

# OKE – Kubernetes in OCI

**Your First Journey in Oracle Cloud World**

Brazil Infrastructure and Security Team

LAD Knowledge Team

**May/2020**



## Safe Harbor

---

The following is intended to outline our general product direction. It is intended for information purposes only, and may not be incorporated into any contract. It is not a commitment to deliver any material, code, or functionality, and should not be relied upon in making purchasing decisions. The development, release, timing, and pricing of any features or functionality described for Oracle's products may change and remains at the sole discretion of Oracle Corporation.

Statements in this presentation relating to Oracle's future plans, expectations, beliefs, intentions and prospects are "forward-looking statements" and are subject to material risks and uncertainties. A detailed discussion of these factors and other risks that affect our business is contained in Oracle's Securities and Exchange Commission (SEC) filings, including our most recent reports on Form 10-K and Form 10-Q under the heading "Risk Factors." These filings are available on the SEC's website or on Oracle's website at <http://www.oracle.com/investor>. All information in this presentation is current as of September 2019 and Oracle undertakes no duty to update any statement in light of new information or future events.

\$> whoami



The Redwoods

## Cristiano Hoshikawa

Cloud Solution Engineering  
Oracle Cloud Team



cristiano.hoshikawa@oracle.com



<https://www.linkedin.com/in/cristianohoshikawa/>



<https://github.com/hoshikawa2>

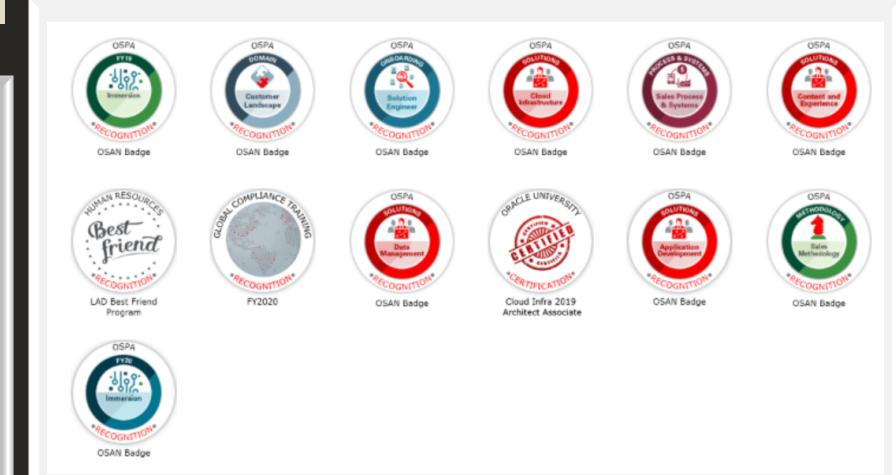
+25 years on IT Market

2.5 years

Working in Oracle Brazil

Solution Eng. Sales Consultant  
LAD Team

DevOps, Cloud Native,  
Integration, Mobile and CRM  
specialist



# Agenda

---

- Abertura
- Cloud Native Computing Foundation
- Kubernetes
- Oracle Cloud Infrastructure Registry & Container Engine for Kubernetes
- Labs
  - Oracle Cloud Infrastructure
  - Containers Docker
  - Kubernetes
  - Repositório de Imagens Docker
  - Deployments
  - Load-Balancer

"The eye sees only what  
the mind is prepared to  
comprehend"

by French philosopher Henri Bergson

# Market Insights



# Business Cycles Are Faster Than Ever

## Before



Over the last 50 years, the average lifespan of companies on the S&P 500 has shrunk from **60** to **18** years

# Software is How Companies Differentiate Themselves

*"More and more major businesses and industries are being run on software and delivered as online services—from movies to agriculture to national defense. Many of the winners are Silicon Valley-style entrepreneurial technology companies that are invading and overturning established industry structures"*

*"As our global economy increasingly comes to run on technology-enabled rails and every company becomes a tech company, demand for high-quality software engineers is at an all-time high. A recent study from Stripe and Harris Poll found that 61 percent of C-suite executives believe access to developer talent is a threat to the success of their business. Perhaps more surprisingly — as we mark a decade after the financial crisis — this threat was even ranked above capital constraints."*

Source: [Software developers are now more valuable to companies than money: Survey](#)



# Software Is Overtaking the World

Time-to-market is key to success

# Buzzwords You Might Hear

A collage of various DevOps and cloud-native technology terms, including Docker, Kubernetes, Istio, Java, Go, CD, CI, Pipelines, Functions, and Microservices.

# Cloud Native Computing Foundation

Overview



# Cloud Native Computing Foundation



Non-profit, part of the Linux Foundation; founded Dec 2015

## Graduated



**kubernetes**  
Orchestration



**Prometheus**  
Monitoring



**OPENTRACING**  
Distributed Tracing API



**fluentd**  
Logging



**gRPC**  
Remote Procedure Call



**rkt**  
Container Runtime



**rkt**  
Container Runtime



**CNI**  
Networking API



**envoy**  
Service Mesh



**ROOK**  
Storage



**spiffe**  
Identity Spec



**SPIRE**  
Identity



**Open Policy Agent**  
Policy



**JAEGER**  
Distributed Tracing



**TUF**  
Software Update Spec



**notary**  
Security



**Vitess**  
Storage



**CoreDNS**  
Service Discovery



**NATS**  
Messaging



**LINKERD**  
Service Mesh



**HELM**  
Package Management



**cloudevents**  
Serverless



**TELEPRESENCE**  
Tooling



**HARBOR**  
Registry



**OPENMETRICS**  
Metrics Spec



**KV**  
Key/Value Store

## Platinum members:



# Today the Linux Foundation is much more than Linux



## Security

We are helping global privacy and security through a program to encrypt the entire internet.



## Networking

We are creating ecosystems around networking to improve agility in the evolving software-defined datacenter.



## Cloud

We are creating a portability layer for the cloud, driving de facto standards and developing the orchestration layer for all clouds.



## Automotive

We are creating the platform for infotainment in the auto industry that can be expanded into instrument clusters and telematics systems.



## Blockchain

We are creating a permanent, secure distributed ledger that makes it easier to create cost-efficient, decentralized business networks.



## Web

We are providing the application development framework for next generation web, mobile, serverless, and IoT applications.



We are regularly adding projects; for the most up-to-date listing of all projects visit [lfprojects.org](http://lfprojects.org)

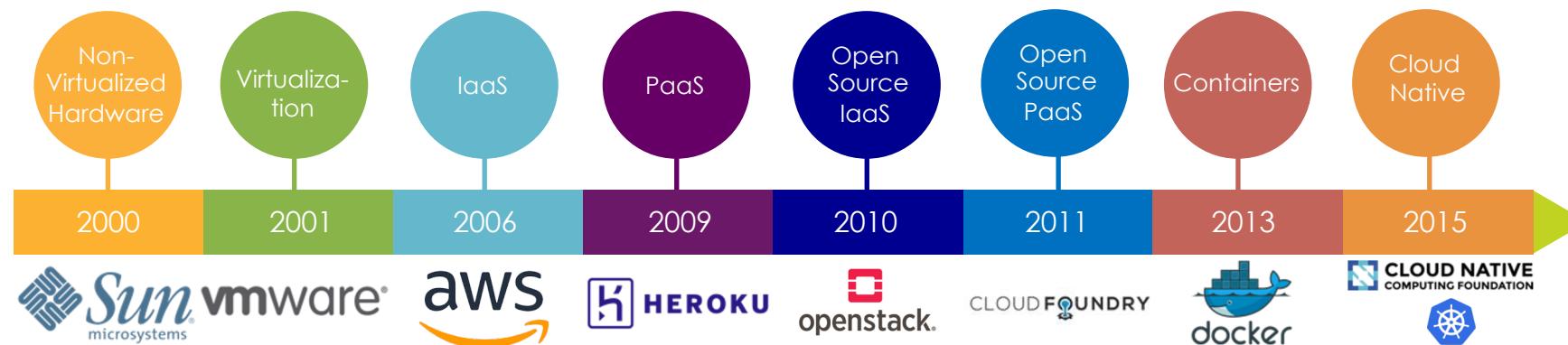


# From Virtualization to Cloud Native



Cloud native computing uses an **open source software** stack to:

- **segment** applications **into microservices**,
- **package** each part **into its own container**
- and **dynamically orchestrate** those containers **to optimize resource utilization**



# 290+ Members and Growing



**Platinum Members**

**Gold Members**

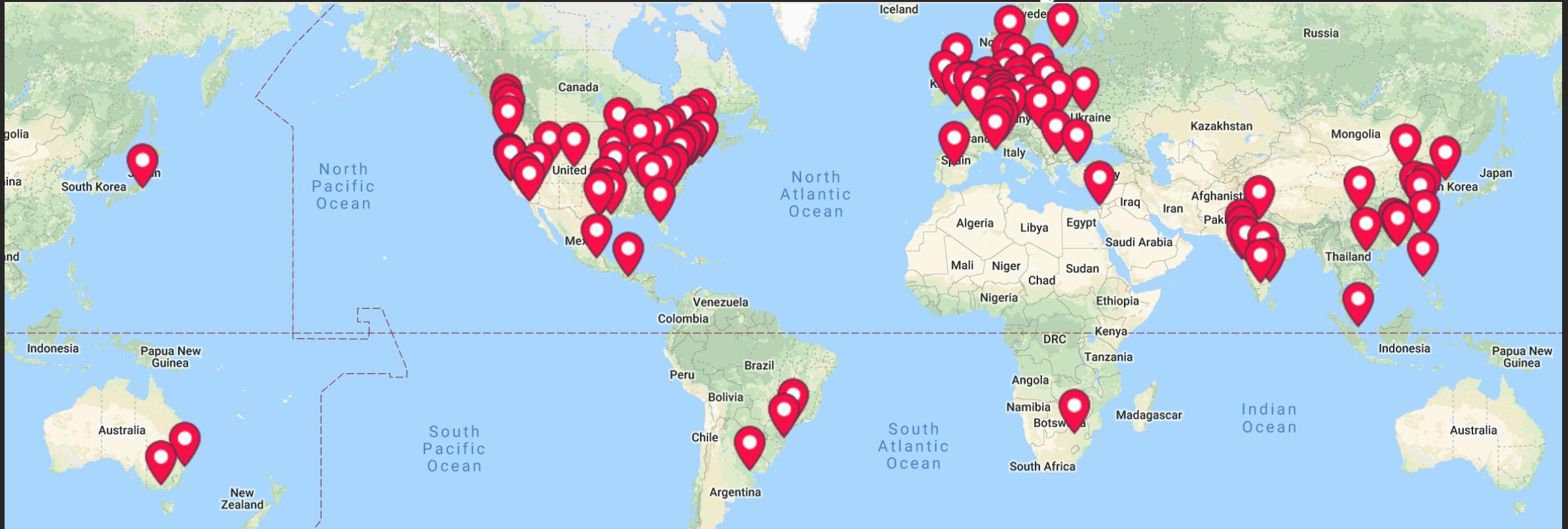
**End User Members**

**End User Supporters**

**Academic/Nonprofit**

Bloomberg, CapitalOne, Comcast, ebay, DENSO, DiDi, GitHub, Goldman Sachs, Indeed, Intuit, JD.COM, Morgan Stanley, NAIC, NCSOFT, Pinterest, salesforce, Spotify, SQUARESPACE, twilio, twitter, M Advanced Research Computing Technology Services University of Michigan, WIKIMEDIA FOUNDATION, YAHOO! JAPAN, TWO SIGMA, Werkspot, woorkrank, workday, WPengine, zalando, zendesk, adidas, box, cruise, FORM3 FINANCIAL CLOUD, Kuelap, Layer, Nasdaq, The New York Times, Clark, PUSHER, reddit, ricardo.ch, SAP Concur, shopify, showmax, Spredfast, stix, textkernel, THREDUP, ticketmaster, CableLabs, CLOUD FOUNDRY, GOLDEN GATE UNIVERSITY, INTERNET, NAIC, NIPR, SEL, ADVANCED RESEARCH COMPUTING TECHNOLOGY SERVICES UNIVERSITY OF MICHIGAN, WIKIMEDIA FOUNDATION.

# CNCF Worldwide Community



**Cloud Native Computing  
Foundation (CNCF)**

Members  
**79,470**

Groups  
**155**

Countries  
**36**

# Cloud Native Value Propositions



# Avoid Vendor Lock-in

Open source software stack enables deployment  
on any public, private cloud or hybrid cloud



# Enable Unlimited Scalability

Scales from several nodes on your laptop to tens of thousands of self-healing multi-tenant nodes



# Increase Agility and Maintainability

By splitting applications into microservices  
with explicitly described dependencies



# Achieve Resiliency

To failures of individual containers, machines, and even data centers and to varying levels of demand



# Improve Efficiency and Resource Utilization

Via a central orchestrating process that dynamically manages and schedules microservices



# Enables High Performance

High-performing teams deploy  
more frequently and have  
much faster lead times.



2016 State of DevOps Report [Infographic](#) from Puppet



# The “Captain” of yours Containers





kubernetes

# What is Kubernetes?

Kubernetes (κυβερνήτης, Greek for "governor", "helmsman" or "captain")

Kubernetes is a "Container Orchestrator" or "Cluster Manager".

- Places containers on nodes
- Recovers automatically from failure
- Basic monitoring, logging, health checking
- Enables containers to find each other.

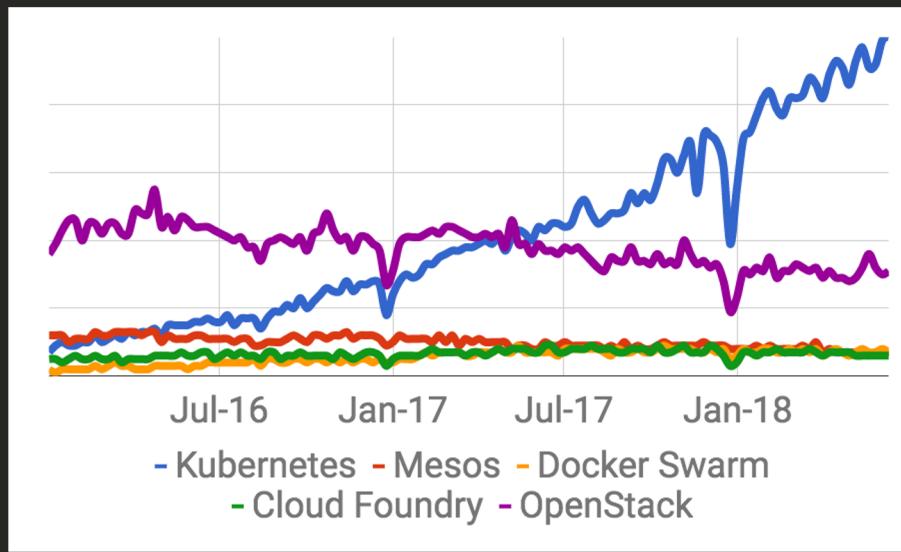
By A.amitkumar [CC BY-SA 3.0], from Wikimedia Commons

 CLOUD NATIVE COMPUTING FOUNDATION

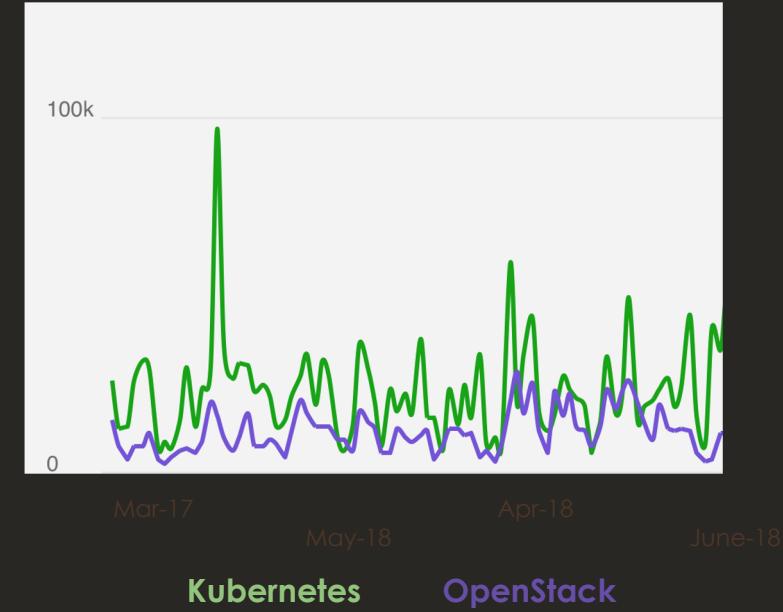


# Kubernetes in Search Trends

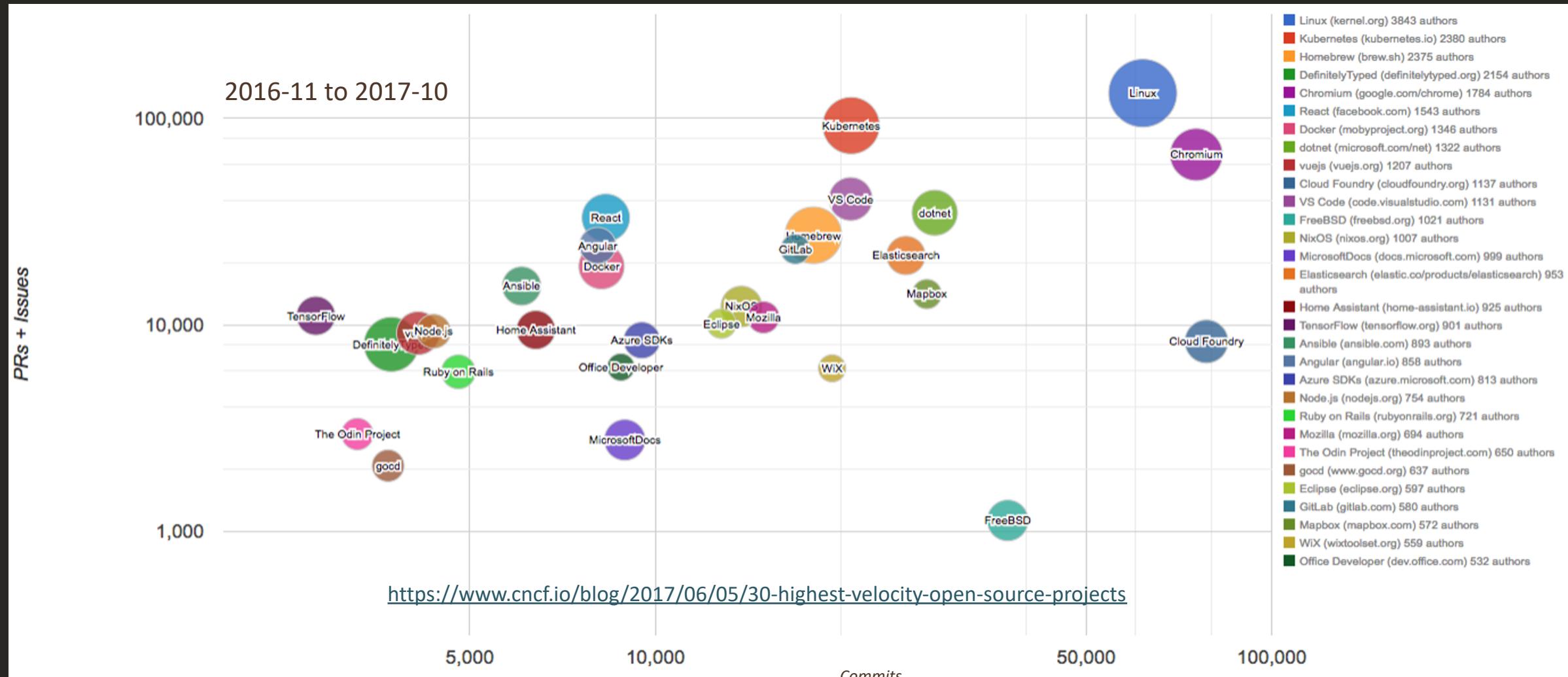
Google Trends



WeChat



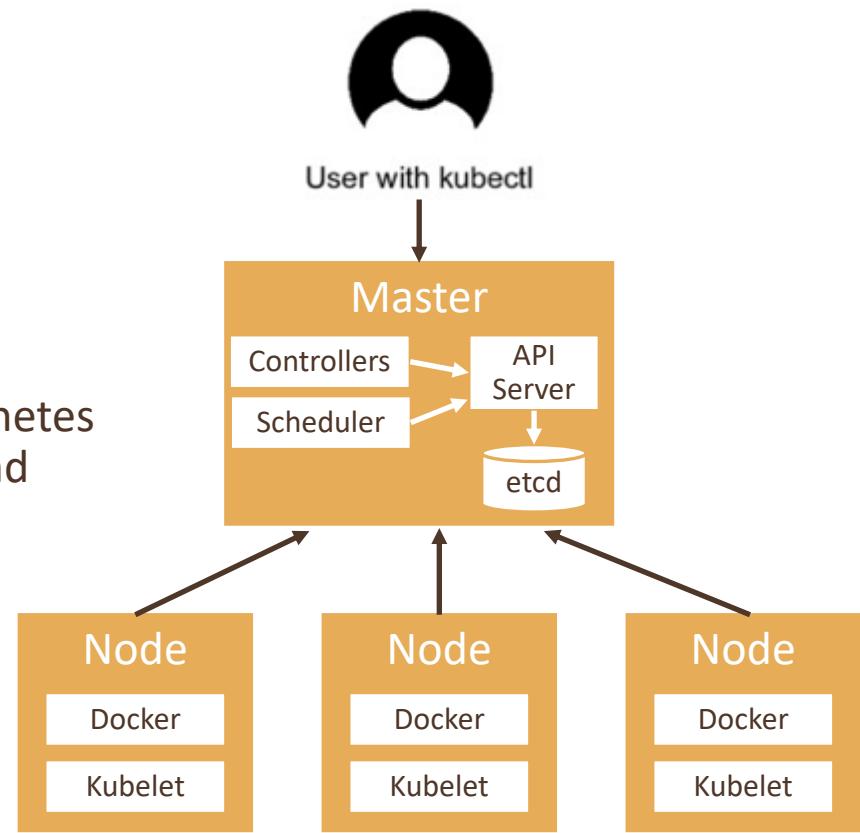
# 30 Highest Velocity Open Source Projects



# Basics: Cluster, Master and Nodes

- Kubernetes coordinates a **highly available cluster** of computers that are connected to work as a single unit (**Kubernetes cluster**). The abstractions in Kubernetes allow you to deploy Docker containerized applications to a cluster without tying them specifically to individual machines.
- You manage your cluster through web UI or through CLI (**kubectl**)
- A Kubernetes cluster consists of two types of resources:
  - The **Master** coordinates the cluster
  - **Nodes** are workers that run the containers (application)
- A node is a VM or a physical computer that is a worker machine in a Kubernetes cluster. Each node has a **Kubelet**, which is the agent for managing nodes and communicating with the Kubernetes master.
- Nodes are where the Docker containers run.

<https://kubernetes.io/docs/concepts/overview/what-is-kubernetes/>



# Basics: Deployments

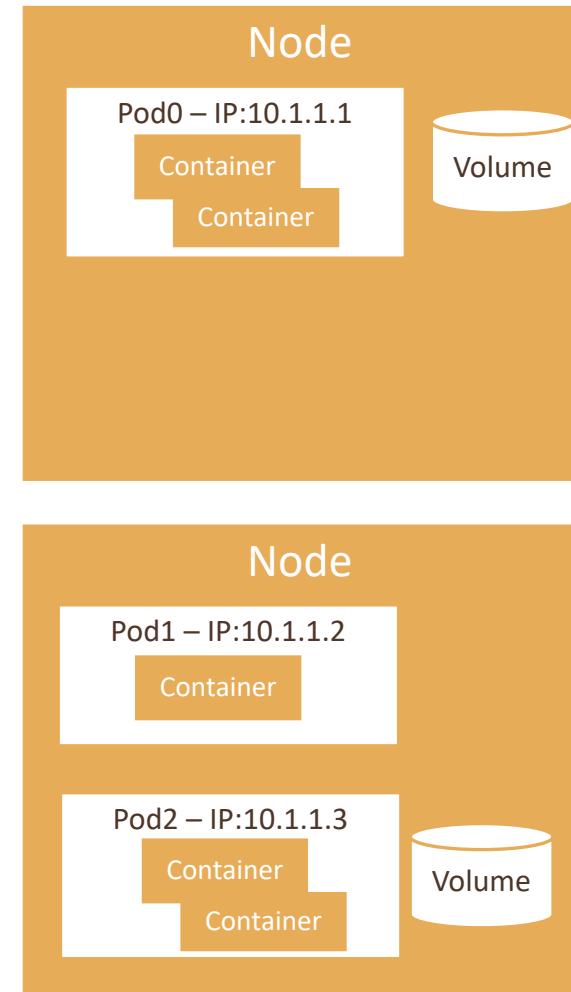
- Once you have a running Kubernetes cluster, you can deploy your containerized applications on top of it. To do so, you create a Kubernetes **Deployment** configuration. The Deployment, written in YAML, instructs Kubernetes how to create and update instances of your application.
- Once you've created a Deployment, the Kubernetes master schedules the containers (application instances) onto individual Nodes in the cluster.

```
nginx-deployment.yaml □

apiVersion: apps/v1beta2
kind: Deployment
metadata:
  name: nginx-deployment
spec:
  replicas: 3
  template:
    metadata:
      labels:
        app: nginx
    spec:
      containers:
        - name: nginx
          image: nginx:1.7.9
          ports:
            - containerPort: 80
```

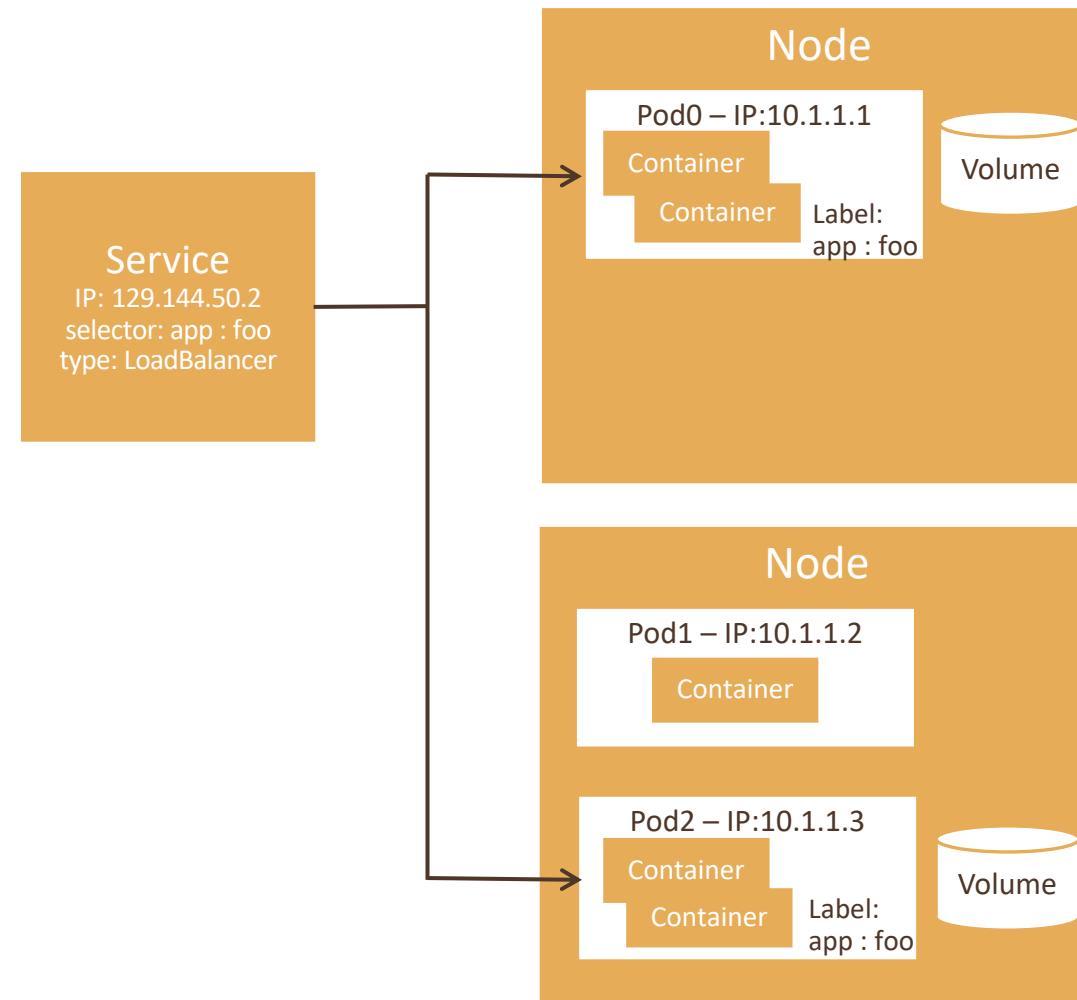
# Basics : Pods

- When you create a Deployment, Kubernetes create **Pods** to host your application instance.
- A **Pod** is a Kubernetes abstraction that represents a group of one or more application containers, and some shared resources for those containers, such as storage resources and a unique network IP.
- Each Pod is tied to the Node where it is scheduled, and remains there until termination (according to restart policy) or deletion. In case of a Node failure, identical Pods are scheduled by the Controller on other available Nodes in the cluster.



# Basics: Applications in Kubernetes – how to access them?

- A Kubernetes **Service** is an abstraction which defines a logical set of Pods running somewhere in your cluster. When created, each Service is assigned a unique IP address. Although each Pod has a unique IP address, those IPs are not exposed outside the cluster without a Service.
- Services allow your applications to receive traffic whether from other Pods within the cluster or outside the cluster, for example, from the Internet via a Load Balancer.



# Other Terminology

- Namespace: A logical isolation method. Most resources are associated with a namespace. You can then group similar workloads and enforce different policies.
- Replication Sets: This controls how many identical copies of a Pod should be running somewhere in the cluster. This is a useful feature, when you may want a Pod in each availability domain for high availability or you want to scale the application and create more Pods.
- Persistent Volume: Volumes setup by the administrator that provide persistent storage for applications that require long lived data. The deployment would use a Persistent Volume Claim to access a Persistent Volume

# Examples of K8s Dashboard and Kubectl Output

The screenshot shows the Kubernetes Dashboard interface at `localhost:8001/api/vt/namespaces/kube-system/services/kubernetes-dashboard/proxy/#/overview?namespace=default`. The left sidebar contains navigation links for Cluster, Namespaces, Nodes, Persistent Volumes, Roles, Storage Classes, and various Workloads like Daemon Sets, Deployments, Jobs, Pods, Replica Sets, Replication Controllers, and Stateful Sets. The main content area displays four tables:

- Deployments:** Shows two entries: `hello-world-dep` (3/3 pods) and `nginx-ingress-controller` (1/1 pod).
- Pods:** Shows two entries: `hello-world-dep-2844294529-b4944` and `hello-world-dep-2844294529-6tcnj`.
- Replica Sets:** Shows one entry: `hello-world-dep-2844294529`.
- Services:** Shows one entry: `hello-world-dep` (Cluster IP: 10.96.108.205).

```
root@demo:~/kube# kubectl get all
NAME                                         READY   STATUS    RESTARTS   AGE
po/default-http-backend-726995137-bm9lk   1/1    Running   0          15d
po/hello-world-dep-2844294529-6tcnj      1/1    Running   0          17h
po/hello-world-dep-2844294529-b4944      1/1    Running   0          17h
po/hello-world-dep-2844294529-k1hpf       1/1    Running   0          17h
po/nginx-ingress-controller-3515728927-719px 1/1    Running   3          15d
po/time-api-1011262983-mjgxt             1/1    Running   0          8d
po/time-web-3773680739-0jpxx              1/1    Running   0          12d

NAME                           CLUSTER-IP   EXTERNAL-IP   PORT(S)           AGE
svc/default-http-backend     10.96.198.77 <none>        80/TCP            15d
svc/hello-world-dep          10.96.108.205 <none>        8090/TCP         13d
svc/kubernetes                10.96.0.1    <none>        443/TCP          15d
svc/nginx-ingress            10.96.172.60  129.213.9.145  80:32080/TCP,443:32443/TCP 15d
svc/time-apl                 10.96.174.172 <none>        8080/TCP         15d
svc/time-web                 10.96.171.236 <none>        8000/TCP         15d

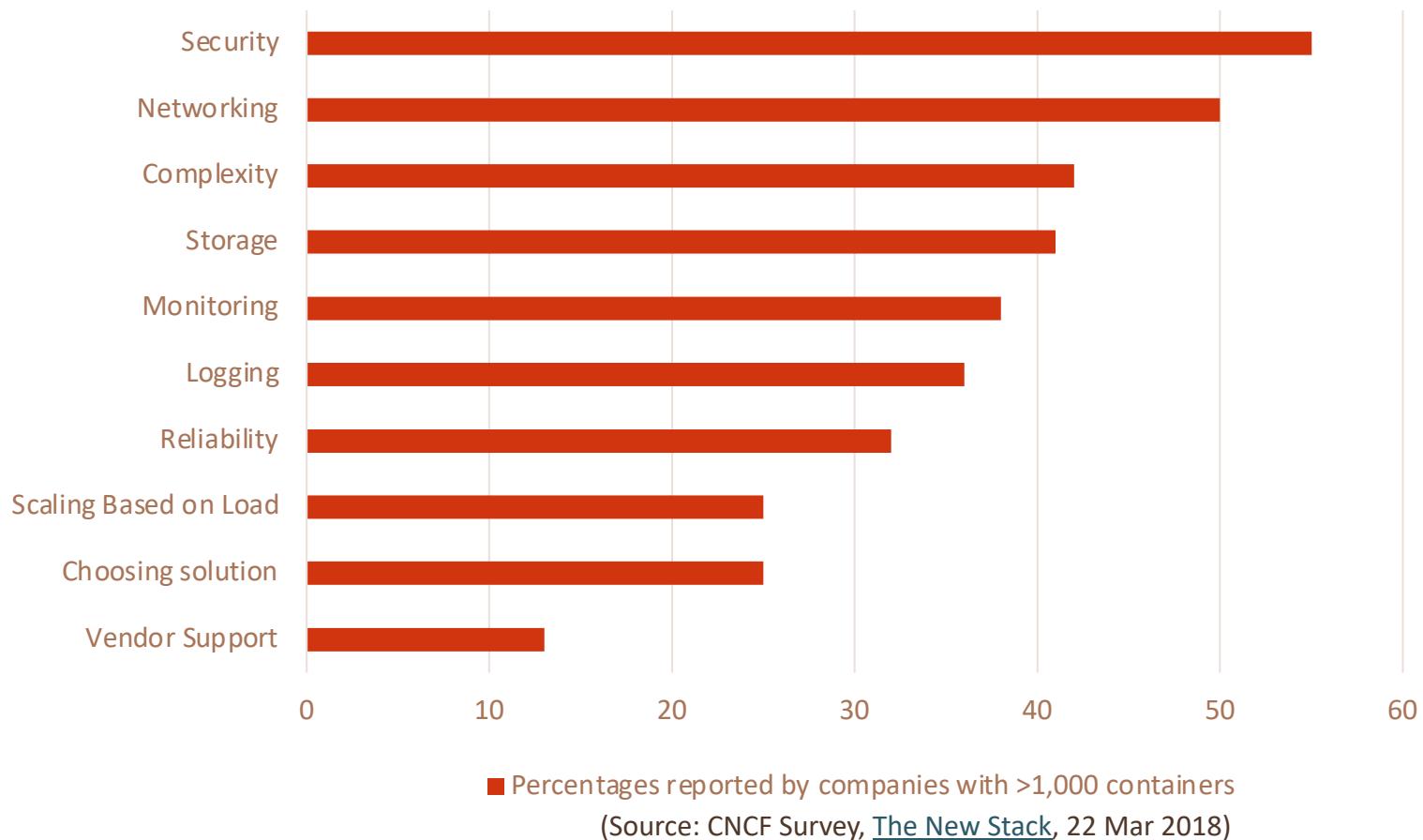
NAME          DESIRED   CURRENT   UP-TO-DATE   AVAILABLE   AGE
deploy/default-http-backend  1          1          1            1           15d
deploy/hello-world-dep       3          3          3            3           13d
deploy/nginx-ingress-controller 1          1          1            1           15d
deploy/time-apl              1          1          1            1           15d
deploy/time-web              1          1          1            1           15d

NAME          DESIRED   CURRENT   READY   AGE
rs/default-http-backend-726995137  1          1          1           15d
rs/hello-world-dep-1029076797  0          0          0           13d
rs/hello-world-dep-1113067381  0          0          0           10d
```

```
root@demo:~/kube# kubectl get pods -o wide
NAME                                         READY   STATUS    RESTARTS   AGE   IP           NODE
default-http-backend-726995137-bm9lk   1/1    Running   0          15d   10.244.76.4  129.213.36.213
hello-world-dep-2844294529-6tcnj      1/1    Running   0          17h   10.244.43.65 129.213.22.34
hello-world-dep-2844294529-b4944      1/1    Running   0          17h   10.244.67.67 129.213.51.49
hello-world-dep-2844294529-k1hpf       1/1    Running   0          17h   10.244.76.66 129.213.36.213
nginx-ingress-controller-3515728927-719px 1/1    Running   3          15d   10.244.43.4   129.213.22.34
time-api-1011262983-mjgxt             1/1    Running   0          8d    10.244.67.55 129.213.51.49
time-web-3773680739-0jpxx              1/1    Running   0          12d   10.244.76.28 129.213.36.213
```

# Kubernetes Challenges

- Managing Kubernetes Infrastructure, upgrading, security
- Container networking & persistent storage
- Managing Teams & Access
- CI/CD Integration, automated testing, conditional release



# Key Containers / Orchestration Use Cases



	Share	Container Use Cases	Orchestration Use Cases
Development	<b>65%</b>	Developer productivity; Consistent appstacks in Dev, Test & Production	Automated deploys to accelerate application release cadence
CI/CD/DevOps	<b>48%</b>	Containerized dependencies; Container registries;	Rolling updates and reversals
Operations	<b>41%</b>	Standardized environments for dev, testing and operations	Resilient, self-healing systems; High Availability; Elastic Scalability
Refactor Legacy Apps	<b>34%</b>	Refactor from N-tier to portable containerized applications	Run distributed, stateful apps on scale-out infrastructure
Migrate to Cloud	<b>33%</b>	Move entire appstacks and see them run identically in the cloud	Cloud bursting; Reduce infrastructure costs by avoiding over-provisioning
New Microservice Apps	<b>32%</b>	Create small purpose-built services that can be assembled to scalable custom applications	Dynamically manage large-scale microservices infrastructure

SOURCE: THE EVOLUTION OF THE MODERN SOFTWARE SUPPLY CHAIN, DOCKER SURVEY 2016

# Oracle Cloud Infrastructure Registry & Container Engine for Kubernetes

# Oracle's Commitment to Containers and Open Source

- Oracle's participation in open source community
  - Active Participation – Cloud Native Compute Foundation and Kubernetes (see <https://www.cncf.io/about/members/>)
  - No forked code – straight from the source
  - Continue precedence of Java, MySQL, Linux
- Lead by example
  - Oracle software on Docker Store
  - Kubernetes engineering in CNCF
  - Java SE/EE open sourcing; transparent processes
- Innovate in open source
  - Utilities like K8S terraform install, smith, railcar, crashcart
- Sponsor & contribute to key conferences

## Active Community Participation



## Innovate in Open Source



OpenJDK 4 Containers



Kubernetes

## Open Sourcing Docker & K8s Utilities



K8S  
installer



smith



crashcart



railcar

# Oracle Strategy for Container Based Infrastructure

## Complete Cloud Native Stack

Deliver tools and services that are complete, integrated and open

- Continuous Integration & Deployment, Container Registry, Orchestration/Scheduling, Management/Operations, Analytics/Introspection
- With an application development platform for serverless and microservices

## Open Source

Actively participate in com-munity driven open source container technologies

- Investing in Kubernetes, Docker, Fn, & CNCF, DevTools, and DevOps, with engineering resources, code contributions & sponsorships
- Active support from Oracle's portfolio of open source assets (Java, etc.)

## Managed Services

Differentiate on quality of implementation, service and operational excellence

- Full, transparent management
- Open Source Compatible
- Standards compliant
- Deployed to Oracle Cloud Infrastructure
- Enterprise grade performance, security, HA, and governance



# Four Ways to Run Kubernetes on Oracle Cloud Infrastructure

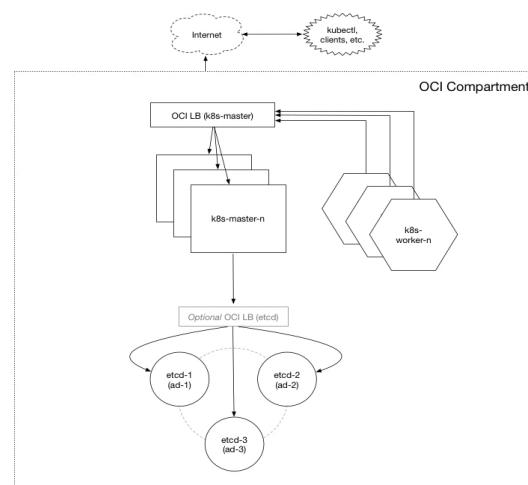
**Roll Your Own, Pre-Built Installer, Managed Service, Tools for Cloud@Customer**

Oracle Cloud  
Infrastructure



Roll-Your-Own  
Container Management

Quickstart Experience  
(OSS Terraform Installer on Github)



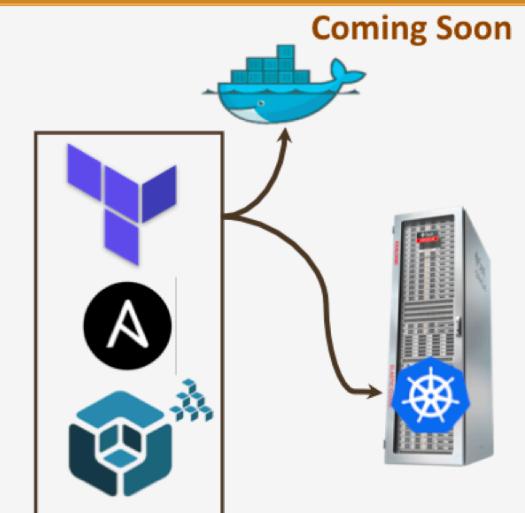
Pre-Built Kubernetes  
Installer

Container Engine for  
Kubernetes



Enterprise Class Managed  
Kubernetes Service

Manager for Kubernetes  
on Cloud@Customer



Enterprise Tools for  
Managing Kubernetes



# Introducing Container Engine for Kubernetes - OKE

What is It?

- Managed Kubernetes container service to deploy and run your own container based apps
- Tooling to create, scale, manage & control your own standard Kubernetes clusters instantly

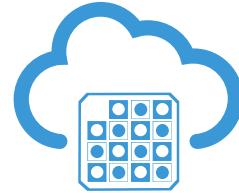
What Problems  
Does it Solve?

- Too complex, costly and time consuming to build & maintain environments
- Too hard to integrate Kubernetes with a registry and build process for container lifecycle management
- Too difficult to manage and control team access to production clusters

Key Benefits

- Enables developers to get started and deploy containers quickly. Gives DevOps teams visibility and control for Kubernetes management.
- Combines production grade container orchestration of open Kubernetes, with control, security, IAM, and high predictable performance of Oracle's next generation cloud infrastructure

# Introducing Oracle Cloud Infrastructure Registry - OCIR



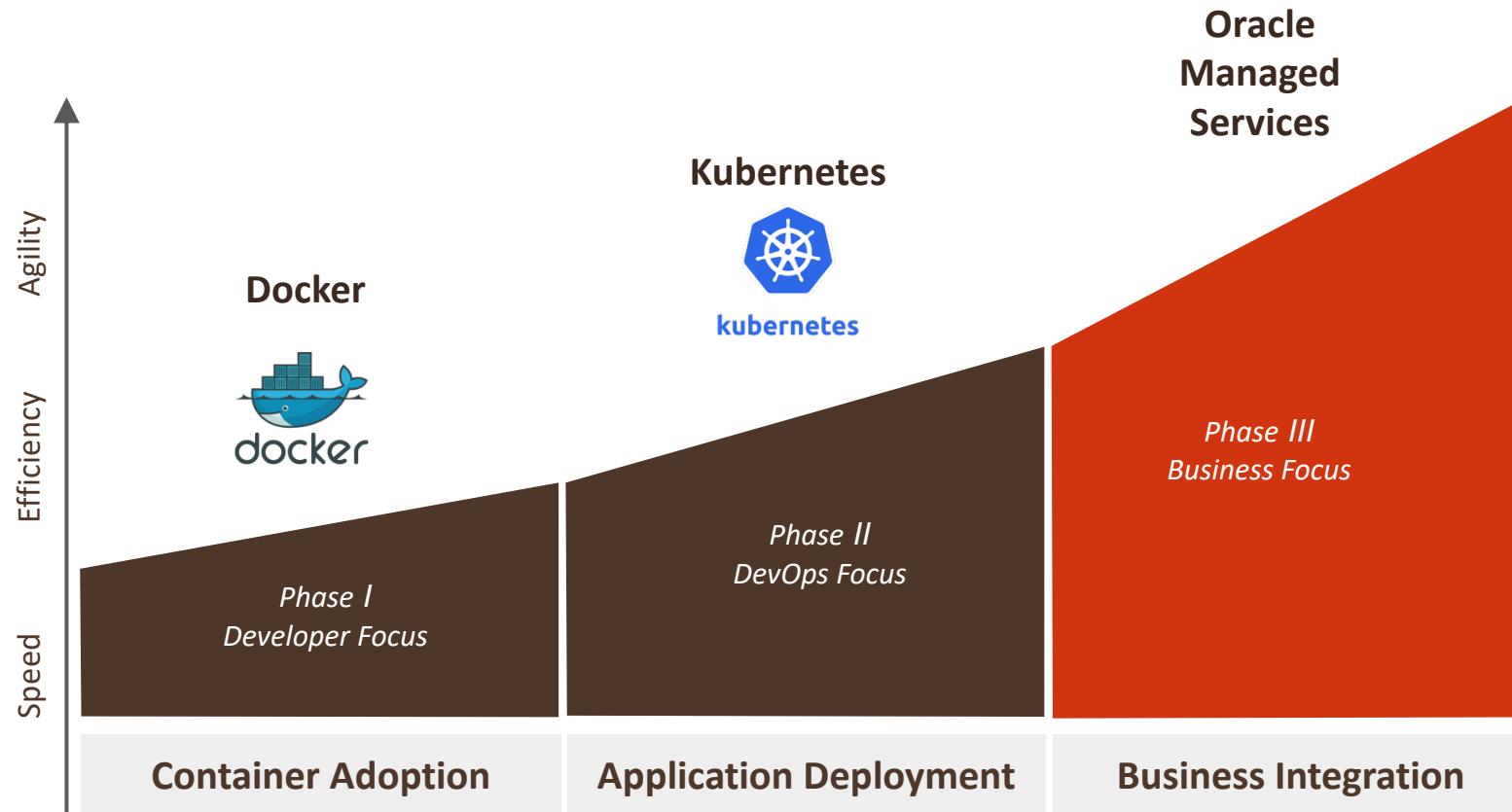
## What is It?

- A high availability Docker v2 container registry service
- Stores Docker Images in Private Repositories
- Runs as a fully managed service on Oracle Cloud Infrastructure

## What Problems Does it Solve?

- Without a registry it is hard for Development teams to maintain a consistent set of Docker images for their containerized applications
- Without a registry it is hard to enforce access rights and security policies for images
- It is too hard to find the right images and have them available in the region of deployment
- Full integration with Container Engine for Kubernetes (OKE)
- Registries are private by default, but can be made public by an admin
- Co-located regionally with Container Engine for low latency Docker image deploys
- Leverages OCI for high performance, low latency and high availability

## Key Benefits



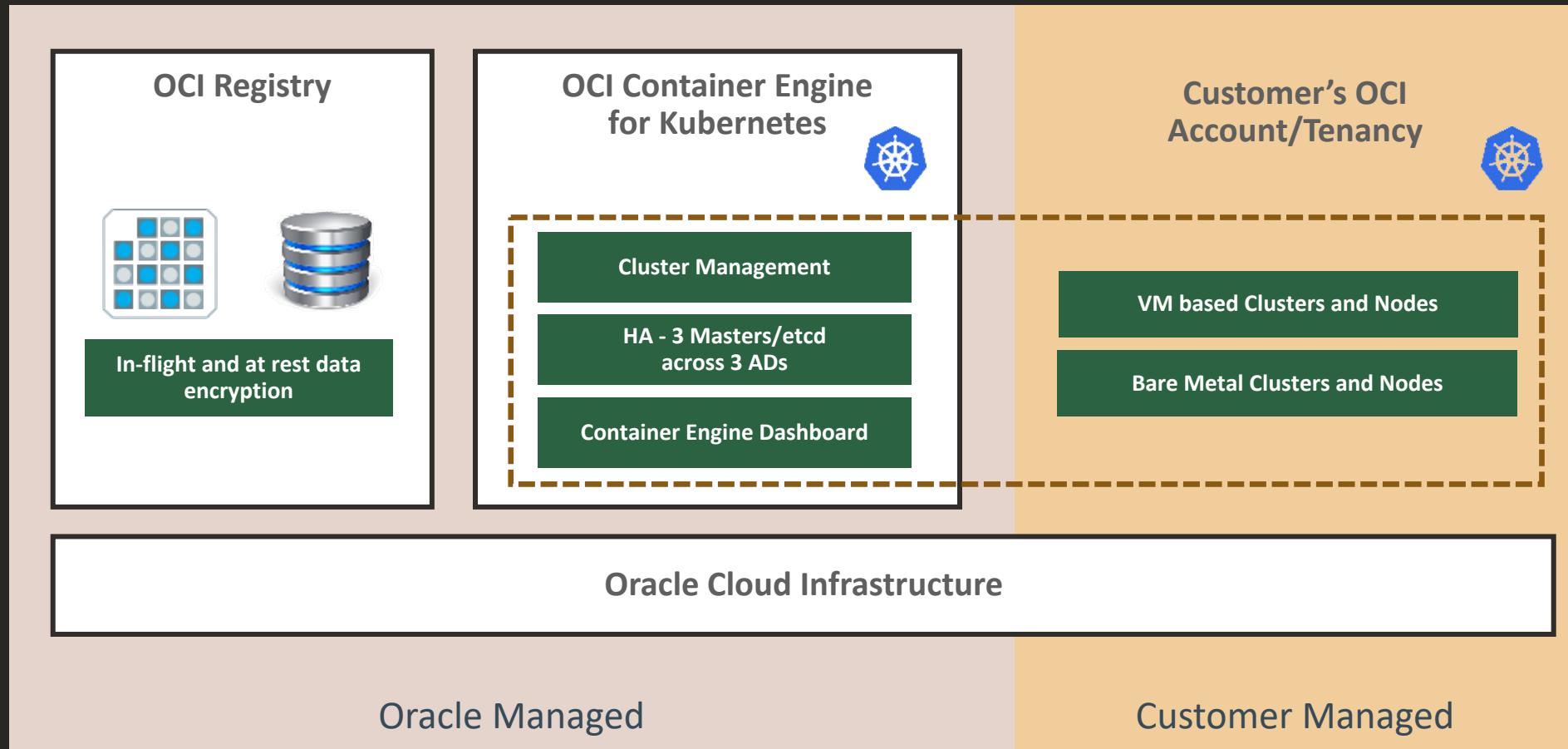
**Focus:**  
**Applications:**  
**Automation:**  
**Community:**

Developer adoption  
 Dev/Test apps  
 Simple orchestration  
 Individual developers

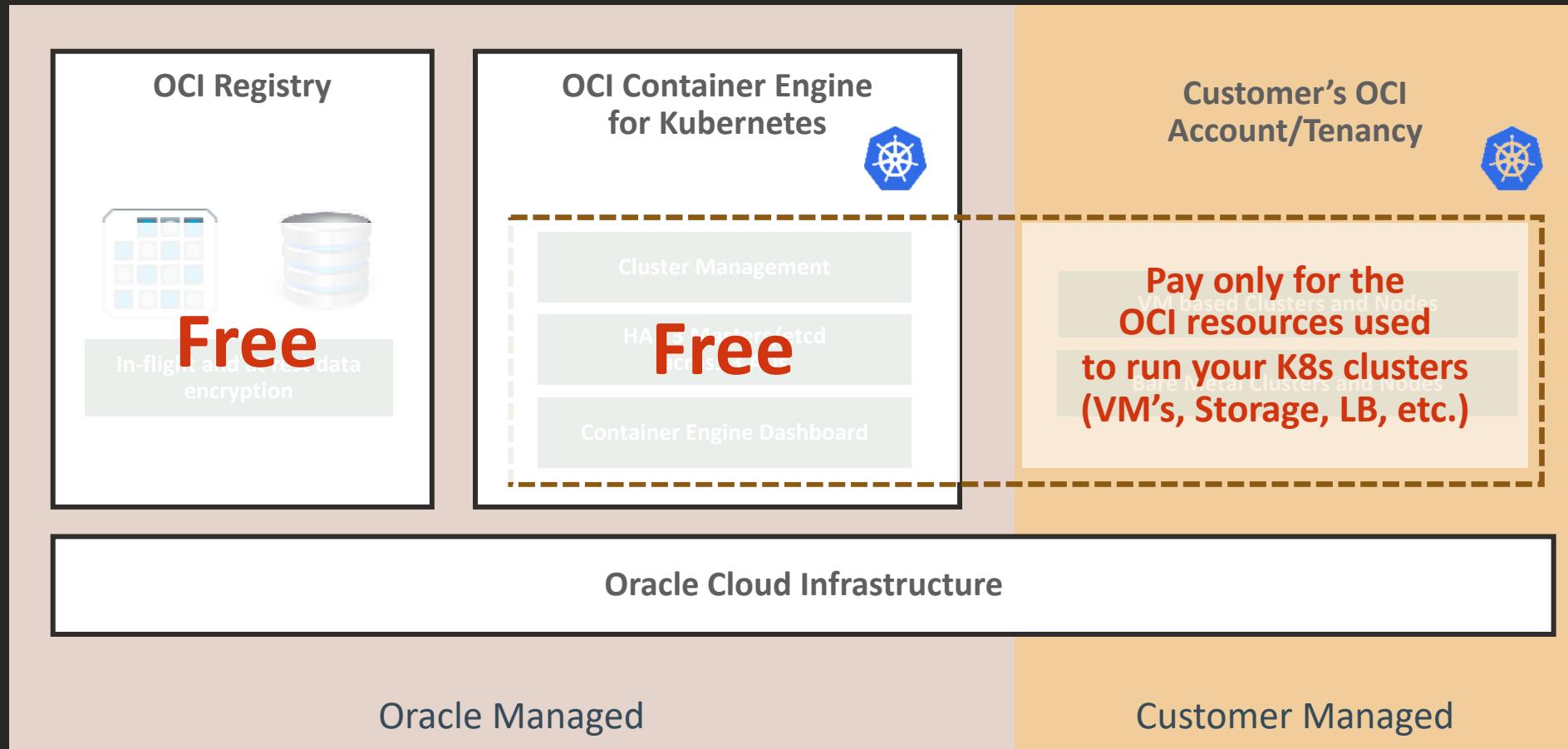
DevOps deployment  
 Production apps  
 Advanced orchestration  
 Teams & lines of business

Broad integration  
 Business apps  
 Self management  
 Enterprises

# Working with OKE and OCIR on OCI



# OKE/OCIR Pricing and Packaging



# Summary: Container Engine & Registry



## Container Native

- **Standard Docker & Kubernetes**
  - Deploy standard & open upstream Docker and Kubernetes versions for compatibility across environments
- **Registry Integration**
  - Full Docker v2 compatible private registry to store and manage images
- **Container Engine**
  - Deploy and operate containers and clusters
- **Full integration to cloud networking and storage**
  - Leverage the enterprise class networking, load balancing and persistent storage of Oracle Cloud Infrastructure

## Developer Friendly

- **Streamlined Workflow**
  - Use your favorite CI to push containers to the registry, then Kubernetes to deploy to clusters and manage operations
- **Full REST API**
  - Automate the workflow, create and scale clusters through full REST API
- **Built In Cluster Add-Ons**
  - Kubernetes Dashboard, DNS & Helm
- **Open Standards**
  - Docker Based Runtime
  - Worker Node SSH Access
  - Standard Kubernetes

## Enterprise Ready

- **Simplified Cluster Operations**
  - Use the fully managed, highly available registry, master nodes and control plane
- **Full Bare Metal Performance and Highly Available IaaS**
  - Combine Kubernetes with bare metal shapes for raw performance
  - Deploy Kubernetes clusters across multiple Availability Domains for resilient applications
- **Team Based Access Controls**
  - Control team access and permissions to clusters
- **Autonomous Clusters**
  - Maintain cluster size and performance in face of node failures and load fluctuations

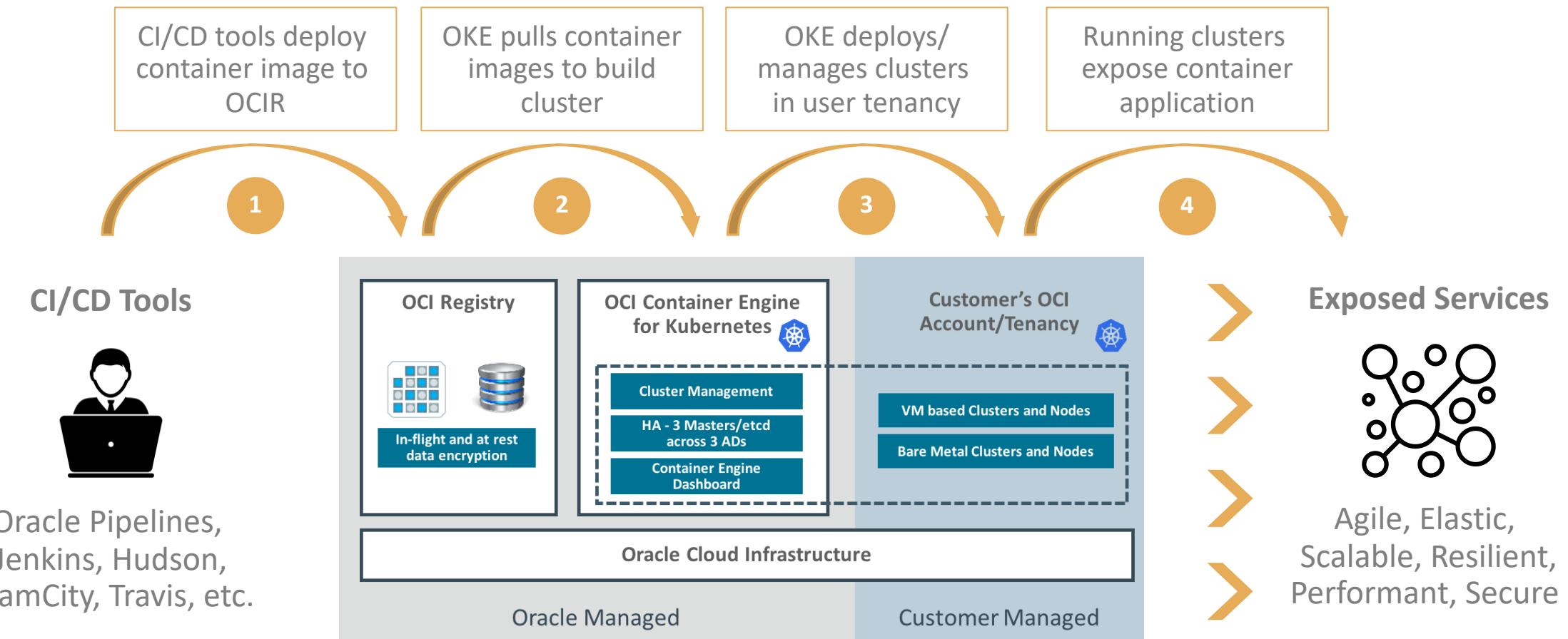
# OKE - Labs



**“Quanto mais tempo demorar para solucionar um problema, mais simples (e idiota) é a solução”**

Cristiano Hoshikawa

# Container Application Life Cycle



## Hands-on

---

<https://github.com/hoshikawa2/OCI-DEV>

# Dashboard Kubernetes

---

## kubectl

O que é: Linha de comando para deployments e gestão dos componentes em kubernetes  
É com esta linha de comando que vamos instalar os componentes do workshop hoje!

## Dashboard Kubernetes

---

Versão nova de Kubernetes necessita trazer o token:

```
TOKEN=$(kubectl -n kube-system describe secret default | awk '$1=="token:{print $2}')
```

```
kubectl config set-credentials kubernetes-admin --token="${TOKEN}"
```

# kubectl proxy

<http://localhost:8001/api/v1/namespaces/kube-system/services/https:kubernetes-dashboard:/proxy/#!/login>

The image shows two side-by-side screenshots of the Kubernetes Dashboard. On the left is the 'Kubernetes Dashboard' login screen, and on the right is the 'Overview' page of the dashboard.

**Kubernetes Dashboard (Left):**

- Kubeconfig:** A section instructing users to select a kubeconfig file to configure access to the cluster. It links to the 'Configure Access to Multiple Clusters' section.
- Token:** A section explaining that every Service Account has a Secret with a valid Bearer Token for logging in. It links to the 'Authentication' section.
- Choose kubeconfig file:** A text input field for selecting a kubeconfig file.
- Kubeconfig:** A button to upload a kubeconfig file.
- SIGN IN:** A blue button to log in.

**Overview (Right):**

- Cluster:** A sidebar menu with options: Namespaces, Nodes, Persistent Volumes, Roles, Storage Classes, and a dropdown for the namespace (set to 'default').
- Overview:** The main content area displays workloads status and deployment details.
- Workloads Statuses:** Three green circles showing 100.00% for Deployments, Pods, and Replica Sets.
- Deployments:** A table listing seven deployments with their names, labels, pods, age, and images.

Name	Labels	Pods	Age	Images
callpdfreport-deployment	app: callpdfreport	1 / 1	14 hours	iad.ocir.io/davixsf5sbx/cristian...
runhtml-deployment	app: runhtml	1 / 1	2 days	iad.ocir.io/davixsf5sbx/cristian...
jenkins-deployment	app: jenkins	1 / 1	3 days	jenkins/jenkins-lts
postgres	app: postgres	1 / 1	12 days	postgres:10.4
nginx-ingress-controller	k8s-app: nginx-ingress-controller	1 / 1	12 days	gcr.io/google_containers/nginx-...
default-http-backend	k8s-app: default-http-backend	1 / 1	12 days	gcr.io/google_containers/default...
sonarcube-latest				

# YAML

---

According to [www.yaml.org](http://www.yaml.org), “YAML is a **human-friendly**, data serialization standard for all programming languages.”

XML	JSON	YAML
<pre>&lt;Servers&gt;   &lt;Server&gt;     &lt;name&gt;Server1&lt;/name&gt;     &lt;owner&gt;John&lt;/owner&gt;     &lt;created&gt;123456&lt;/created&gt;     &lt;status&gt;active&lt;/status&gt;   &lt;/Server&gt; &lt;/Servers&gt;</pre>	<pre>{   Servers: [     {       name: Server1,       owner: John,       created: 123456,       status: active     }   ] }</pre>	<pre>Servers:   -   name: Server1       owner: John       created: 123456       status: active</pre>

# YAML - Deployment

---

Deployment é o componente de software em si

Você define quantas replicas para definir quantos pods precisam entrar em funcionamento para performance

Image define uma imagem de container já previamente criada em um repositório

ContainerPort define a porta que este pod está acessível

VolumeMounts e PersistentVolumeClaim definem a localização física do seu container

```
apiVersion: extensions/v1beta1
kind: Deployment
metadata:
  name: jenkins-deployment
spec:
  replicas: 1
  selector:
    matchLabels:
      app: jenkins
  template:
    metadata:
      labels:
        app: jenkins
    spec:
      containers:
      - name: jenkins
        image: jenkins/jenkins:lts
        securityContext:
          runAsUser: 0
        ports:
        - containerPort: 8080
      volumeMounts:
      - name: jenkins-home
        mountPath: /var/jenkins_home
volumes:
- name: jenkins-home
  persistentVolumeClaim:
    claimName: jenkins-pv-claim
```

## YAML - Service

---

O serviço é a porta de acesso do mundo para o pod

Você consegue definir se o acesso é feito via:

- Load Balancer (mundo externo através de um IP externo)
- Cluster IP (acesso apenas via o cluster de kubernetes)
- NodePort (acesso via as VMs e também via cluster kubernetes)

```
apiVersion: v1
kind: Service
metadata:
  name: jenkins-service
  labels:
    app: jenkins
spec:
  selector:
    app: jenkins
  ports:
    - port: 8080
      targetPort: 8080
  type: ClusterIP
```

## YAML - Storage

---

É necessário definir muitas vezes um storage para persistência de dados, pois caso contrário, seu pod será stateless

A forma como é feita em OCI precisa que storageClassName seja “oci”

Na definição de Deployment, para utilizar o storage definido aqui, é necessário fazer a seleção pelo label

```
kind: PersistentVolume
apiVersion: v1
metadata:
  name: jenkins-pv-volume
  labels:
    type: local
    app: jenkins
spec:
  storageClassName: "oci"
  capacity:
    storage: 10Gi
  accessModes: [ "ReadWriteOnce" ]
  persistentVolumeReclaimPolicy: Retain
  hostPath:
    path: "/mnt/data"
---
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: jenkins-pv-claim
  labels:
    app: jenkins
spec:
  storageClassName: "oci"
  accessModes: [ "ReadWriteOnce" ]
  resources:
    requests:
      storage: 10Gi
```

## YAML - Ingress

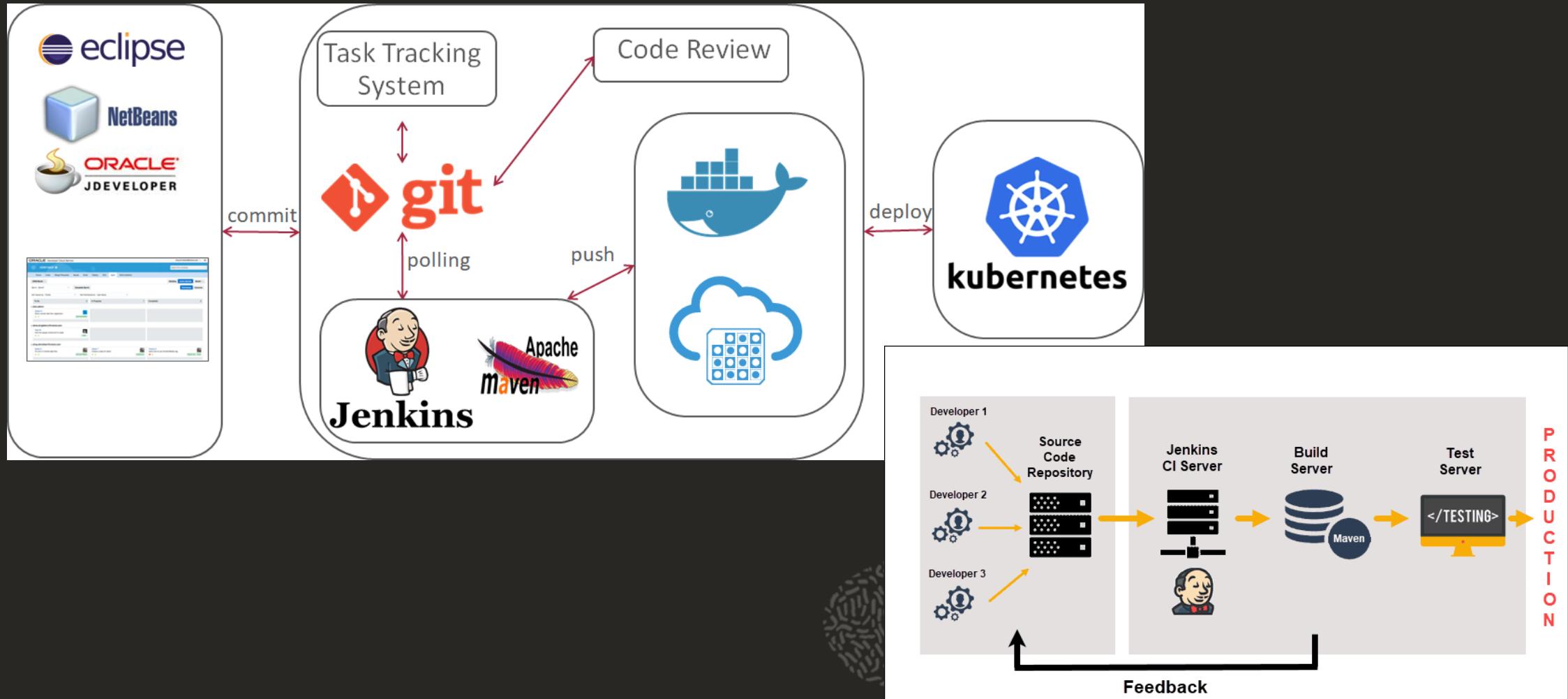
---

Ingress permite ter controle de direcionamento para os serviços desejados

Ao invés de usar LoadBalancer e configurar tudo manualmente

```
apiVersion: extensions/v1beta1
kind: Ingress
metadata:
  name: jenkins-ingress
  annotations:
    kubernetes.io/ingress.class: "nginx"
spec:
  tls:
  - secretName: tls-secret
  rules:
  - http:
      paths:
      - backend:
          serviceName: jenkins-service
          servicePort: 8080
```

## Lab - Jenkins



## Lab – Install jenkins on OKE

---

<https://github.com/hoshikawa2/oci-oke-jenkins-kubernetes.git>

Executar nesta ordem:

```
kubectl create -f jenkins-storage.yaml  
kubectl create -f jenkins-deployment.yaml  
kubectl create -f jenkins.yaml
```

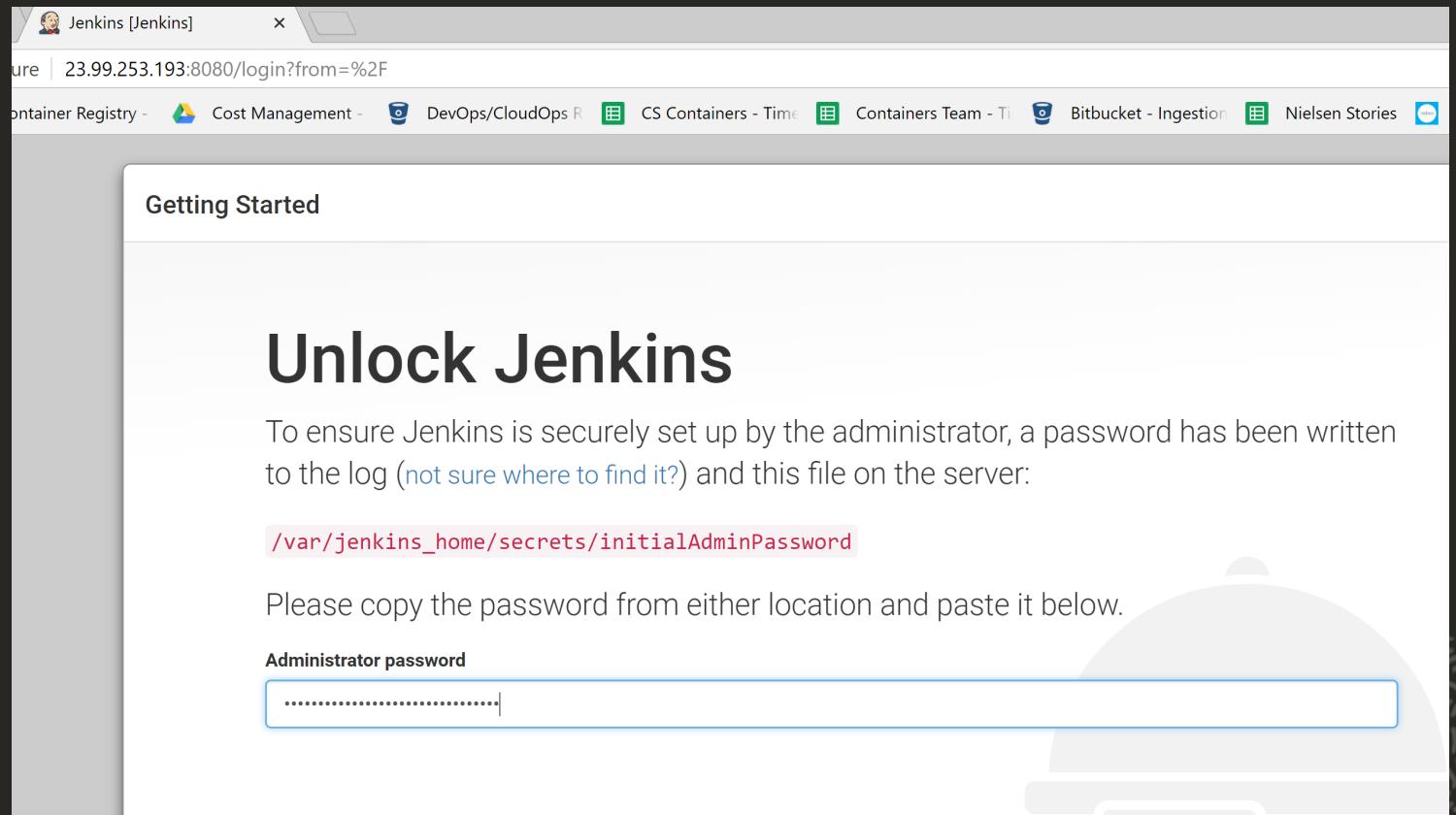
Obs: Não vamos executar o jenkins-ingress.yaml, deixaremos para um próximo evento

```
kubectl -n logging port-forward $(kubectl -n logging get pod -l app=jenkins -o jsonpath='{.items[0].metadata.name}') 8080:8080
```

Abrir no browser: <http://localhost:8080>

# Lab – Using Jenkins

Necessita incluir a senha inicial



# Lab – Using Jenkins

Localizar o pod do Jenkins no dashboard  
Kubernetes

Clicar no POD

The screenshot shows the Kubernetes Dashboard interface in a Safari browser. The top navigation bar includes links for Oracle, XCode, Mac, Música, Viagens, Corridas, Filmes, Conhecimentos, Fotografia, YouTube, Apple, WhatsApp Web, Oracle Cloud, Correios, MercadoLivre Brasil, UOL, and e-bay. The main title is "localhost" and the sub-page title is "Install and Set Up kubectl - Kubernetes". The left sidebar has sections for Cluster, Namespaces, Nodes, Persistent Volumes, Roles, Storage Classes, and Namespace (set to default). The "Overview" section is currently selected. The right panel has a "Pods" section with a table:

Name	Node	Status	Restarts	Age	Actions
callpdfreport-deployment-7c9f59c85f-tcz6v	10.0.10.2	Waiting: ImagePullBackOff	0	14 hours	⋮ ⋮
runhtml-deployment-65f94fff6b-l5xjb	10.0.10.2	Running	0	a day	⋮ ⋮
<b>jenkins-deployment-877b8c7c4-thhsz</b>	10.0.10.2	Running	2	2 days	⋮ ⋮
sonarqube-58b8f5db5f-lvk8t	10.0.10.2	Running	2	2 days	⋮ ⋮
kafka-broker0-7d9dc9c874-gmcdt	10.0.10.2	Running	3	2 days	⋮ ⋮
nginx-ingress-controller-5f9cd47546-xgdh5	10.0.10.2	Running	6	2 days	⋮ ⋮
default-http-backend-6c66f74d79-ntmls	10.0.10.2	Running	6	3 days	⋮ ⋮
zookeeper-deployment-1-684478678c-s69tg	10.0.10.2	Running	2	3 days	⋮ ⋮
grafana-768d6869c6-lzr8k	10.0.10.2	Running	7	3 days	⋮ ⋮
postgres-5dbcc6764c-vp68r	10.0.10.2	Running	2	3 days	⋮ ⋮

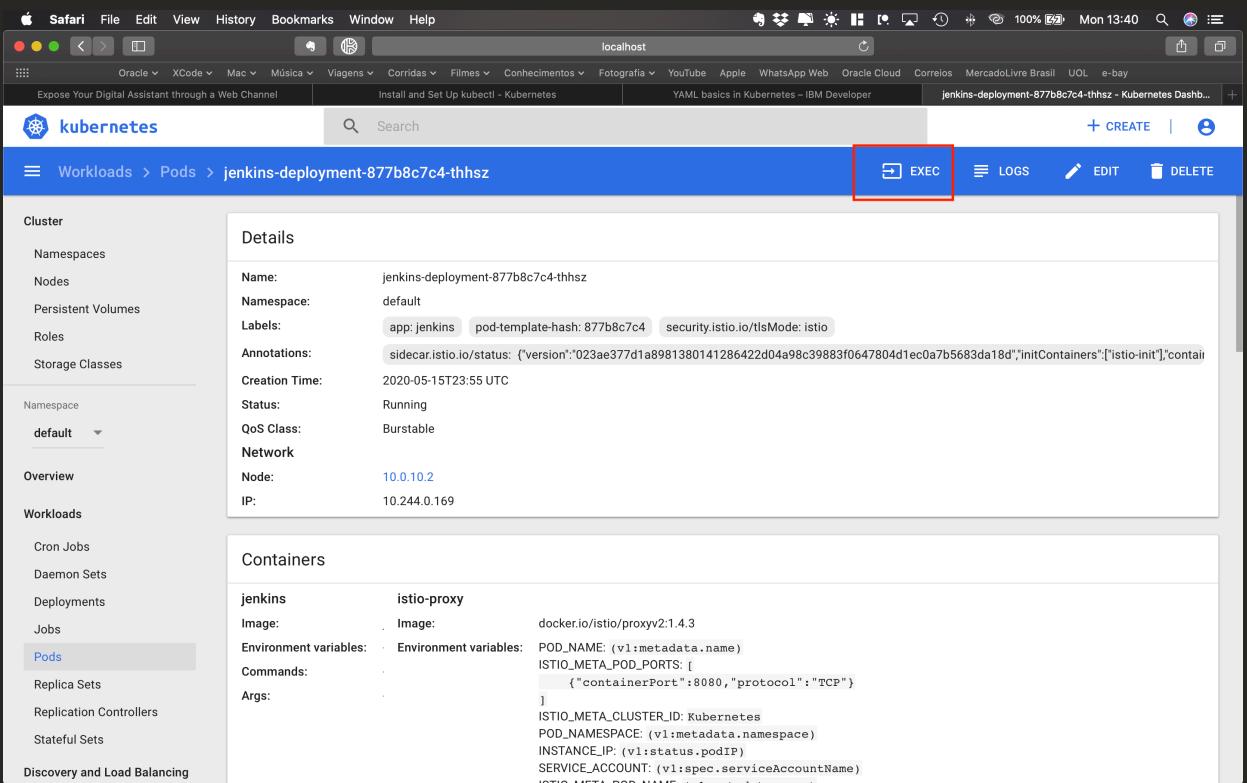
At the bottom of the table, there is a link: "Open #!/pod/default/jenkins-deployment-877b8c7c4-thhsz?namespace=default on this page in a new tab".

# Lab – Using Jenkins

Clicar em EXEC

E no terminal digitar

cat /var/jenkins\_home/secrets/initialAdminPassword



The screenshot shows a web browser window with the address bar set to 'localhost'. The main content is a Kubernetes pod details page for 'jenkins-deployment-877b8c7c4-thhsz'. The left sidebar shows 'Namespaces', 'Nodes', 'Persistent Volumes', 'Roles', and 'Storage Classes'. Under 'Namespaces', 'default' is selected. The 'Overview' section includes links for 'Cron Jobs', 'Daemon Sets', 'Deployments', 'Jobs', 'Pods' (which is selected), 'Replica Sets', 'Replication Controllers', 'Stateful Sets', and 'Discovery and Load Balancing'. The 'Details' section on the right lists the pod's metadata: Name: jenkins-deployment-877b8c7c4-thhsz, Namespace: default, Labels: app:jenkins, pod-template-hash:877b8c7c4, security.istio.io/tlsMode:istio, Annotations: sidecar.istio.io/status: {"version":"023ae377d1a8981380141286422d04a98c39883f0647804d1ec0a7b5683da18d", "initContainers": ["istio-init"]}, and creation time: 2020-05-15T23:55 UTC. It also shows the pod is running on node 10.0.10.2 with IP 10.244.0.169. The 'Containers' section shows one container named 'jenkins' using image 'istio/proxy' from 'docker.io/istio/proxy:2.1.4.3'. Environment variables include POD\_NAME, ISTIO\_META\_POD\_PORTS (with a value of [{"containerPort": 8080, "protocol": "TCP"}]), and ISTIO\_META\_CLUSTER\_ID, POD\_NAMESPACE, INSTANCE\_IP, SERVICE\_ACCOUNT, and ISTIO\_META\_POD\_NAME. The 'Logs' tab is also visible at the bottom of the interface.

## Lab – Using Jenkins

---

Copiar a senha e colar no painel Jenkins

E isso vira assunto para um próximo encontro...

**ORACLE®**  
Cloud Platform



Oracle Cloud Platform  
Transform Your Business With Innovation

A person is visible from the chest up, wearing a yellow and black horizontally striped shirt. The background is a plain, light gray.

**ORACLE**