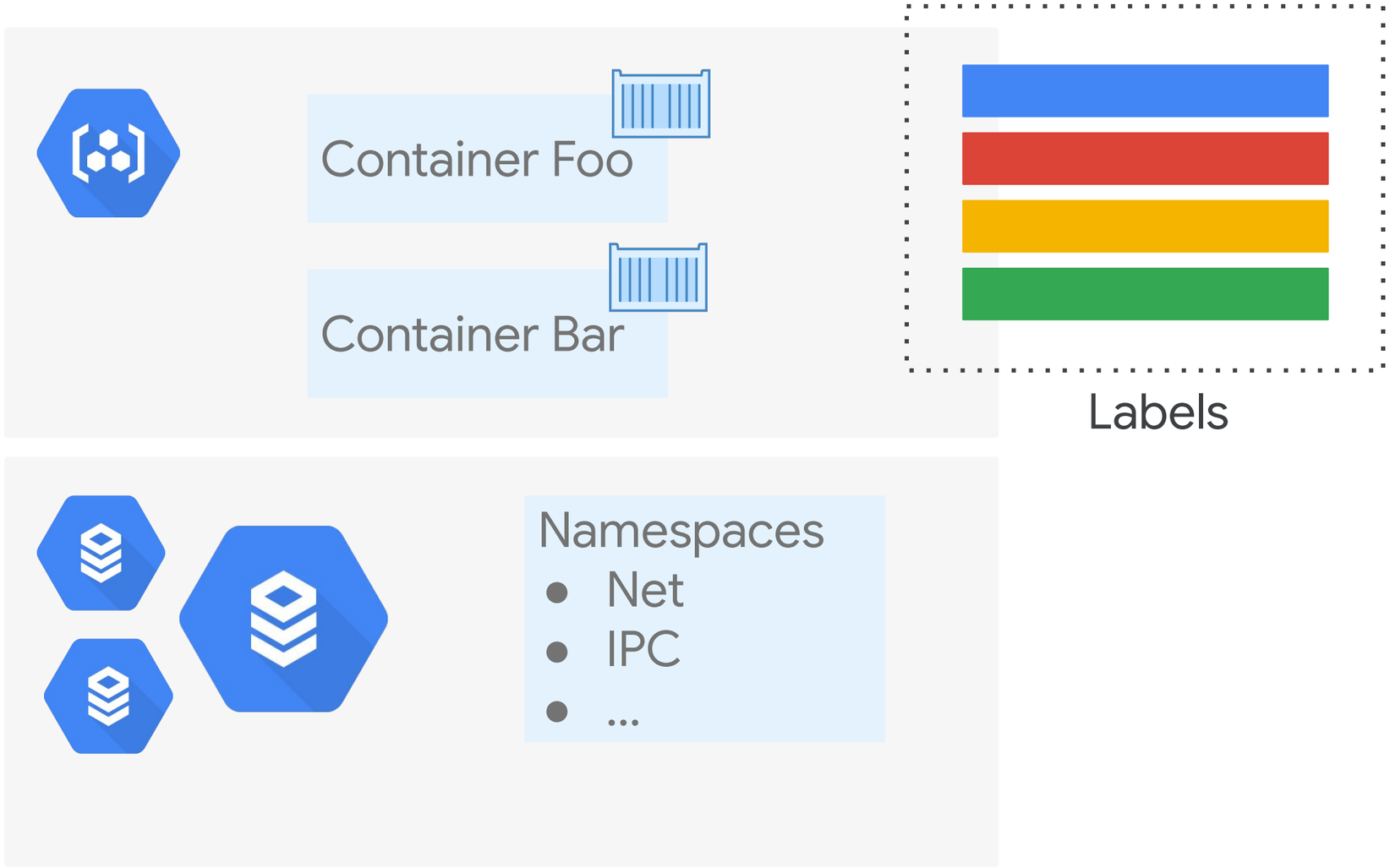
GO JEK

Kubernetes Concepts: Pods



GO JEK

Kubernetes Concepts: Pods



GO JEK

Kubernetes Concepts: Labels

frontend

production

gkuncoro

testing

backend

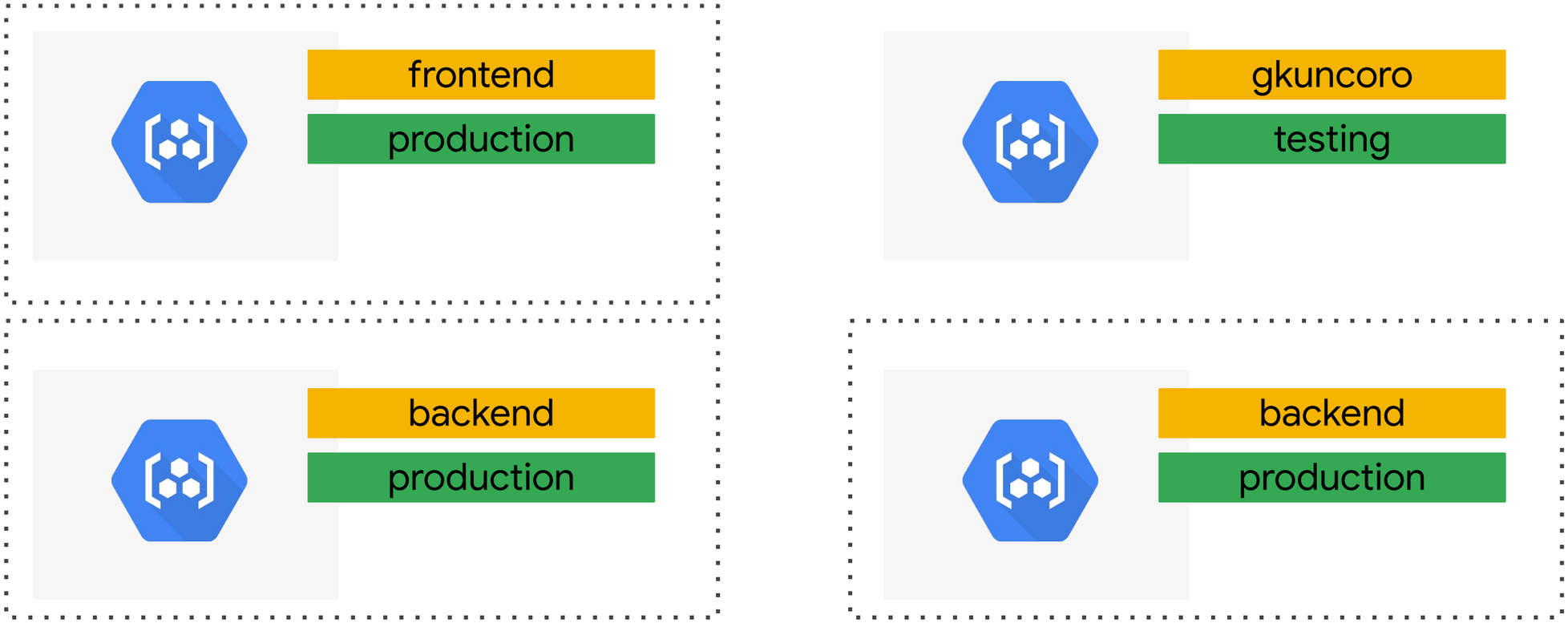
production

backend

production

GO JEK

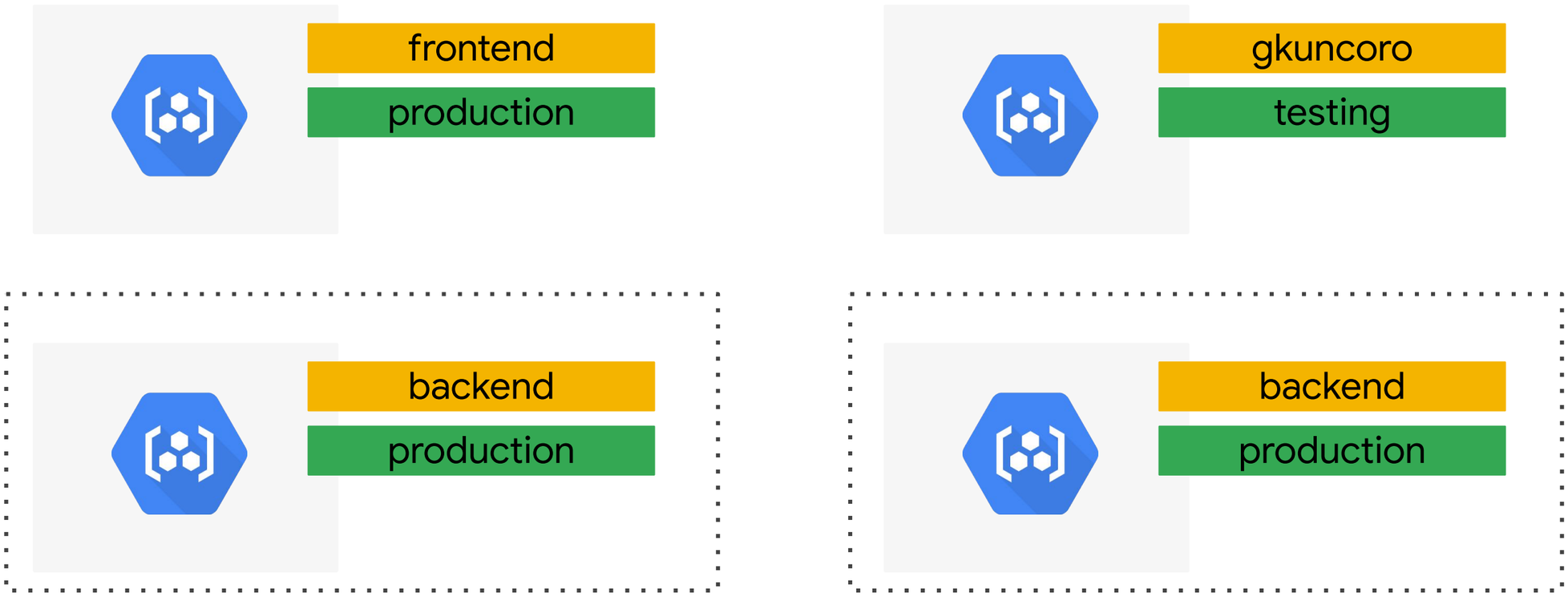
Kubernetes Concepts: Labels



Stage: production

GO JEK

Kubernetes Concepts: Labels



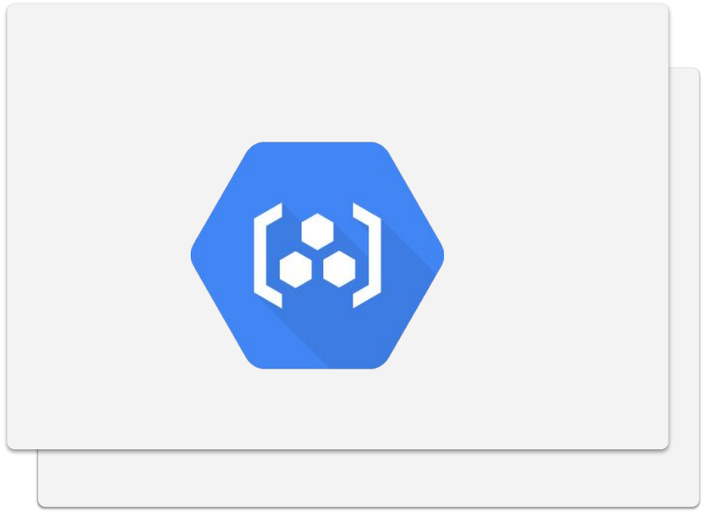
Stage: production

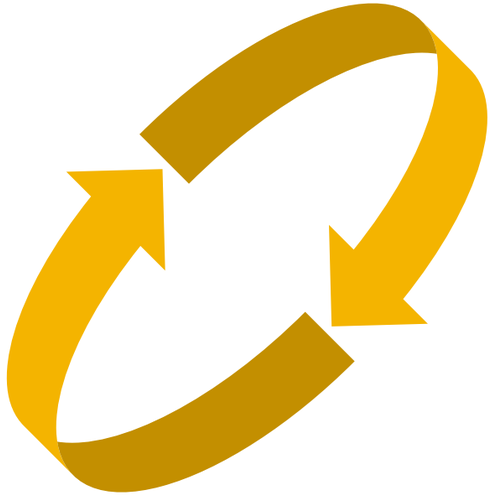
Role: backend

GO JEK

Reconciliation

Desired

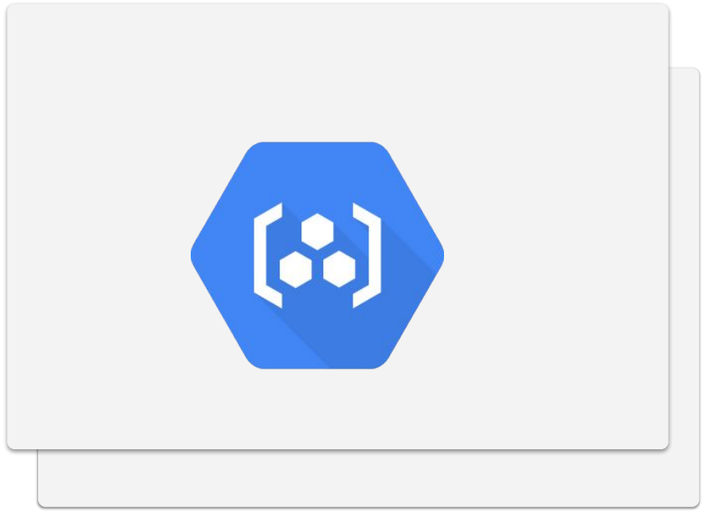




Actual

Reconciliation

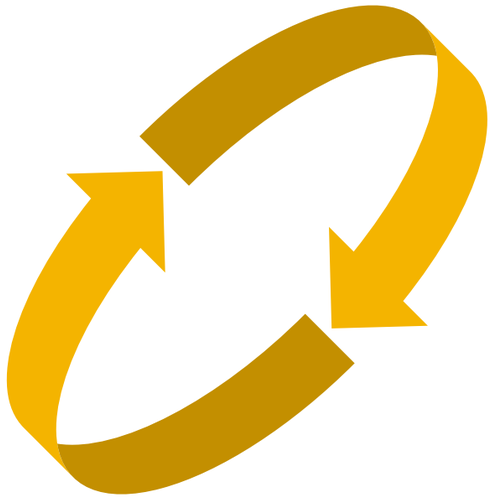
Desired



Pods:

- Foo
- Bar

Create



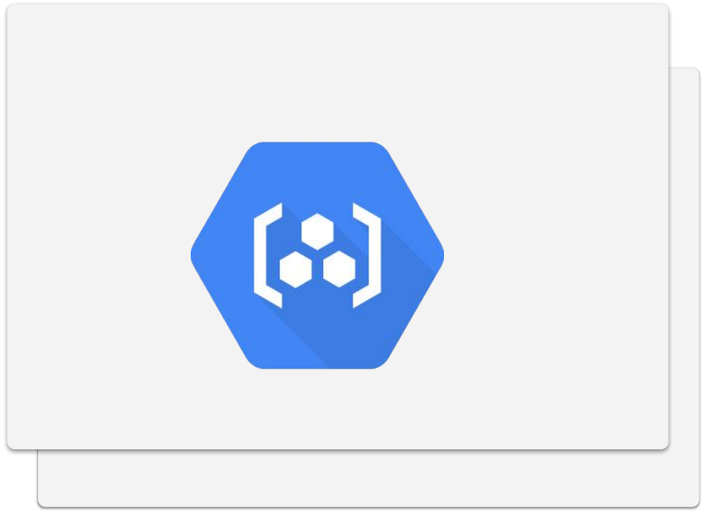
Actual

Pods:

- Foo

Reconciliation

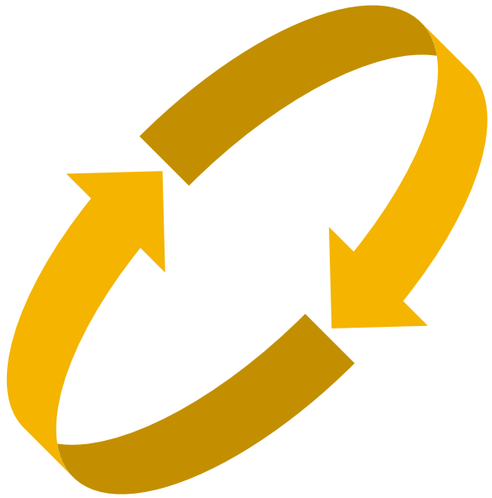
Desired



Pods:

- Foo
- Bar

Create



Create “bar”

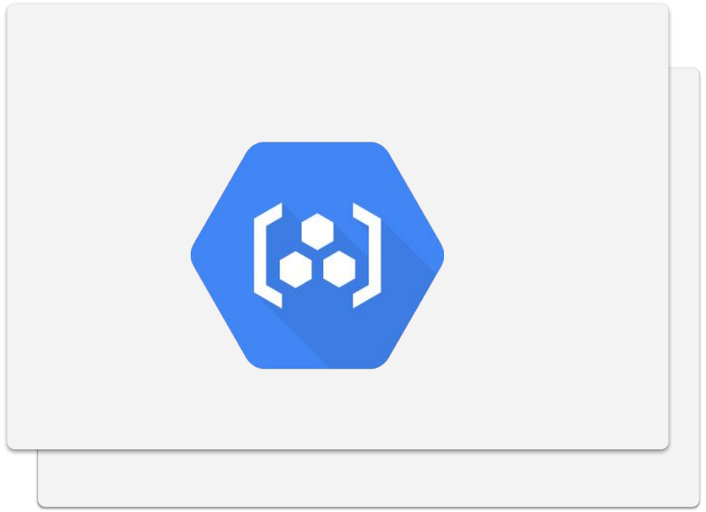
Actual

Pods:

- Foo

Reconciliation

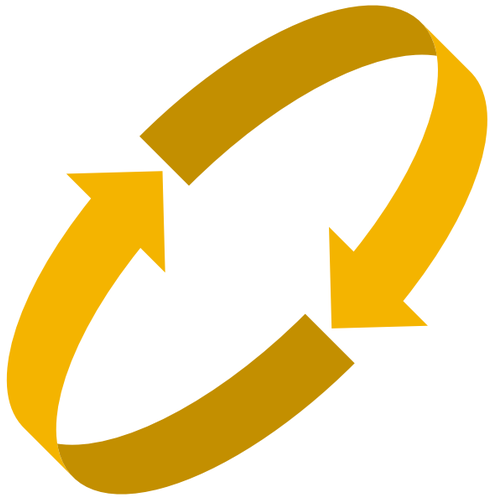
Desired



Pods:

- Foo
- Bar

Health Check



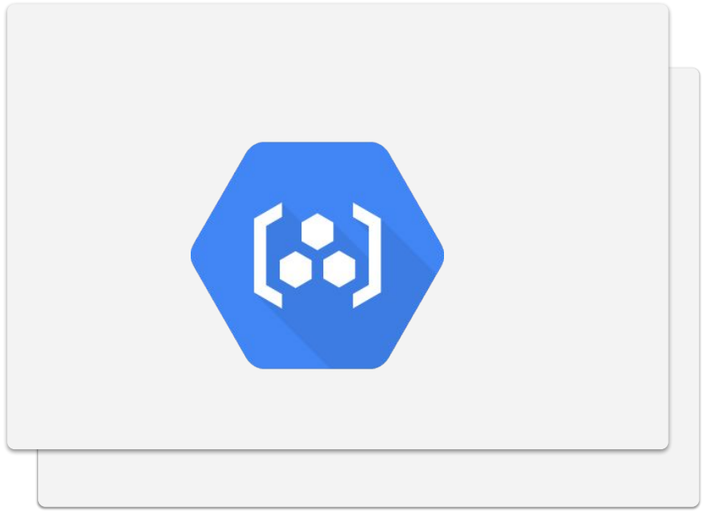
Actual

Pods:

- Foo

Reconciliation

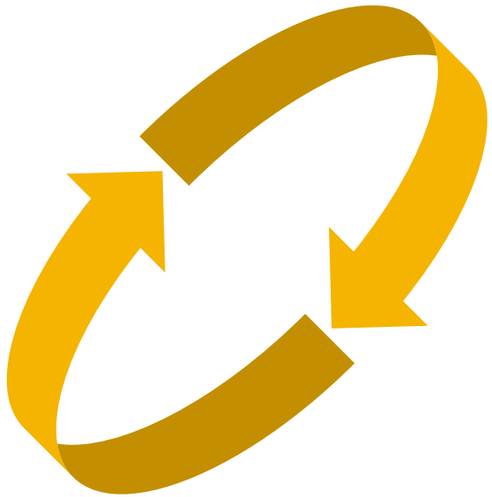
Desired



Pods:

- Foo

Delete



Actual

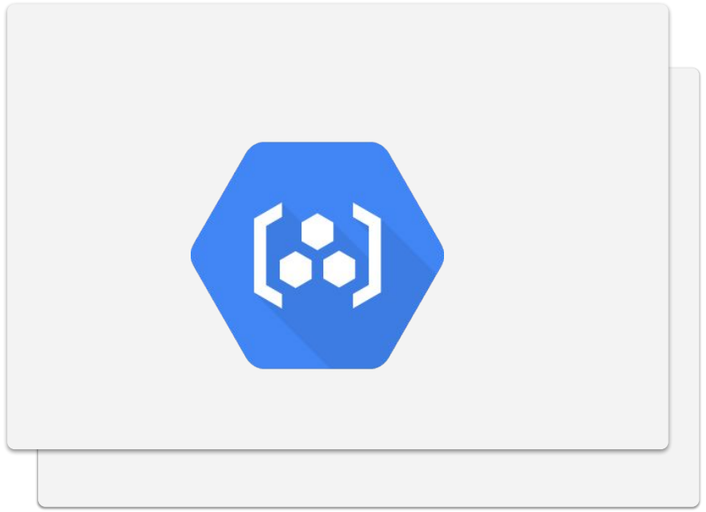
Pods:

- Foo
- Bar

GO JEK

Reconciliation

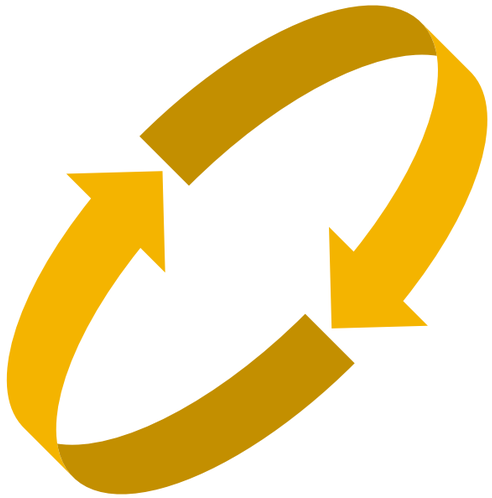
Desired



Pods:

- Foo

Delete



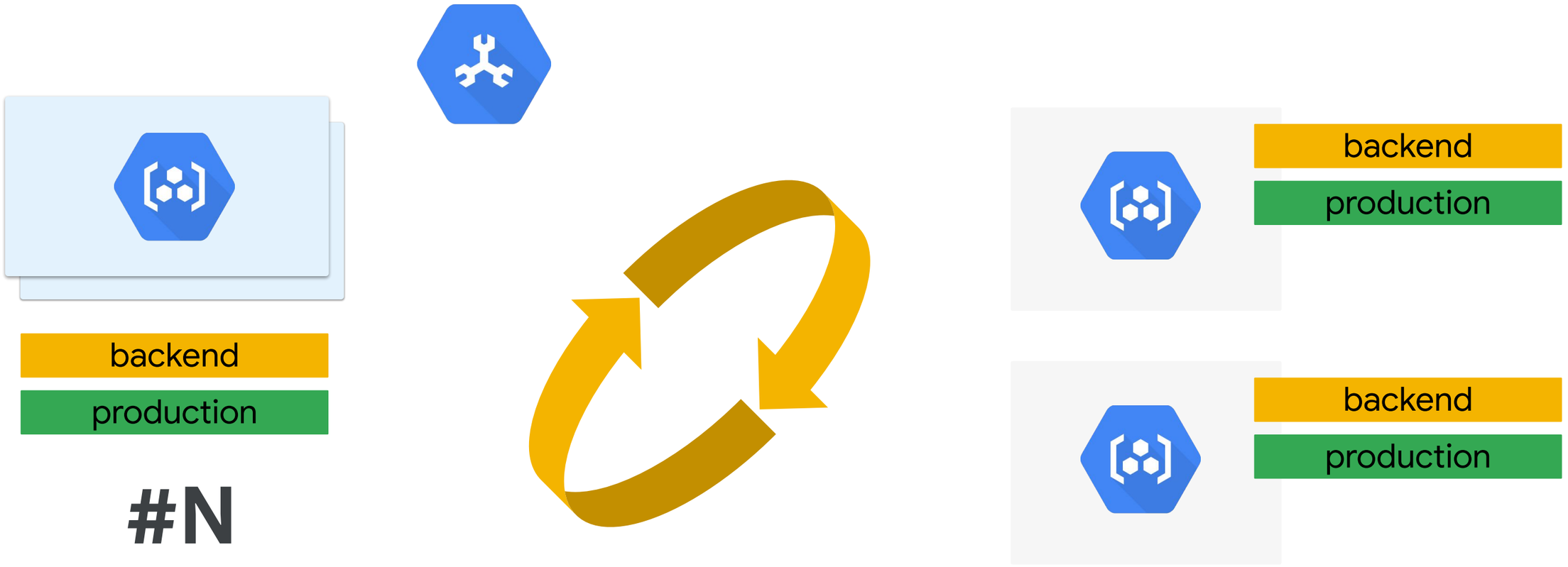
Delete “bar”

Actual

Pods:

- Foo
- Bar

Kubernetes Concepts: Replication Controller



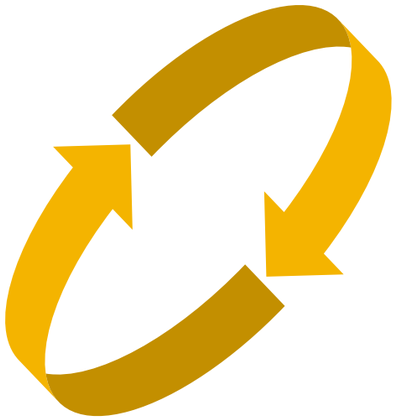
Kubernetes Concepts: Services

name

backend

production

port[s]



1.2.3.4
“name”

backend

production

backend

production

GO JEK

Tools

- GKE / EKS / AKS / DO
- PKS / OKD
- Minikube
- Kubeadm
- Kops
- Kube-aws
- Kubespray
- Kubicorn

Using GKE

Need account on Google Cloud and **gcloud** tool

```
$ gcloud container clusters create jakarta

$ gcloud container clusters list

$ kubectl get nodes
```

GO JEK

Using Minikube

Install **minikube**

```
$ minikube start
```

```
$ kubectl get nodes
```

GO JEK

Using Kubeadm

- Create VM in any provider
- Run on control plane node

  ```
  $ kubeadm init
  ```

- Run on worker node

  ```
  $ kubeadm join --token <token> control-plane-ip
  ```

- Deploy CNI (container networking plugins)

  ```
  $ kubectl create -f https://git.io/weave-kube
  ```

GO JEK

Installing Network Plugins

More than 18+ CNI plugins

- Flannel
- Calico
- Weave Net
- Kube-router
- Romana

https://kubernetes.io/docs/concepts/cluster-administration/networking/

GO JEK

Kubecon Talk on CNI

Source: https://www.youtube.com/watch?v=6DvLOXsHwd4

GOJEK

Using Hyperkube

All-in-one binary

```
gcr.io/google_containers/hyperkube:v1.13.3

$ CONTAINER_IMAGE=gcr.io/google_containers/hyperkube:v1.13.3

$ docker run $CONTAINER_IMAGE /hyperkube apiserver
$ docker run $CONTAINER_IMAGE /hyperkube scheduler
$ docker run $CONTAINER_IMAGE /hyperkube controller-manager
```

GO JEK

Compiling from Source

Kubernetes binary releases

```
$ cd $GOPATH
$ git clone https://github.com/kubernetes/kubernetes
$ cd kubernetes
$ make
```

GO JEK

Installation Considerations

- Which provider?
- Which OS?
- Which networking?
- Where ETCD?
- HA required?

Best Way to Learn Kubernetes Install

- Kelsey Hightower, Kubernetes the hard way

Demo time!

GO JEK

Thank you!

Giri Kuncoro from GO-JEK
@girikuncoro

Q&A

@girikuncoro

https://kubernetes.io

https://www.meetup.com/jakarta-kubernetes

https://t.me/kubernetesindonesia