



Advance Docker with Kubernetes (101)



ເວັງ ລົສໜຶ່ງ
ໄມ້ໂລນ ກົດ້າໄດ້

By Praparn Luengphoonlap
Email: eva10409@gmail.com

Kubernetes 101



Agenda

- What's new on container world
- Container concept (Recap) and Use Case
- Principle of Kubernetes
- K8S vs Docker (Swarm) what is the difference ?
- How to acquire K8S on your machine ?
- Demo Session
 - Enable K8S feature and deploy dashboard for K8S
 - Demo: Deploy simple application with K8S and Docker Native
 - Demo: Deploy compose with K8S and Swarm



Senior Infrastructure / System Engineer

Compute, Network & Storage System (CNS)

Infrastructure Enterprise Equipment

Virtual Machine for Open System in Prd

Core Application on IBM AS/400 System

Instructor / Consult for Docker's Solution

The background of the slide has a green wood-grain texture.

← → ⌂ Twitter, Inc. [US] <https://twitter.com/solomonstre>

Apps NMac Ked - Mac OS... Medium jenkins vagrant Mesos Vue docker NGINX Taiwan Other Bookmarks

หน้าแรก เกี่ยวกับ ค้นหาทวิตเตอร์ มีบัญชีผู้ใช้อื่นๆแล้ว? เช้าสู่ระบบ ▾

ทวีต 7,496 ก้าวสั้นติดตาม 7,514 ผู้ติดตาม 37.2K ความชอบ 2,645 รายชื่อ 1 [ติดตาม](#)

Solomon Hykes
@solomonstre

Hacker & entrepreneur. Co-founder of Docker, formerly known as Dotcloud.
docker.io github.com/shykes

docker.io
 เข้าร่วมเมื่อ ตุลาคม 2550
 195 รูปภาพหรือวิดีโอ

ทวีต ทวีตและการตอบกลับ สื่อ

ทวีตป้าหมุด
Solomon Hykes @solomonstre · 18 ชั่วโมง
Hi everyone, I've got some news to share with you today...
blog.docker.com/2018/03/au-rev...

Today I'm announcing my departure from Docker, the company I helped create ten years ago and have been building ever since. A founder's departure is usually seen as a dramatic event. Sadly, I must report that reality is far less exciting in this case. I've had many roles at Docker over the years, and today I have a new, final one – as an active board member, a major shareholder and, I expect, a high maintenance Docker user. But I will no longer be part of day-to-day operations. Instead, after obsessing for so many years over my own ideas, I am rediscovering the joys of putting myself at the service of others – my friends, my family, and the brilliant entrepreneurs I've been lucky enough to advise and invest in over the years. Over the coming months I plan to use my experience to help them in any way I can.

This transition is simply another chapter in a long story of change, growth, hard work... and a lot of luck.

Ten years ago, I quit my job, returned to live with my mother in Paris and, together with my friends Kamel Founadi and Sebastien Pahl, started a company called Dotcloud. Our goal was to harness an obscure technology called containers, and use it to create what we called "tools of mass innovation": programming tools which anyone could use. I was 24 and had no idea what I was doing. We needed a CEO, so that became my new role.

Five years ago, Dotcloud reinvented itself as Docker, around a battle-hardened core of five people: Eric Bardin, Sam Alba, Jerome Petazzoni, Julien Barbier and myself. Soon growth was off the charts, and we hired an experienced CEO to help us sustain it. I was 29 and eager to do my part. Docker needed a CTO, so that became my new role.

Today, as I turn 34, Docker has quietly transformed into an enterprise business with explosive revenue growth and a developer community in the millions, under the leadership of our CEO, the legendary Steve Singh. Our strategy is simple: every large enterprise in the world is preparing to migrate their applications and infrastructure to the cloud, en masse. They need a solution to do so reliably and securely, without expensive code or process changes, and without locking themselves to a single operating system or cloud. Today the only solution meeting these requirements is Docker Enterprise Edition. This puts Docker at the center of a massive growth opportunity. To take advantage of this opportunity, we need a CTO by Steve's side with decades of experience shipping and supporting software for the largest corporations in the world. So I now have a new role: to help find that ideal CTO, provide the occasional bit of advice, and get out of the team's way as they continue to build a juggernaut of a business. As a shareholder, I couldn't be happier to accept this role.

As a founder, of course, I have mixed emotions. When you create a company, your job is to make sure it can one day succeed without you. Then eventually that one day comes and the celebration can be bittersweet.

It's never easy for a founder to part ways with their life's work. But I realize how incredibly lucky I am to have this problem. Most ideas never materialize. Most software goes unused. Most businesses fail in their first year. Yet here we are, one of the largest open-source communities ever assembled, collectively building software that will run on millions of computers around the world. To know that your work was meaningful, and that a vibrant community of people will continue building upon it... can any founder ask for anything more?

I want to thank from the bottom of my heart every member of the Docker team and community, past and present, for making Docker what it is today. Thanks to you, this founder's bittersweet moment is mostly sweet. We have built something great together. I look forward to seeing where you will take it next.

Happy hacking,

Solomon

What's new on container world

Cloud Native Landscape

v2.1

See the interactive landscape at landscape.cncf.io

Greyed logos are not open source

App Definition & Development



CNCF Incubating



Orchestration & Management



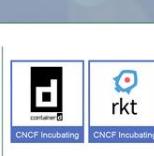
CNCF Graduated



Runtime



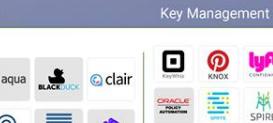
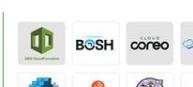
CNCF Sandbox



Provisioning



CNCF Sandbox



Cloud



github.com/cncf/landscape

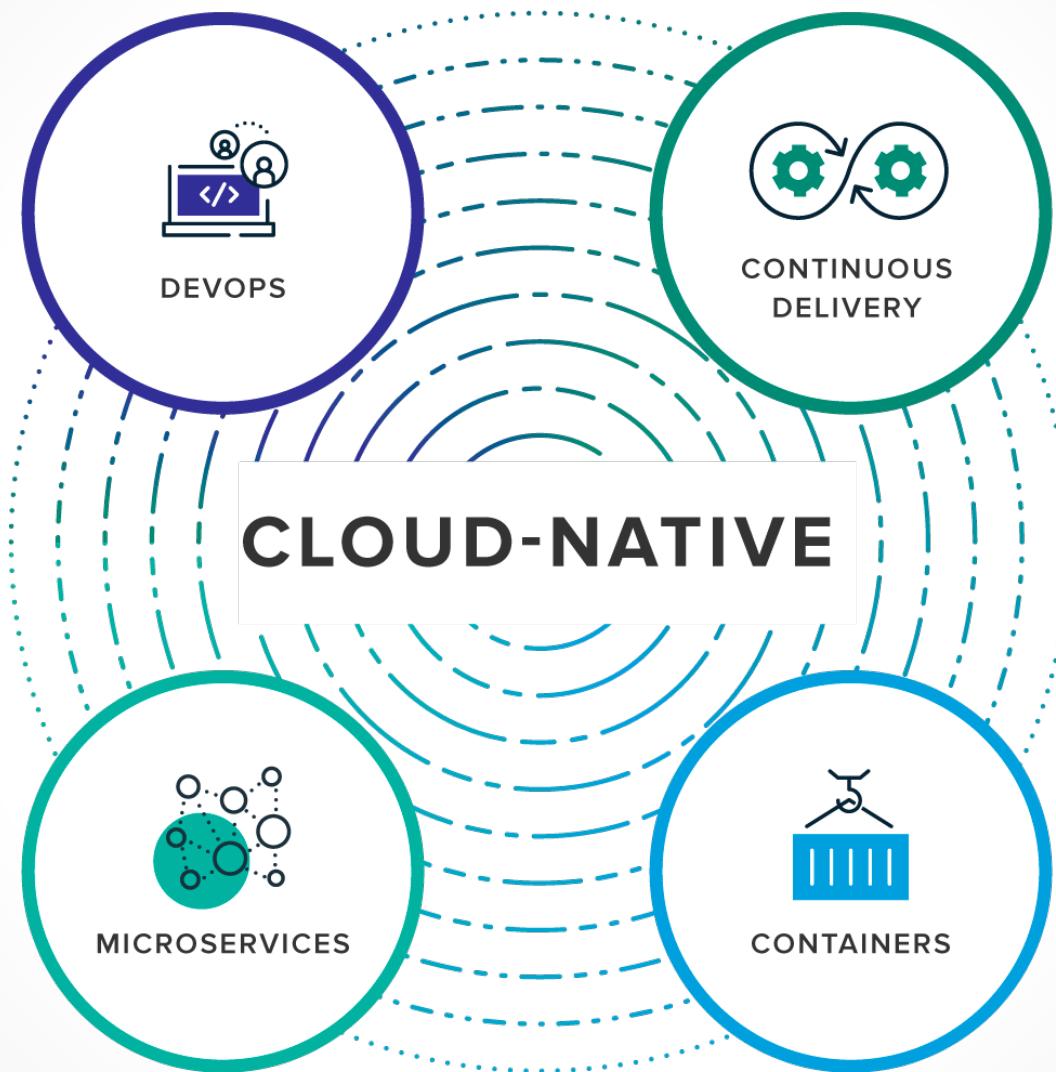
This landscape is intended as a map through the previously uncharted terrain of cloud native technologies. There are many routes to deploying a cloud native application, with CNCF Projects representing a particularly well-traveled path.



Kubernetes Certified Service Provider



What's new on container world



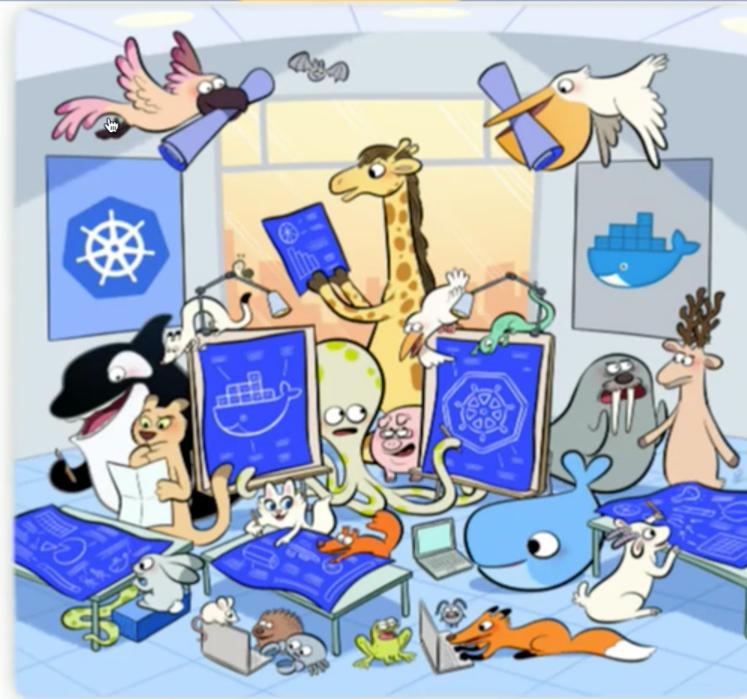
What's new on container world



Kubernetes 101

What's new on container world

WE ARE
ONE BIG
COMMUNITY



docker
con 17 EU

What's new on container world

Docker with Swarm and Kubernetes

1 →

The best enterprise
container security and
management

← 2

The best container
development workflow

3 →

Native Kubernetes
integration provides full
ecosystem
compatibility

← 4

Industry-standard
container runtime

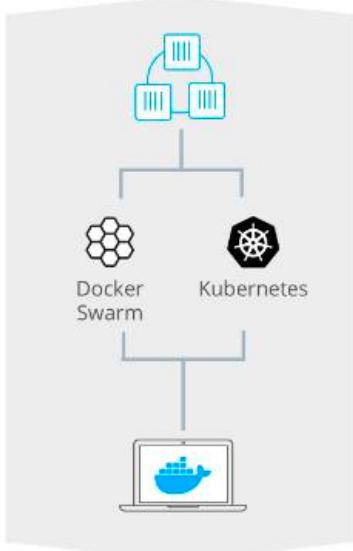


dockercon¹⁷EU

What's new on container world

Docker for Mac Docker for Windows*

Build Test Ship

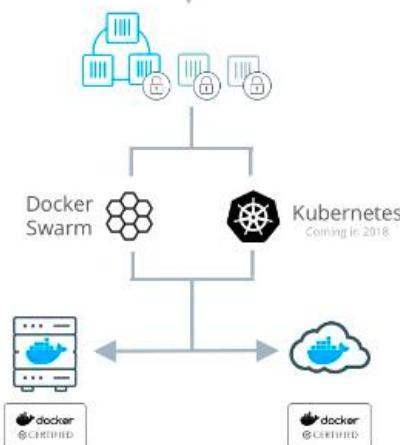


Docker Enterprise Edition

Secure Deploy Manage



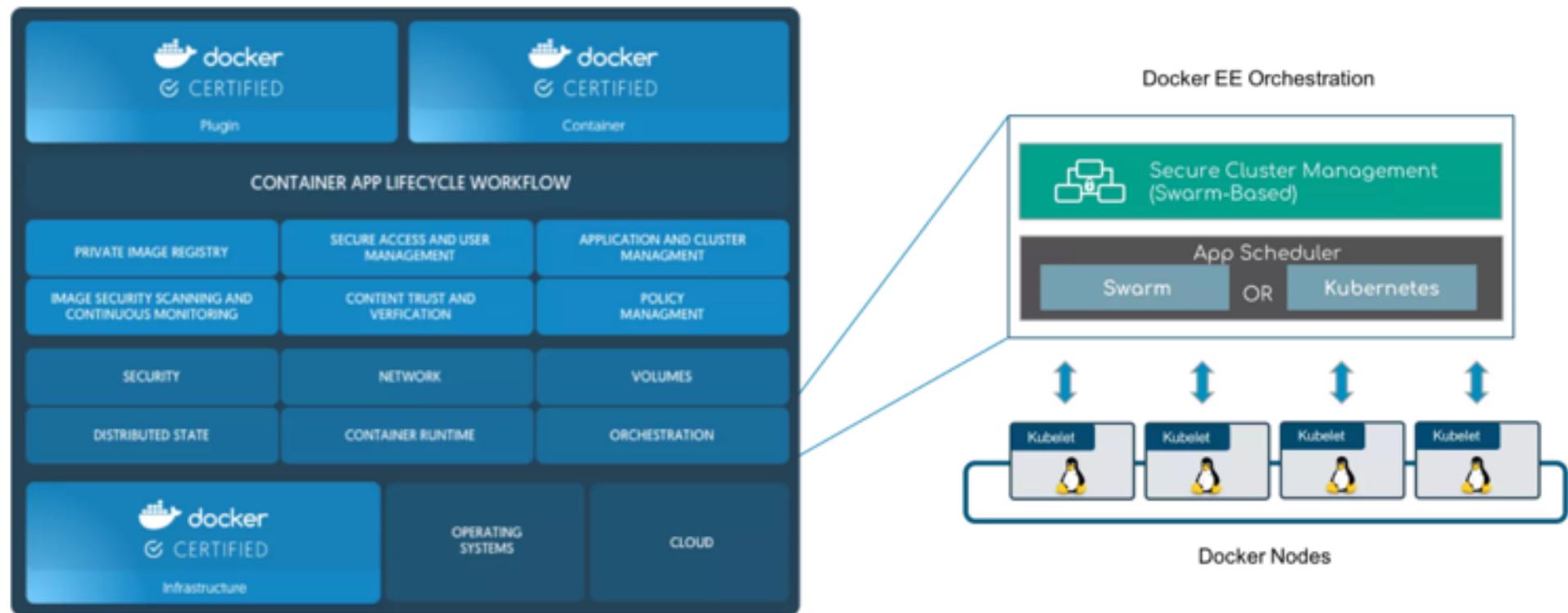
Docker Universal Control Plane (UCP)



*Kubernetes support in Docker for Windows will be available at a later date.

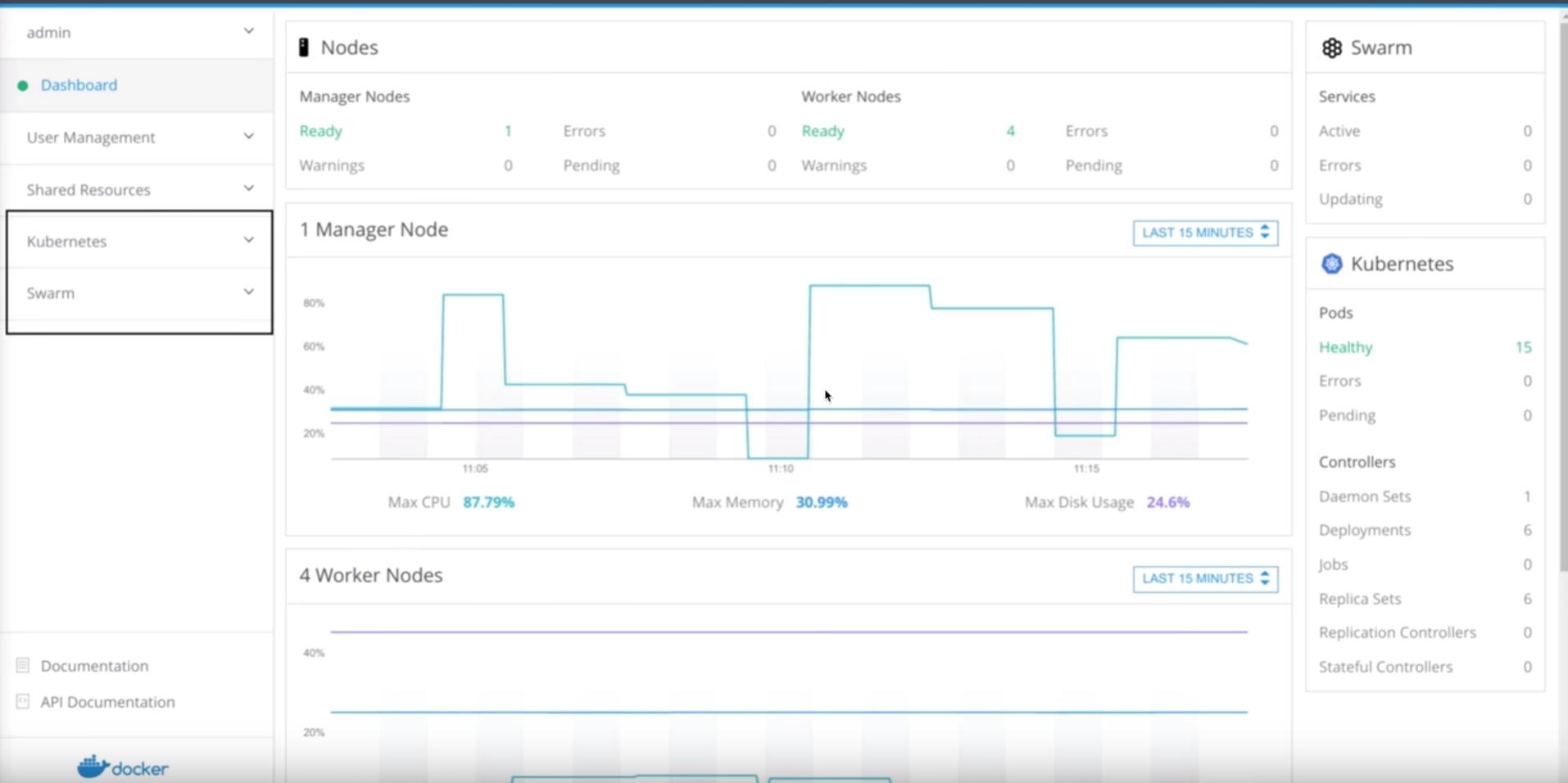
What's new on container world

- Platform of docker to support K8S
- Docker Enterprise Edition



What's new on container world

- Docker Enterprise Edition (EE)



What's new on container world

- Docker Enterprise Edition (EE)

The image shows two interface screenshots and a terminal window. The top left is the Docker Swarm interface with sections for Services (0), Networks (14), Volumes (5), and Secrets (0). The top right is the Kubernetes interface with sections for Create (+), Controllers, Load Balancers, Pods, Configurations, and Storage. The bottom part is a terminal window with two tabs. Tab 1 shows the command `docker node ls` output:

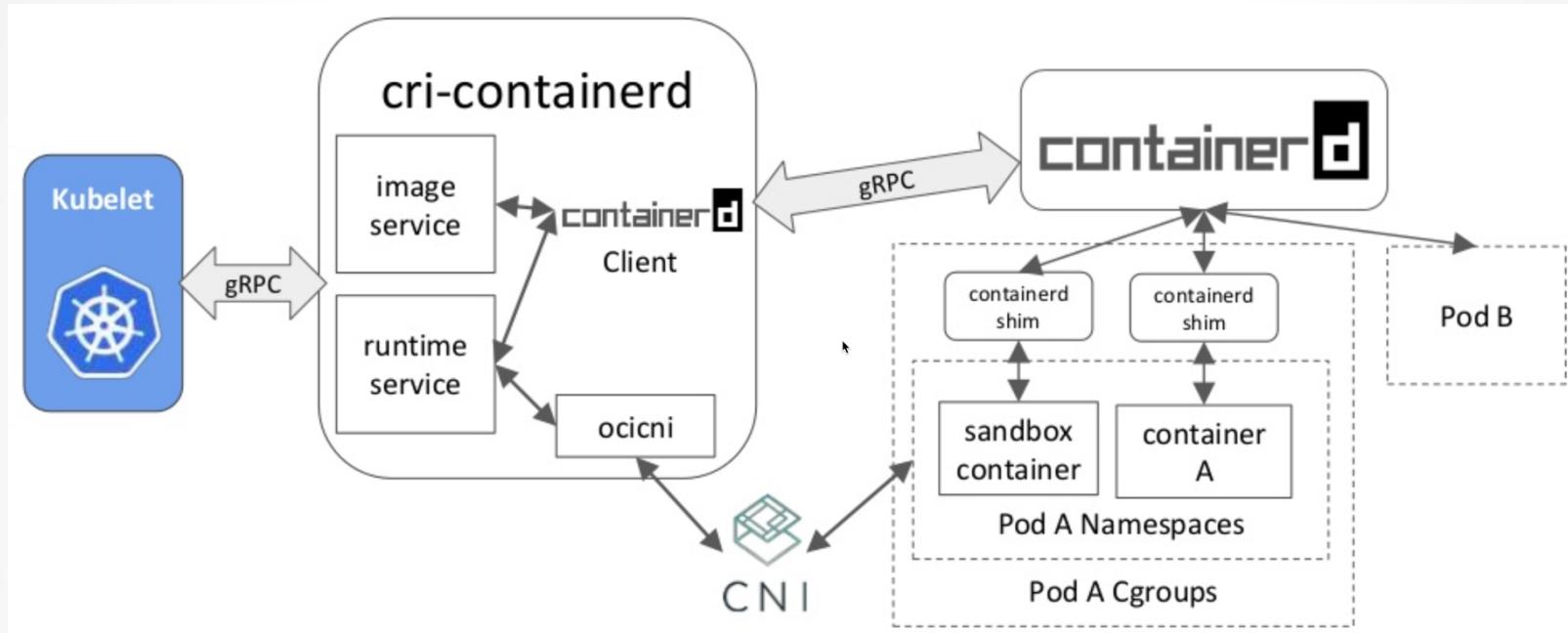
ID	HOSTNAME	STATUS	AVAILABILITY	MANAGER STATUS
6q3c3s9jw9vq3hfd68a60npf1	dockercon1-ubuntu-3	Ready	Active	
kb17pilk4ey3qdsim41finjer	dockercon1-ubuntu-4	Ready	Active	
uvcf5da05vv34jktdqovpsmxw	dockercon1-ubuntu-5	Ready	Active	
uvrhlku71k23081d6lfhrhqja	dockercon1-ubuntu-1	Ready	Active	Reachable
xjpjsdrc88qew4kkvtpwrvooff *	dockercon1-ubuntu-0	Ready	Active	Leader
xqjltu9980fhagsystl0in1l2	dockercon1-ubuntu-2	Ready	Active	Reachable

Tab 2 shows the command `kubectl get nodes` output:

NAME	STATUS	AGE	VERSION
6q3c3s9jw9vq3hfd68a60npf1	Ready	1d	v1.7.6
kb17pilk4ey3qdsim41finjer	Ready	1d	v1.7.6
uvcf5da05vv34jktdqovpsmxw	Ready	1d	v1.7.6
uvrhlku71k23081d6lfhrhqja	Ready	1d	v1.7.6
xjpjsdrc88qew4kkvtpwrvooff	Ready	1d	v1.7.6
xqjltu9980fhagsystl0in1l2	Ready	1d	v1.7.6

What's new on container world

- Moby Project



kubernetes

What's new on container world

Kubernetes Support in Docker Enterprise Edition

Vivek Saraswat, Product Manager



<https://blog.docker.com/2018/01/docker-ee-kubernetes/>

Kubernetes: Production Workload Orchestration



What's new on container world

Slant ▾



Search or Ask a Question

Log In or Join Now

ASK QUESTION

Development

Backend Development

Sysadmin

Follow

...

What are the best Docker orchestration tools?

11

OPTIONS CONSIDERED

77

RECOMMENDATIONS

Mar 13, 2018

LAST UPDATED

THE BEST 1 OF 11 OPTIONS WHY?

11 Options Considered	Price	Last Updated
 THE BEST Kubernetes	 GET IT HERE	Mar 13, 2018
 Docker Swarm	 GET IT HERE	Dec 11, 2017
 Docker Compose	 GET IT HERE	Sep 23, 2017
 Nomad	 GET IT HERE	Sep 23, 2017
 OpenShift	 GET IT HERE	Jan 22, 2018

 See Full List

RELATED QUESTIONS

[What are the best power user tools for macOS?](#)

[What are the best continuous integration tools?](#)

[What are the best developer tools for Mac OSX?](#)

[What are the best software tools for live streaming?](#)

[What are the best log aggregation & monitoring tools?](#)

[What are the best mockup and wireframing tools for websites?](#)

[What are the best diff tools for Git?](#)

[What are the best mind mapping tools?](#)

[What are the best 2D animation tools for game development?](#)

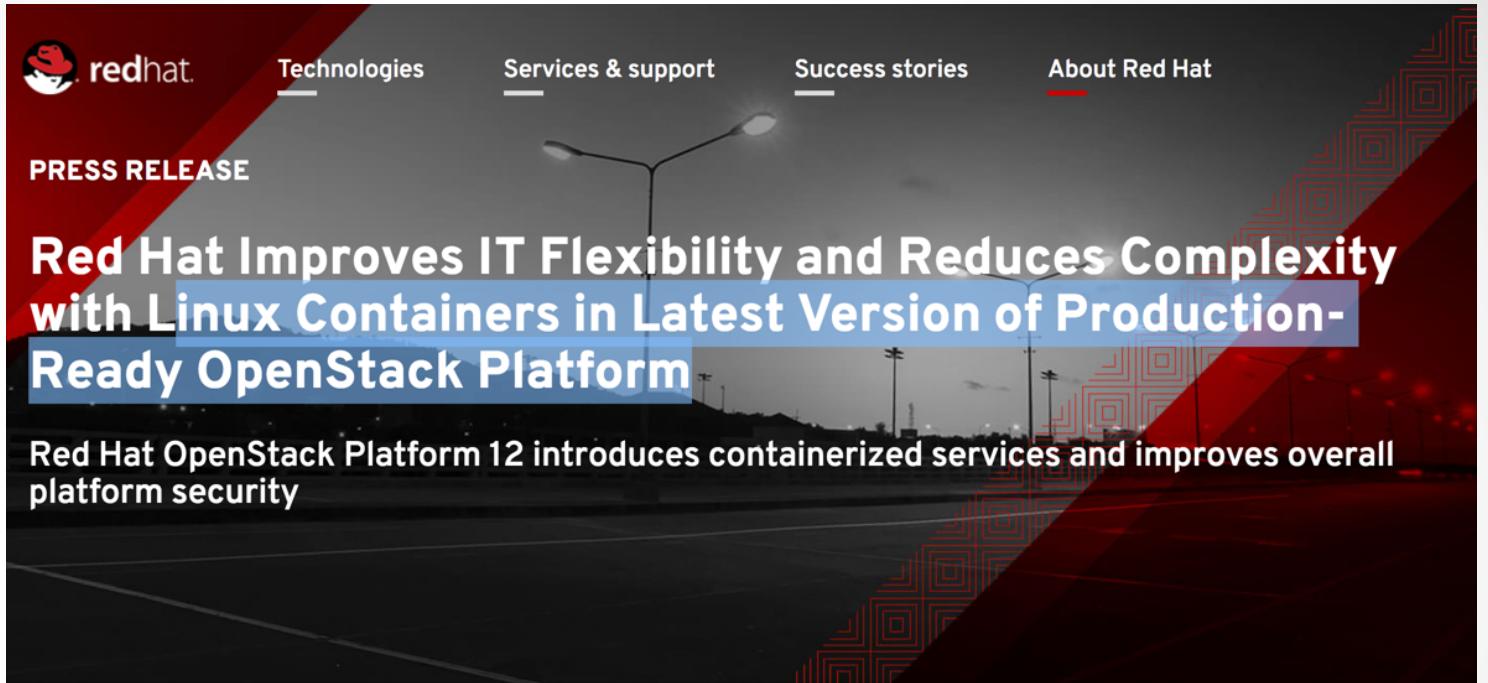
[What are the best power user tools for Windows?](#)

<https://www.slant.co/topics/3929/~docker-orchestration-tools>

Kubernetes 101



What's new on container world



The banner features the Red Hat logo at the top left. A navigation bar with links to 'Technologies', 'Services & support', 'Success stories', and 'About Red Hat' is positioned above a 'PRESS RELEASE' section. The main headline reads: 'Red Hat Improves IT Flexibility and Reduces Complexity with Linux Containers in Latest Version of Production-Ready OpenStack Platform'. Below the headline, a subtext states: 'Red Hat OpenStack Platform 12 introduces containerized services and improves overall platform security'. The background of the banner shows a blurred image of a road at night with streetlights.



The banner for the 'OPENSTACK SUMMIT SYDNEY' features the text 'IN SHORT' at the top. It highlights the introduction of 'Containerized services (DCI)', 'Improved security', and 'Improved security'. Below this, it says 'THIS' and 'Platform, Red'. At the bottom, it includes the location 'SYDNEY, AUSTRALIA', the dates 'NOVEMBER 06th Thru 08th', and the registration link 'Register @ openstack.org/summit'.

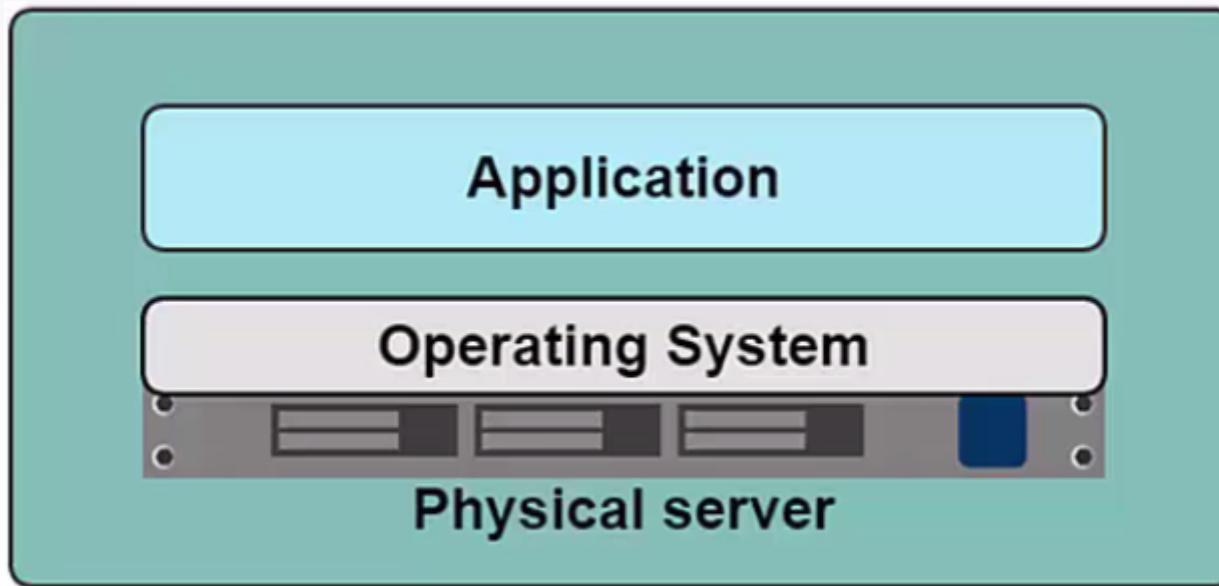
SYDNEY – OpenStack Summit Sydney 2017 – November 6, 2017 – Red Hat, Inc. (NYSE: RHT), the world's leading provider of open-source solutions, today announced Red Hat OpenStack Platform 12, the latest version of Red Hat's massively scalable and agile cloud Infrastructure-as-a-Service (IaaS). Based on the OpenStack "Pike" release, Red Hat OpenStack Platform 12 introduces containerized services, improving flexibility while decreasing complexity for faster application development. Red Hat OpenStack Platform 12 delivers many new enhancements, including upgraded DCI (distributed continuous integration) and improved security to help maintain data compliance and manage risk.

“To achieve the benefits offered by digital transformation, enterprises need to update their infrastructure to better support the next-generation of applications that take

Container Concept and Use Case

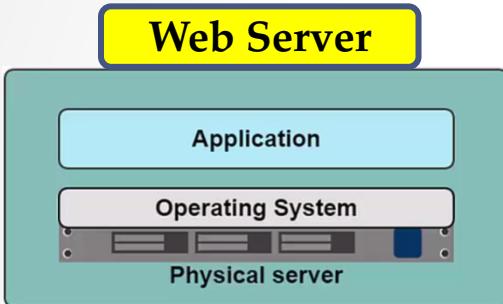


Container Concept and Use Case

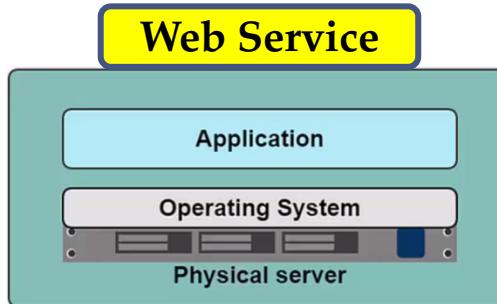


Container Concept and Use Case

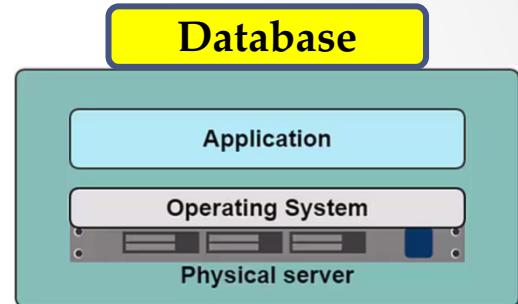
- Production Environment (Best design)
- Day 1: Application 1: Implement



- Apache 2.20 Web Server
- PHP 5.5 Engine
- Larevel 4.1 Framework



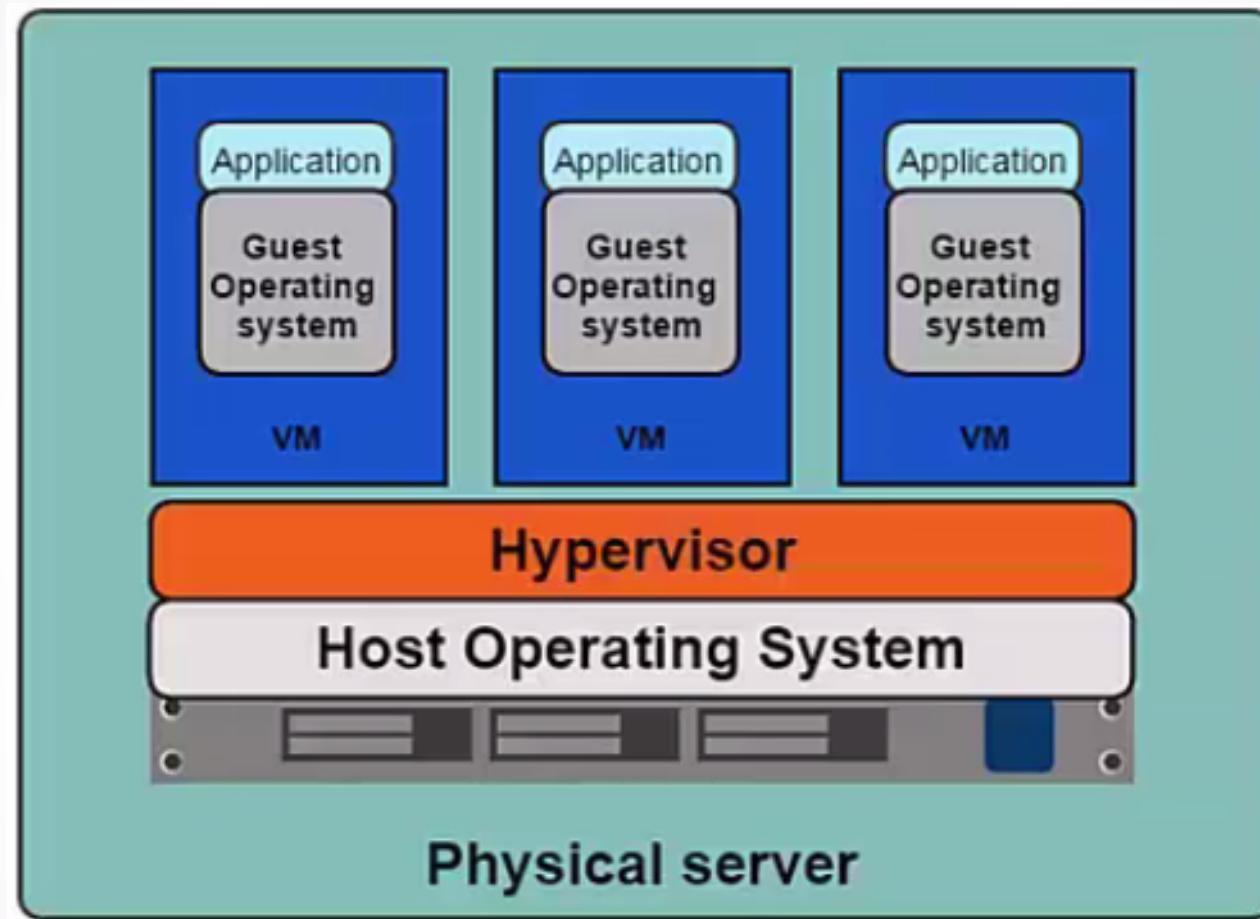
- IIS 8
- .Net FrameWork 3.5



- MariaDB 5.1

- Day 2: Application 2: Need to implement
 - Need PHP 7.0 ?
 - MariaDB 10.1.14 (Need search feature on 10.1)
- Problem ?
 - Possible to upgrade PHP to 7.0 ? / How to test existing application ?
 - What effect to MariaDB upgrade ?

Container Concept and Use Case



Container Concept and Use Case

- Production Environment
- Day 1: Application 1: Implement



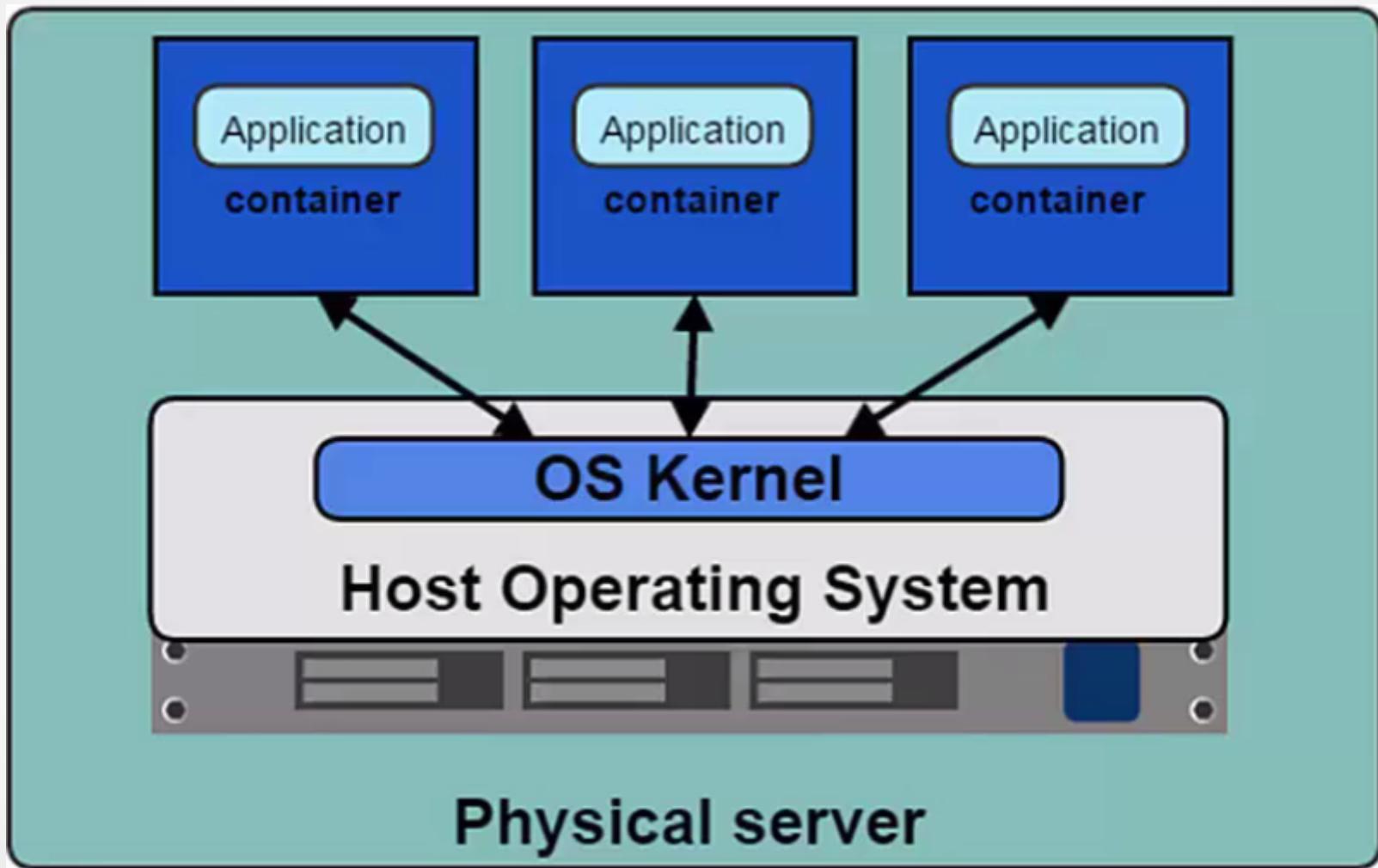
- Apache 2.20 Web Server
- PHP 5.5 Engine
- Laravel 4.1 Framework

- IIS 8
- .Net FrameWork 3.5

- MariaDB 5.1

- Day 2: Application 2: Need to implement
 - Need PHP 7.0 ?
 - MariaDB 10.1.14 (Need search feature on 10.1)
- So... The problem still exist.

Container Concept and Use Case



Container Concept and Use Case



Docker Enterprise Edition (EE) and Community Edition (CE)

Enterprise Edition (EE)

- CaaS enabled platform subscription (integrated container orchestration, management and security)
- Enterprise class support
- Quarterly releases, supported for one year each with backported patches and hotfixes.
- Certified Infrastructure, Plugins, Containers

Community Edition (CE)

- Free Docker platform for "do it yourself" dev and ops
- Monthly Edge release with latest features for developers
- Quarterly release with maintenance for ops

Lifecycle

Squaring the circle: Faster releases and better stability



Docker EE Availability

From Docker



OEM: Direct L2 / L2 Support Included



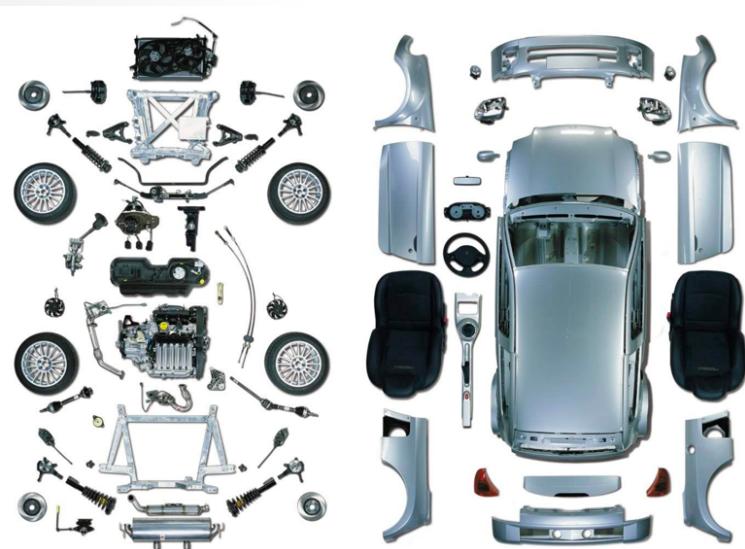
Cloud Marketplaces



Ref: <https://blog.docker.com/2017/03/docker-online-meetup-recap-docker-enterprise-edition-ee-community-edition-ce/>

Container Concept and Use Case

The screenshot shows the homepage of the Moby Project. At the top, there's a dark header bar with a lock icon, the URL "https://mobyproject.org", and several project links: NMac Ked - Mac O..., Medium, Jenkins, vagrant, Mesos, Vue, docker, NGINX, Taiwan, Kubernetes, PWA_Progressive_..., and Other B... Below the header is the Moby Project logo, which is a white circle containing a dark blue stylized bird or whale shape. To the right of the logo, the text "Moby Project" is displayed in a large, white, sans-serif font. Underneath, a subtitle reads: "An open framework to assemble specialized container systems without reinventing the wheel."



The screenshot shows the Moby project's component library and tool interface. At the top, there's a navigation bar with icons for GitHub, Twitter, Blog, Community, and Projects. The main area has a dark background with a large, stylized graphic of a ship's hull and a road leading towards it. On the left, a sidebar lists components: Orchestration, Image Management, Secret Management, Configuration Management, Networking, Provisioning, and a section for "Your Component here". On the right, sections for "Component Library", "Assemblies", and "Moby Tools" are visible.

Container Concept and Use Case

The screenshot shows the official website for rkt. At the top is a dark blue header with the rkt logo and the text "A security-minded, standards-based container engine". Below the header is a navigation bar with three items: "Overview" (selected), "Documentation", and "GitHub Project". The main content area has a light gray background. On the left, there's a section titled "Overview" with text about rkt's purpose and features. To the right, there's a sidebar with "THE LATEST ON RKT" containing two links, and a "MORE INFORMATION" section with a "Download" link. A large orange rocket ship icon with a padlock on its side is positioned in the center of the page.

Overview

rkt is an application container engine developed for modern production cloud-native environments. It features a pod-native approach, a pluggable execution environment, and a well-defined surface area that makes it ideal for integration with other systems.

The core execution unit of rkt is the *pod*, a collection of one or more applications executing in a shared context (rkt's pods are synonymous with [the concept in the Kubernetes orchestration system](#)). rkt allows users to apply different configurations (like isolation parameters) at both pod-level and at the more granular per-application level. rkt's architecture means that each pod executes directly in the classic Unix process model (i.e. there is no central daemon), in a self-contained, isolated environment. rkt implements a modern, open, standard container format, the App Container (appc) spec, but can also execute other container images, like those created with Docker.

Since its introduction by CoreOS in December 2014, the rkt project has greatly matured and is widel



CoreOS teams with Intel to make
Rocket containers more
secure

Container Concept and Use Case

Breaking the
Pattern:
Docker
Enterprise
and
Microservices



Container Concept and Use Case

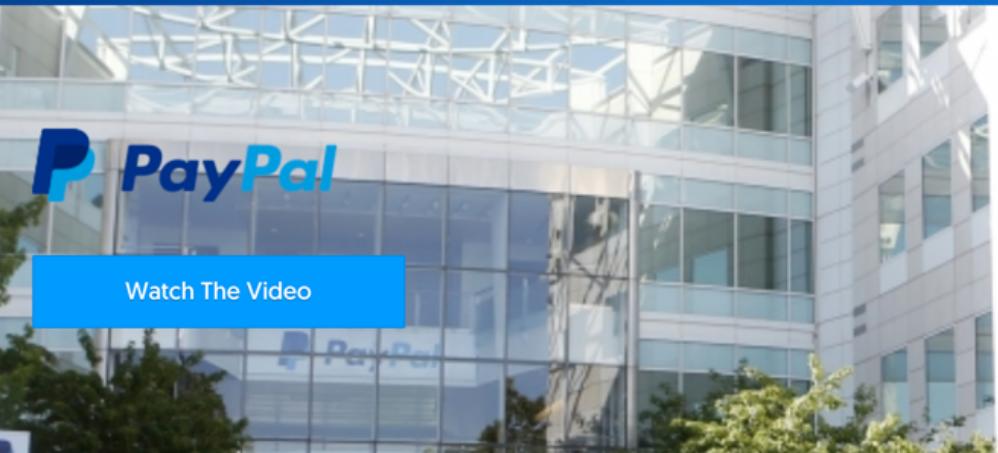
Visa Today

- Live in production for 6 months
- 100K transactions per day
- 10X app scalability
- Multiple clusters across multiple regions

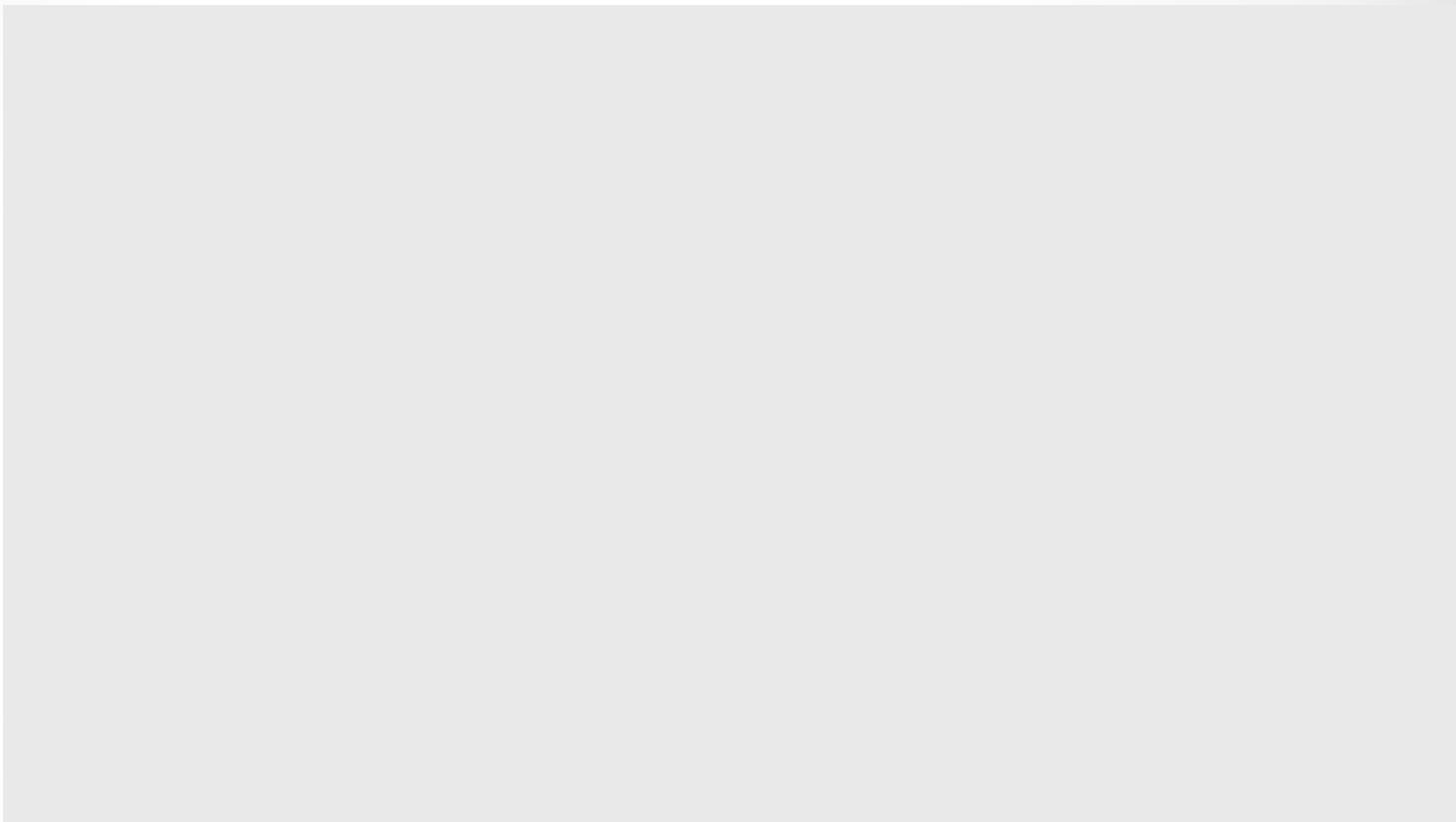


Container Concept and Use Case

PAYPAL USES DOCKER TO CONTAINERIZE EXISTING APPS, SAVE MONEY AND BOOST SECURITY



Container Concept and Use Case



Container Concept and Use Case



Stage 1 Benefits

Decouple

Standardize deployments around Docker containers, rather than individual processes built around app stacks.

Modernize

With apps & dependencies Dockerized, move to modern OS & kernel.

10-20% performance boost to some apps for free.

150,000 containers

700+ apps

18 months

0 code changes

Container Concept and Use Case

Later Stage Benefits



1 Resource Efficiency

50% fewer vCPUs in QA

25% fewer vCPUs in Prod

2 Security

Revoke access to Prod

Automate patching

3 No VM Provisioning

Scale containers (fast)

Not VMs (slow)

4 Availability

Increased resiliency

"Anything can run anywhere"

5 Availability Zones

Application deployments of
an entire AZ in a few hours.

6 Consistent Platform

Consistent tooling

Standard playbook across
app stacks

> 50% boost

to developer local
build-deploy-test
speed

Developer
Freedom

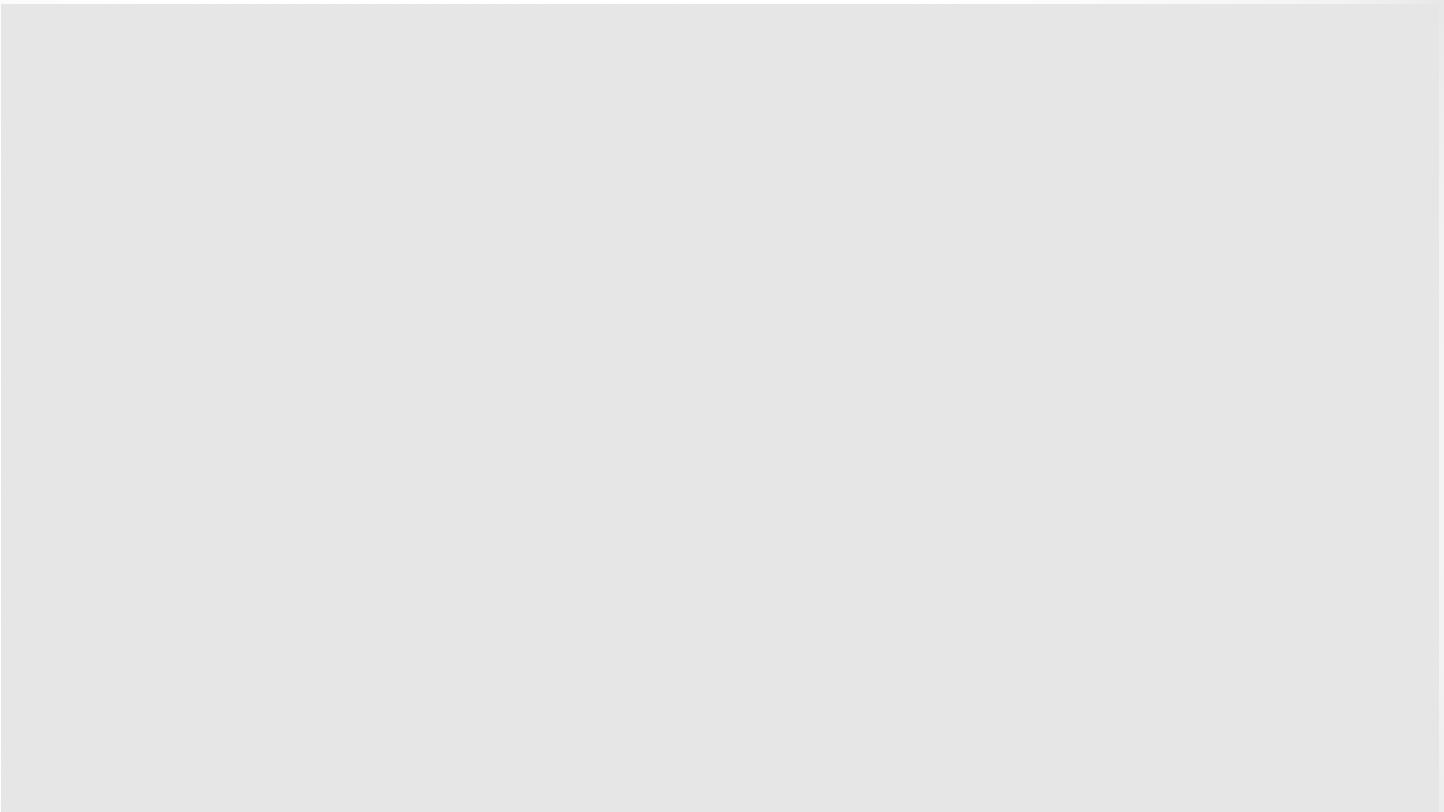
for greenfield apps

Container Concept and Use Case



U B E R

Kubernetes 101



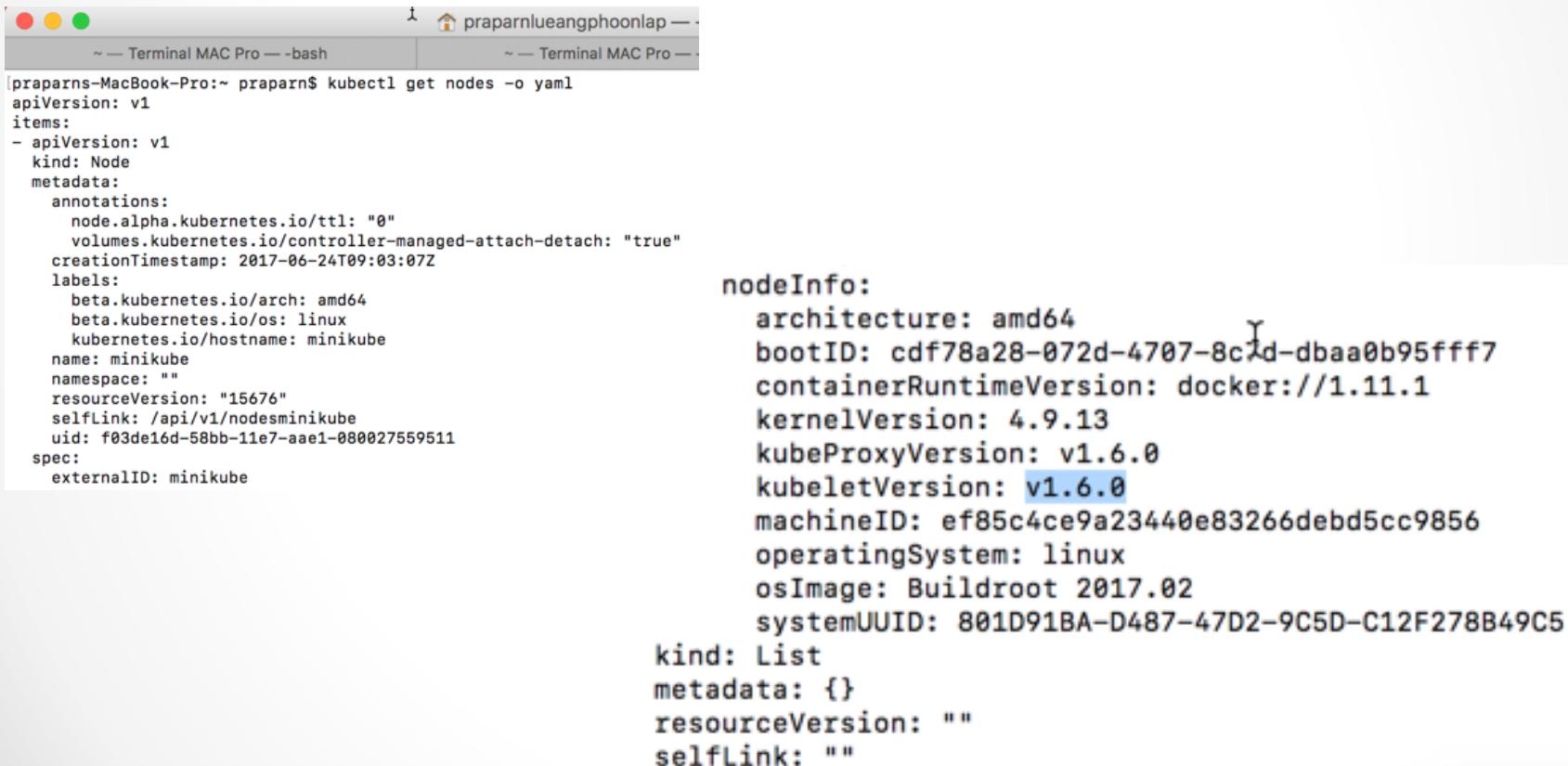
Principle of Kubernetes



Principle of Kubernetes

- Check kubernetes version

```
kubectl get nodes -o yaml
```



```
[praparns-MacBook-Pro:~ praparn$ kubectl get nodes -o yaml
apiVersion: v1
items:
- apiVersion: v1
  kind: Node
  metadata:
    annotations:
      node.alpha.kubernetes.io/ttl: "0"
      volumes.kubernetes.io/controller-managed-attach-detach: "true"
    creationTimestamp: 2017-06-24T09:03:07Z
    labels:
      beta.kubernetes.io/arch: amd64
      beta.kubernetes.io/os: linux
      kubernetes.io/hostname: minikube
    name: minikube
    namespace: ""
    resourceVersion: "15676"
    selfLink: /api/v1/nodes/minikube
    uid: f03de16d-58bb-11e7-aae1-080027559511
  spec:
    externalID: minikube
          nodeInfo:
            architecture: amd64
            bootID: cdf78a28-072d-4707-8c1d-dbaa0b95fff7
            containerRuntimeVersion: docker://1.11.1
            kernelVersion: 4.9.13
            kubeProxyVersion: v1.6.0
            kubeletVersion: v1.6.0
            machineID: ef85c4ce9a23440e83266debd5cc9856
            operatingSystem: linux
            osImage: Buildroot 2017.02
            systemUUID: 801D91BA-D487-47D2-9C5D-C12F278B49C5
    kind: List
    metadata: {}
    resourceVersion: ""
    selfLink: "
```

Principle of Kubernetes

THE LINUX FOUNDATION PROJECTS



About ▾ Projects ▾ People ▾ Community ▾ Newsroom ▾



NOW AVAILABLE: SELF-PACED CLASS KUBERNETES FUNDAMENTALS

Learn how to deploy a containerized application and manipulate resources via the API.

REGISTER NOW

CURRENTLY HOSTED PROJECTS



Prometheus



containerd



Kubernetes 101



Principle of Kubernetes

- What is Orchestration (Computing)?
 - Align business request with Application/Data/Infrastructure
 - Centralize management for:
 - Resource Pool
 - Automated Workflow
 - Provisioning
 - Scale Up/Down
 - Monitoring
 - Billing
 - etc

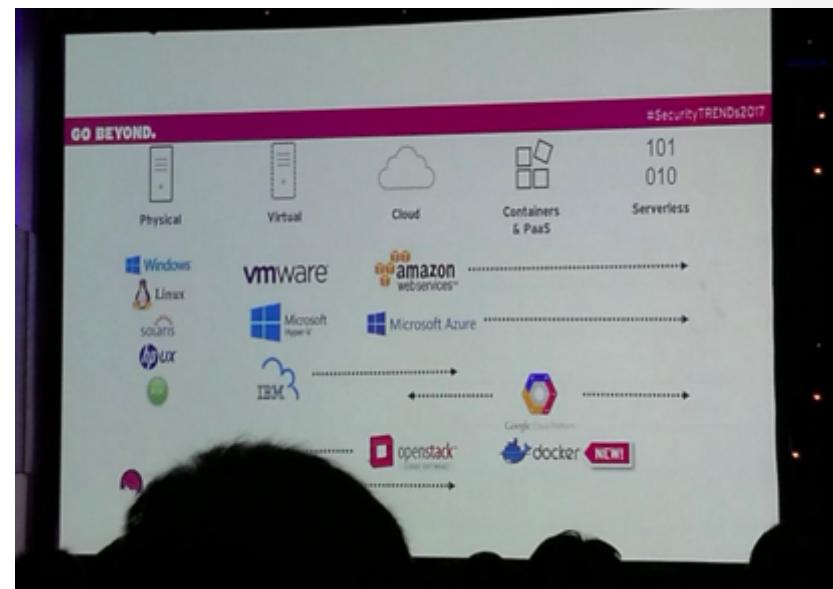
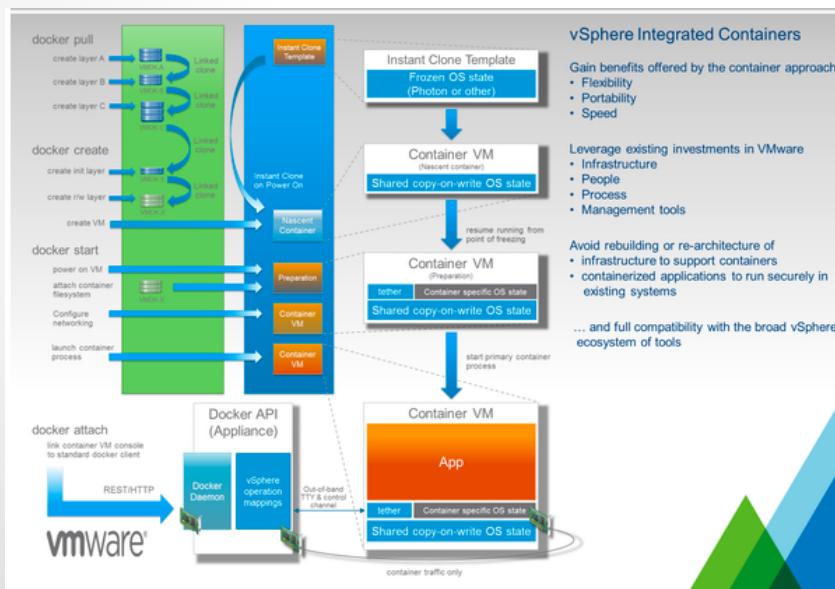


Ref: [https://en.wikipedia.org/wiki/Orchestration_\(computing\)](https://en.wikipedia.org/wiki/Orchestration_(computing))

Kubernetes 101

Principle of Kubernetes

- But why Container Orchestration ?
 - Container Platform is next generation that partner/beat existing virtualize technology
 - Easy to build-ship-run on dev and production
 - Zero Configure
 - Suitable for Microservice architecture
 - Good for application, Good for business (Win/Win !!!)



Principle of Kubernetes

How HPE Synergy works

Explore the Engine of the Idea Economy

Imagine infrastructure so intelligent, it can adapt to every workload and deliver services at lightning speed.

Introduction to HPE Synergy Composable Infrastructure Software-defined Intelligence Pooling and Provisioning Simple Scalability Open Integration



Server Profiles Search

fra-lin150

General

Managed by Appliance: M.114.220.5
Description: Not set
Server hardware: ERX BladeSystem11.bug.5
Affinity: Bay
Serial Number (Virtual): VCGZGE04N0R0
UUID (Virtual): 92127379-0ef1-4f7e-b14d-f952987705
Mac Type: Virtual
WWN Type: Virtual
Location: Badminton

Status

Status: Normal

Profile Name	State
fra-lin115	Online
fra-lin116	Online
fra-lin117	Online
fra-lin118	Online
fra-lin119	Online
fra-lin120	Online
fra-lin121	Online
fra-lin122	Online
fra-lin123	Online
fra-lin124	Online
fra-lin125	Online
fra-lin126	Online
fra-lin127	Online
fra-lin128	Online
fra-lin129	Online
fra-lin130	Online
fra-lin131	Online
fra-lin132	Online
fra-lin133	Online
fra-lin134	Online
fra-lin135	Online
fra-lin136	Online
fra-lin137	Online
fra-lin138	Online
fra-lin139	Online
fra-lin140	Online
fra-lin141	Online
fra-lin142	Online
fra-lin143	Online
fra-lin144	Online
fra-lin145	Online
fra-lin146	Online
fra-lin147	Online
fra-lin148	Online
fra-lin149	Online
fra-lin150	Online
fra-lin151	Online
fra-lin152	Online
fra-lin153	Online
fra-lin154	Online
fra-lin155	Online

Appliances Search

test-appliance.hpcoc.com

Normal

Version: 2.00.02-019211
Model: HP OneView - Demo VM

Credentials

Username: Administrator
Location: https://10.93.22.135

Resources

149 Server Profiles

Reconnect Remove

Appliance Name	Status
AP-SOUTH-EASTuse.rdb.hpecorp.net	Online
AP-CENTRALcn.hpecorp.net	Online
US-EASTuse.rdb.hpecorp.net	Online
US-WESTrdb.hpecorp.net	Online
EU-CENTRALuse.rdb.hpecorp.net	Online
AP-SOUTH-EASTsh.hpecorp.net	Online
AP-SOUTH-EASThk.hpecorp.net	Online
AMS-SOUTHuse.rdb.hpecorp.net	Online
MENAuse.rdb.hpecorp.net	Online
AFRICAuse.rdb.hpecorp.net	Online
US-CENTRALuse.rdb.hpecorp.net	Online
AP-NORTH-EASTuse.rdb.hpecorp.net	Online
AP-SOUTH-EASTuse.rdb.hpecorp.net	Online
AP-CENTRALuse.rdb.hpecorp.net	Online
EU-CENTRALuse.rdb.hpecorp.net	Online
MENAuse.rdb.hpecorp.net	Online
AFRICAuse.rdb.hpecorp.net	Online
AMS-NORTHuse.rdb.hpecorp.net	Online
US-CENTRALuse.rdb.hpecorp.net	Online
INDIAuse.rdb.hpecorp.net	Online
test-appliance.hpcoc.com	Online

Your infrastructure as code

Gain efficiency and control with HPE Synergy, a powerful software-defined solution that lets you manage your infrastructure as code, deploying IT resources quickly and for any workload. Through a single interface, you can compose fluid pools of physical and virtual compute, storage, and fabric resources into any configuration for any application. With HPE Synergy, IT goes beyond the role of internal service broker to offer new applications that elevate and enhance the business.

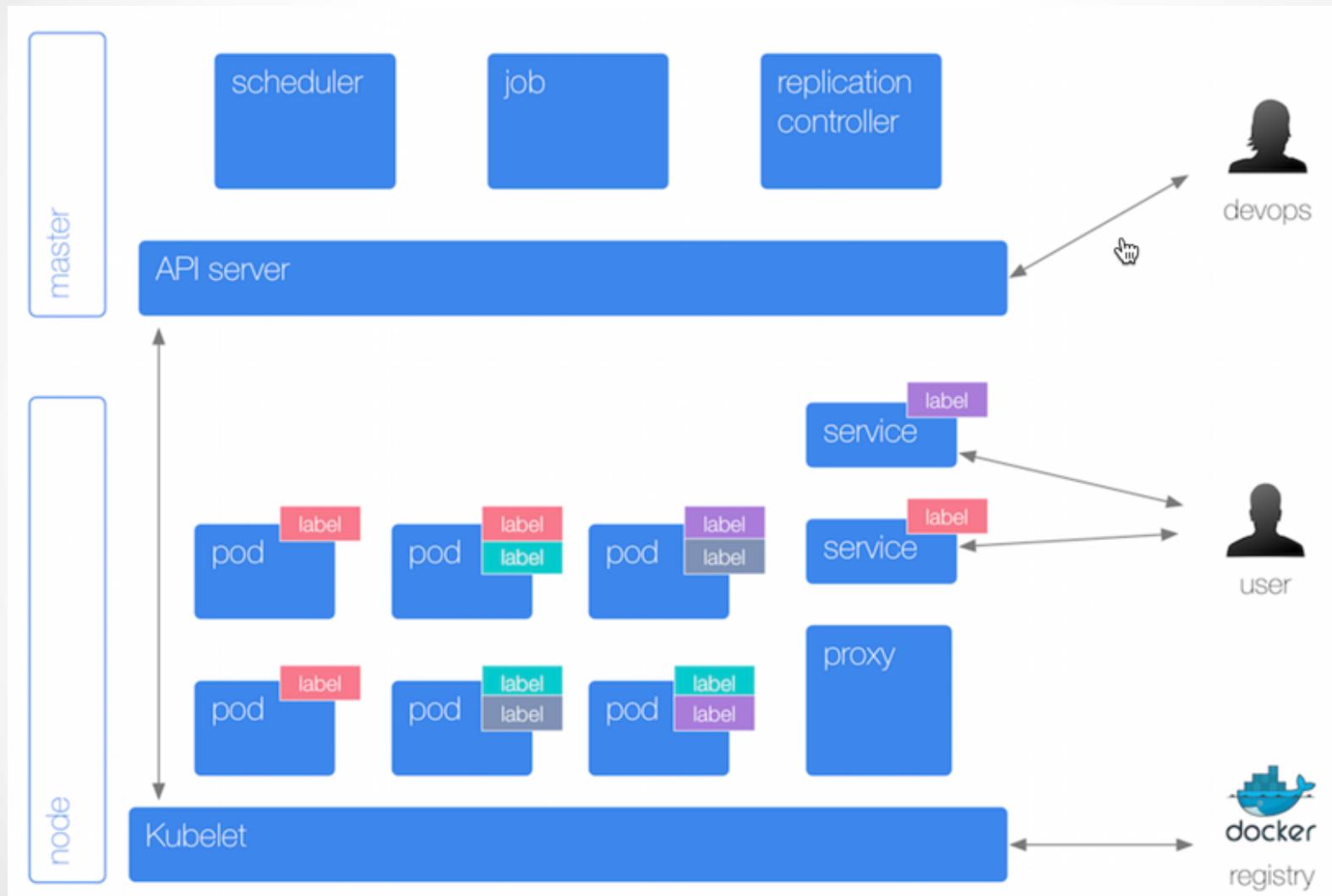
Principle of Kubernetes

- But why we need Container Orchestration ?
 - Production environment is cluster system
 - Microservice maintain connect was required
 - Application state-full will run on stateless architecture
 - Application scale up/Down was required
 - Many native command/shell for maintain container in production environment
 - etc

Principle of Kubernetes



Principle of Kubernetes



Principle of Kubernetes

- Key Feature
 - Automatic binpacking
 - Horizontal Pod Autoscaling (HPA)
 - Automated rollouts and rollbacks
 - Storage orchestration
 - Self-healing
 - Service discovery and load balancing
 - Secret and configuration management
 - Batch execution



Principle of Kubernetes

- Automatic binpacking

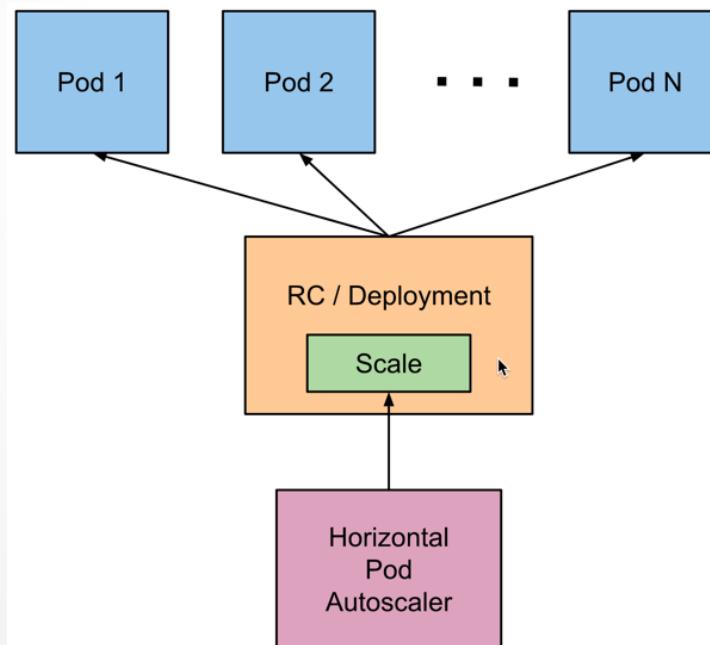
- CPU/Memory's utilization can define on Pods (Smallest Unit of Kubernetes)
- Schedule will select the node by ensure all resource is enough for running Pods as required
- If reach memory limit
 - Current Pods will be terminate (Kill)
 - If restart flag was set. Kubenetes will try to restart Pods on other node
- If reach cpu limit
 - Schedule will not kill Pods and waiting for it back to normal state
- Check available node resource by command: kubectl describe node

Non-terminated Pods: (3 in total)					
Namespace	Name	CPU Requests	CPU Limits	Memory Requests	Memory Limits
kube-system	kube-addon-manager-minikube	5m (0%)	0 (0%)	50Mi (2%)	0 (0%)
kube-system	kube-dns-268032401-mx7s3	260m (13%)	0 (0%)	110Mi (5%)	170Mi (8%)
kube-system	kubernetes-dashboard-hdhrz	0 (0%)	0 (0%)	0 (0%)	0 (0%)

Allocated resources:					
(Total limits may be over 100 percent, i.e., overcommitted.)					
CPU Requests	CPU Limits	Memory Requests	Memory Limits		
265m (13%)	0 (0%)	160Mi (8%)	170Mi (8%)		
Events:	<none>				

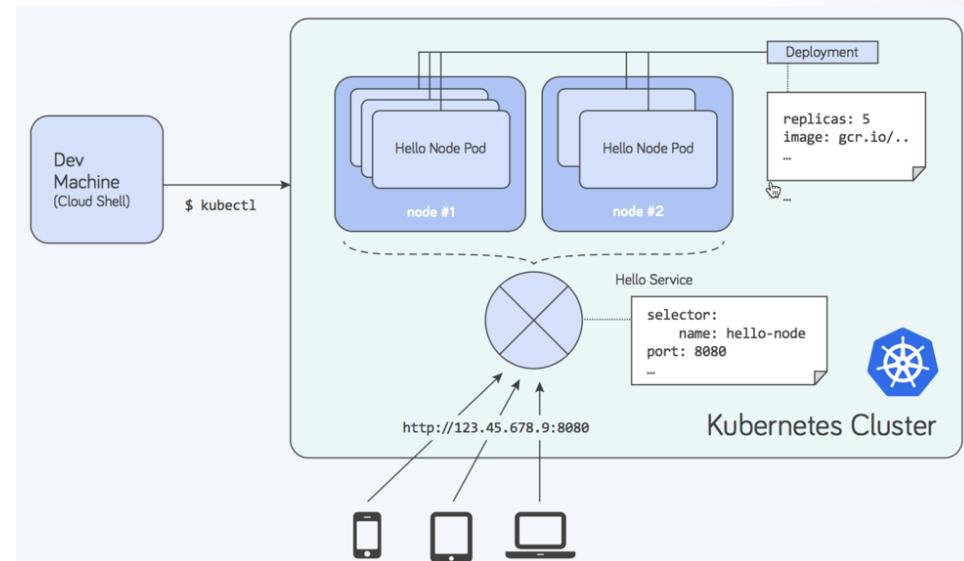
Principle of Kubernetes

- Horizon Pods Autoscaling (hpa)
 - Talk with RC (Replication Controller) for complete task
 - Automatic scaling Pods's number base cpu's utilization
 - Support multiple criteria (metric) for scale (Alpha feature)
 - Support customize criteria (metric) for scale (Alpha feature)
 - Looping check resource's utilization every 30 seconds (default)



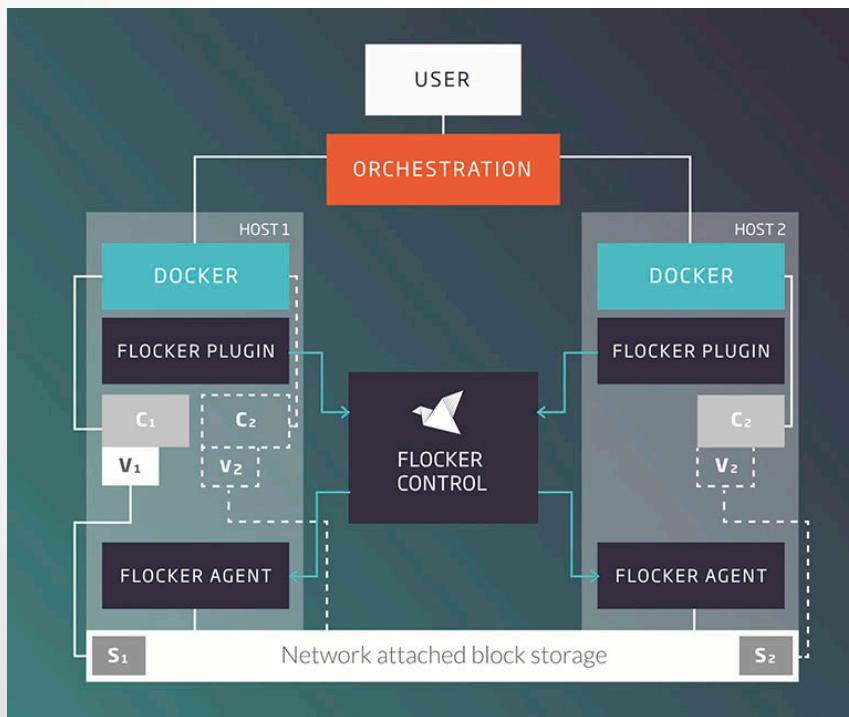
Principle of Kubernetes

- Automated Rollout and Rollbacks
 - Update will base on Pod's template
 - Rollout existing deployment with all remain replicas number as desired
 - If rollout is success. New version will be provision until success
 - Rollback will possible with single command
 - Rollout process can pause/resume as need



Principle of Kubernetes

- Storage Orchestrator
 - Support several storage type:
 - Local Storage
 - Network Storage (NFS, iScsi, Gluster, Ceph, Cinder, Flocker)
 - Cloud Storage (AWS, GCE, Azure Disk etc)



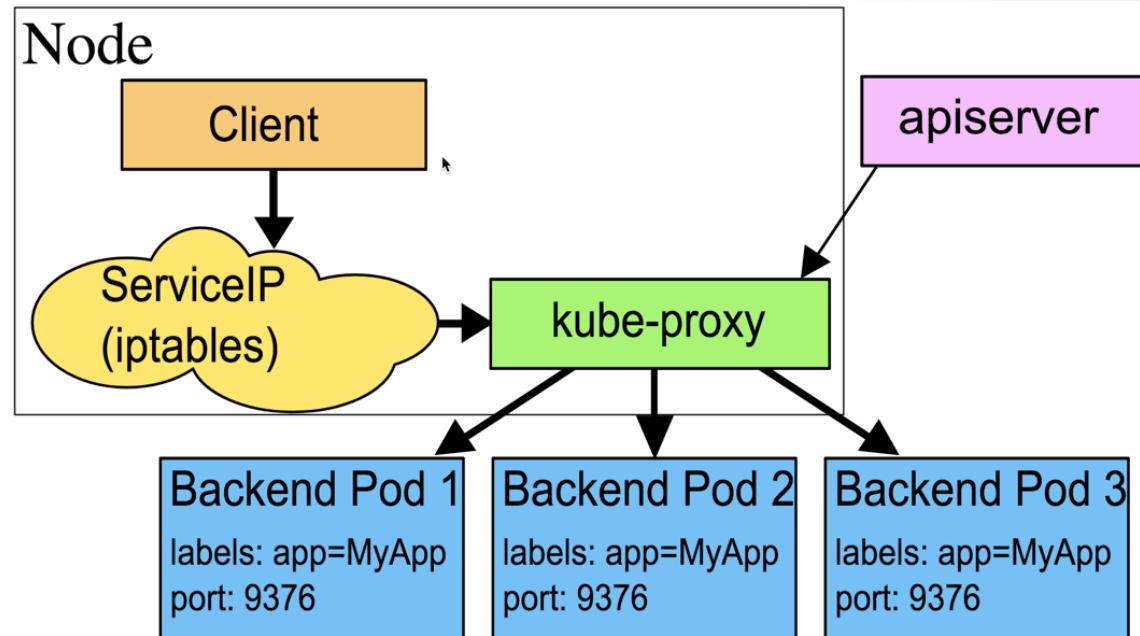
Principle of Kubernetes

- Self-healing
 - Replication Controller (RC) will maintain unit of Pods as design (not to much (kill) and not to few (create))
 - Full-fill Pods on every failure case with automatic by system



Principle of Kubernetes

- Service Discovery and Load Balancing
 - Service will act like “connector” for client need to connect with Pods
 - Discovery will use for service to look “Pods” by environment variable or dns service
 - Support load balancing between multiple Pods (replica)



Principle of Kubernetes

- Secret and Configuration Management
 - Kubernetes can keep confidential data (such as username/password) for running application on encrypt format.
 - Can reference on Pods instead plan text configuration.

```
$ kubectl get secrets
NAME          TYPE        DATA  AGE
db-user-pass  Opaque      2     51s

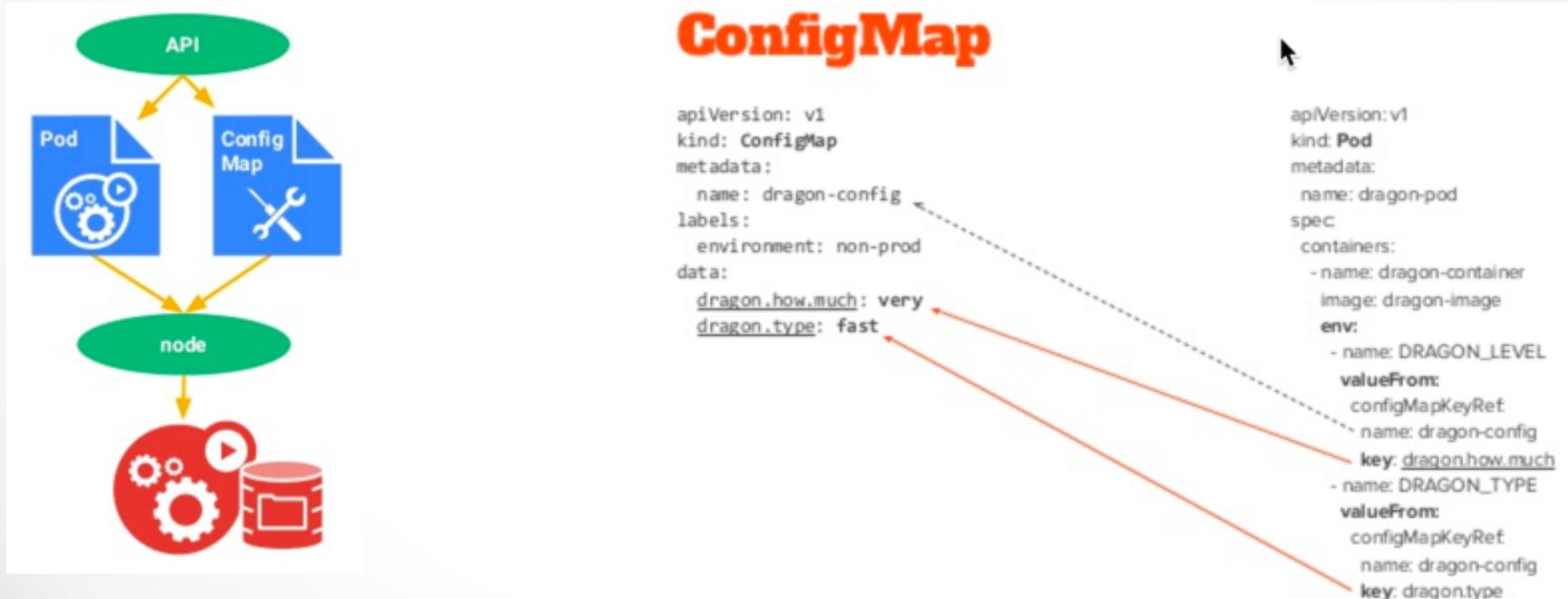
$ kubectl describe secrets/db-user-pass
Name:         db-user-pass
Namespace:    default
Labels:       <none>
Annotations: <none>

Type:         Opaque

Data
====
password.txt: 12 bytes
username.txt: 5 bytes
```

Principle of Kubernetes

- Secret and Configuration Management
 - Sometime we have a many configuration that need to specify on each application, But it should be change every time need.
 - Idea is create configuration file (ConfigMap) for define all configuration that reference on Pods instead



Principle of Kubernetes

- Batch Execution
 - A job will response some kind of batch execute by create special Pods (Terminate when batch complete)
 - Normally kubernetes will use job for maintain many background process for cluster system such as Replication Controller (RC) will submit job for start new Pods when existing is fail or delete.
 - Type of Job
 - Non-parallel jobs
 - Parallel jobs with fix-completion
 - Parallel jobs with work queue

K8S vs Docker what is the difference ?



K8S vs Docker what is the difference ?

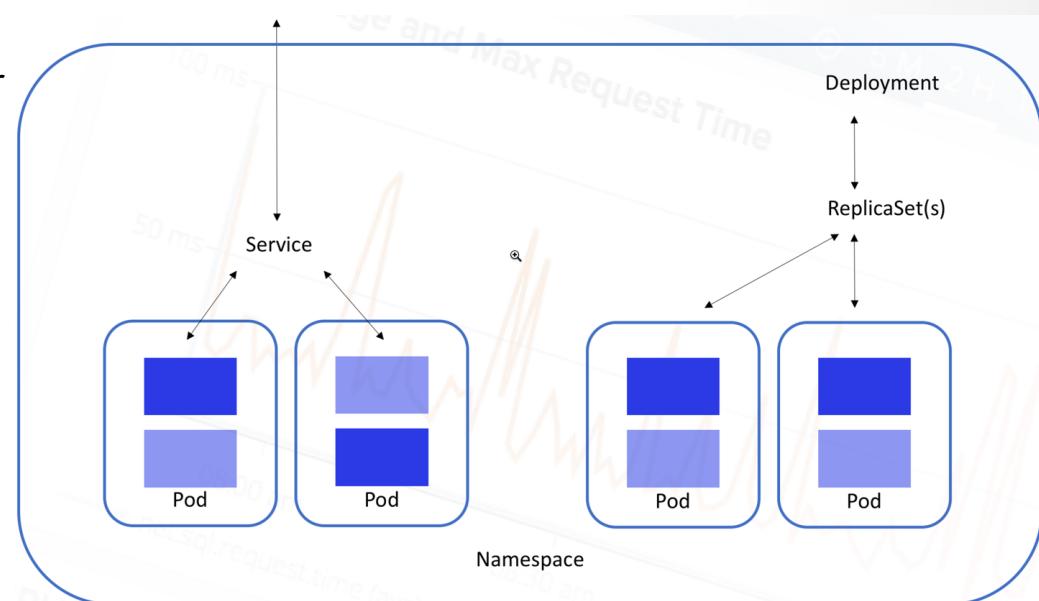
- The **difference concept** of Docker/K8S
 - Docker Native/Swarm Mode:
 - Simplify (Single command for initial/manage/maintain)
 - Provide all resource within suite (overlay network, key value etc) by default
 - Fastest deployment
 - K8S (Kubernetes):
 - Production grade orchestrator (Customizable)
 - Customize by design (network plugin, storage etc)
 - Complex configuration support for handle application

K8S vs Docker what is the difference ?

Topic	K8S	Docker/Swarm
Architecture	Open-system (Base on cluster manager "Borg" for support complex workload)	Swarm: Proprietary of Docker product, "Easy to use", "Extend capability of Docker in cluster"
Operation command	Almost operate by "YAML" file (Declarative Command)	Almost operate by "command" (Imperative Command)
Unit of Work	Pods (Pods >= Container)	Container
How to Identify Work	"Label operation"	Docker: By container name Swarm: By service/stack name
Level of workload management	Service Level: (Simple) Replication Level: (Auto healing) Deployment Level: (Auto healing + Roll Update)	Docker: N/A Swarm: Service Level (Snag with service/stack)
Auto scaling	HPA (Horizontal Pods Scaling) base on CPU	No
Health check	Liveness & Readiness (Multi option to check application health)	Service health only

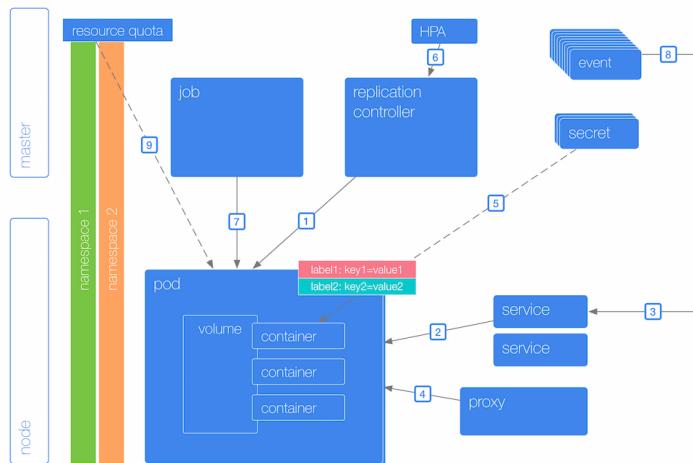
K8S vs Docker what is the difference ?

- Workload Management
 - Swarm operation
 - docker service (run/ps/stop/rm etc)
 - docker stack (deploy/ls/ps/services etc)
 - K8S Operation
 - Service
 - Replication Controller
 - Deployment + RS

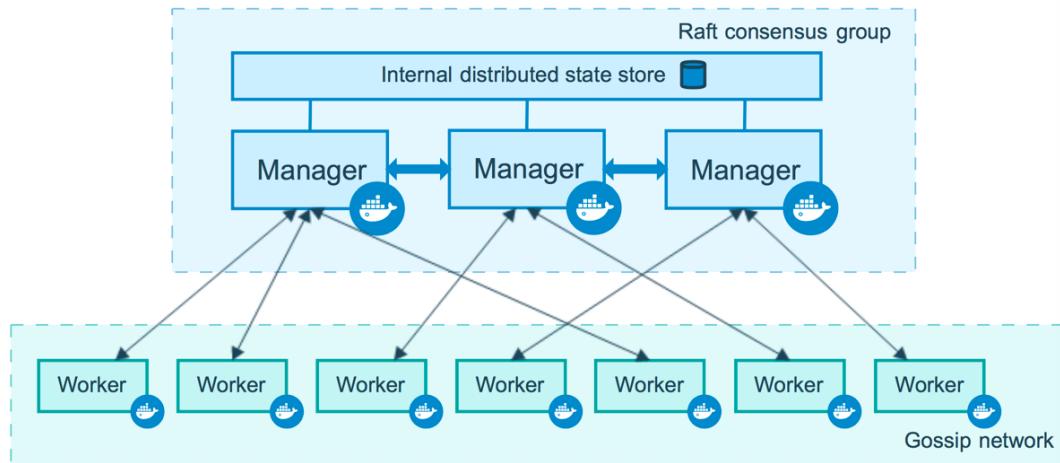


K8S vs Docker what is the difference ?

- K8S Architecture



- Docker Swarm Architcture



K8S vs Docker what is the difference ?

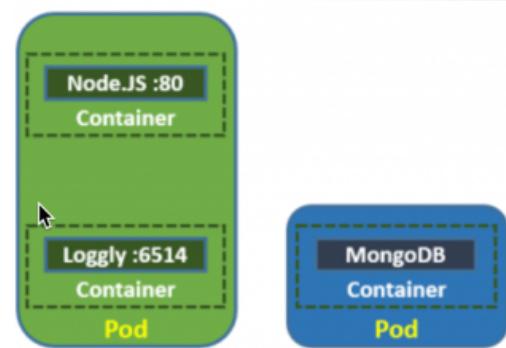
- Docker Command
 - “docker container run -dt --name webtest -p 5000:5000 labdocker/cluster:webserviceelite”
- K8S Pods&Service (YAML File)

```
1 apiVersion: "v1"
2 kind: Pod
3 metadata:
4   name: webtest
5   labels:
6     name: web
7     owner: Praparn_L
8     version: "1.0"
9   module: WebServer
10  environment: development
11 spec:
12   containers:
13     - name: webtest
14       image: labdocker/cluster:webserviceelite
15       ports:
16         - containerPort: 5000
17           protocol: TCP
```

```
1 apiVersion: v1
2 kind: Service
3 metadata:
4   name: webtest
5   labels:
6     name: web
7     owner: Praparn_L
8     version: "1.0"
9   module: WebServer
10  environment: development
11 spec:
12   selector:
13     name: web
14     owner: Praparn_L
15     version: "1.0"
16     module: WebServer
17     environment: development
18
19   type: NodePort
20   ports:
21     - port: 5000
22       name: http
23       targetPort: 5000
24       protocol: TCP
25
```

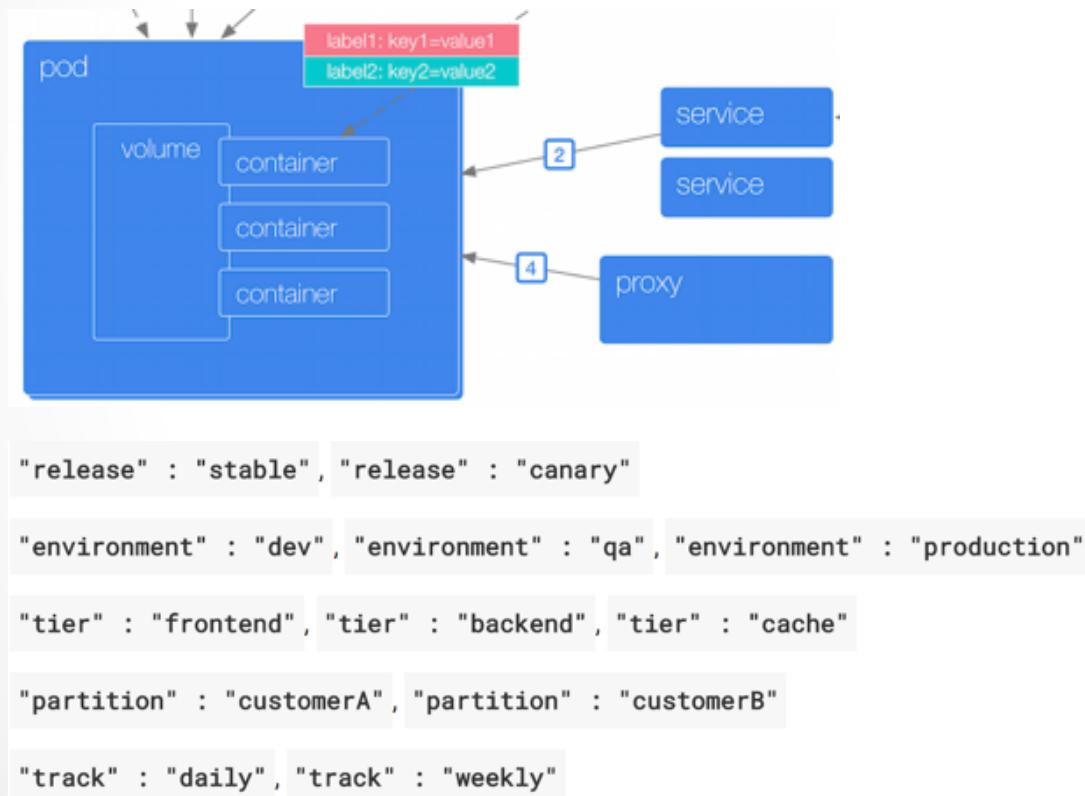
K8S vs Docker what is the difference ?

- Unit of Work (Pods vs Container)
 - All container on same Pods will share:
 - Process ID (PID)
 - Network access (Communicate to each other via “localhost”)
 - Internal Process Command (IPC)
 - Unix Time-Sharing (UTS)
 - Hostname
 - IP Address/Ports
 - Use Case for Multiple Pods:
 - Apache (1 Container) +Tomcat (1 Container)
 - Apache(1 Container) + PHP (1 Container)
 - Nginx (Cache: 1 Container) + Apache/PHP (1 Container)
 - Web Server (1 Container) + Data Volume(Cache: 1 Container)
 - Pods will can create replicas of 1000+ set on cluster system



K8S vs Docker what is the difference ?

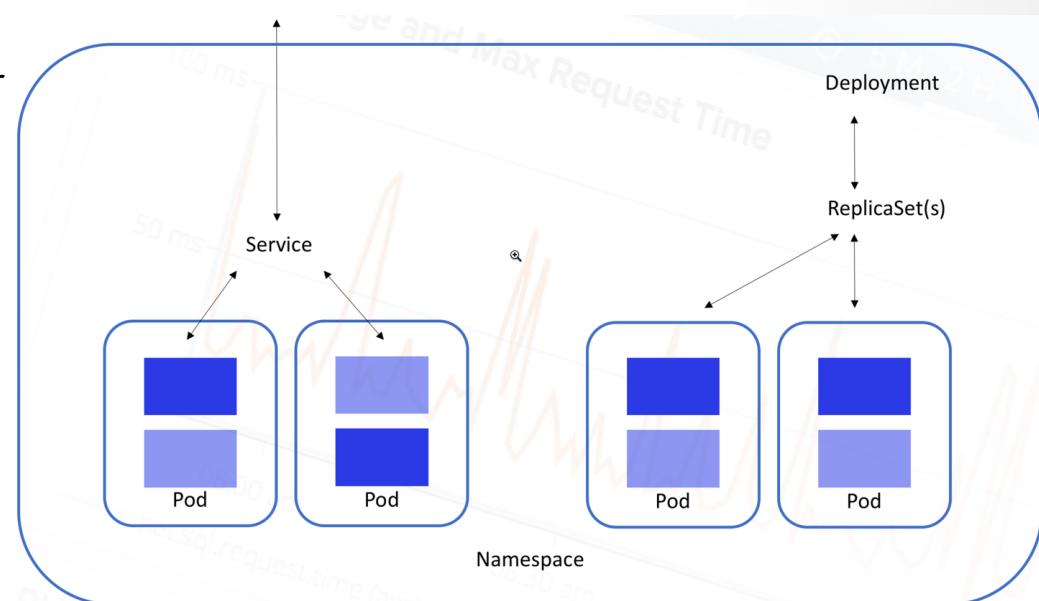
- Identify Work (Label is base advantage of K8S)
 - All component of K8S will blinding with “label”
 - Label is free-form, Customize by yourself !!!



```
1  apiVersion: v1
2  kind: Service
3  metadata:
4    name: webtest
5    labels:
6      name: web
7      owner: Praparn_L
8      version: "1.0"
9      module: WebServer
10     environment: development
11
12   spec:
13     selector:
14       name: web
15       owner: Praparn_L
16       version: "1.0"
17       module: WebServer
18       environment: development
19
20     type: NodePort
21     ports:
22       - port: 5000
23         name: http
24         targetPort: 5000
25         protocol: TCP
```

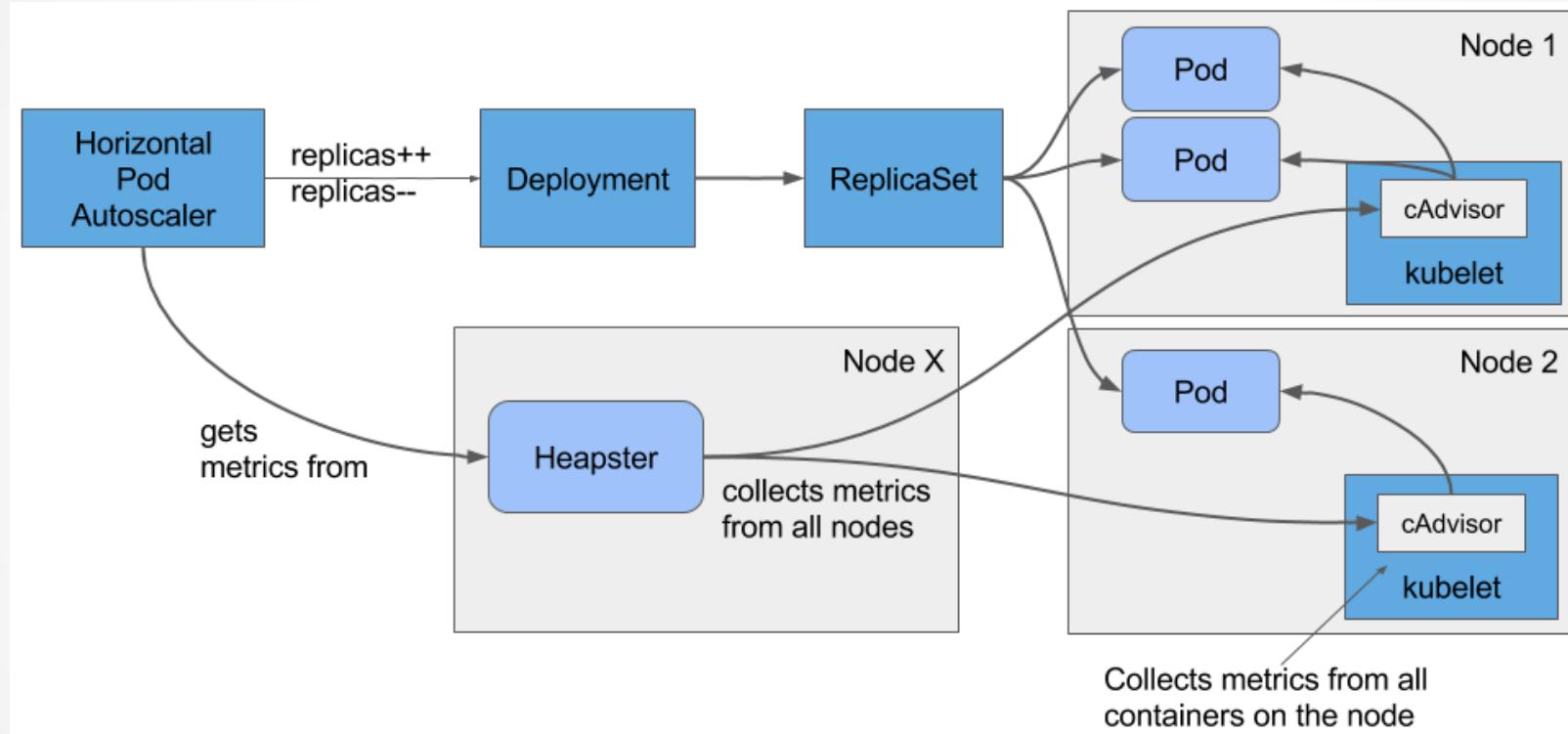
K8S vs Docker what is the difference ?

- Workload Management
 - Swarm operation (Service/Stack)
 - docker service (run/ps/stop/rm etc)
 - docker stack (deploy/ls/ps/services etc)
 - K8S Operation
 - Service
 - Replication Controller
 - Deployment + RS



K8S vs Docker what is the difference ?

- Auto Scale (K8S Horizontal Pod Autoscaler (HPA))



K8S vs Docker what is the difference ?

- Health Check
 - Docker: monitor by service on container
 - Kubernetes: monitor by service + customize
 - Liveness: Pods/Container is still alive (if not restart pods)
 - Readiness Pods/Container is ready to use (if not redirect workload to another pods)
 - Technic for check (Both Liveness/Readiness):
 - ExecAction:
 - TCPSocketAction:
 - HTTPGetAction:



How to acquire K8S on your machine



How to acquire K8S to your machine

- Way1: Minikube (VirtualBox: Native Way)

```
praparns-MacBook-Pro% minikube start
There is a newer version of minikube available (v0.26.0). Download it here:
https://github.com/kubernetes/minikube/releases/tag/v0.26.0

To disable this notification, run the following:
minikube config set WantUpdateNotification false
Starting local Kubernetes v1.9.0 cluster...
Starting VM...
Getting VM IP address...
Moving files into cluster...
Setting up certs...
Connecting to cluster...
Setting up kubeconfig...
Starting cluster components...
Kubectl is now configured to use the cluster.
Loading cached images from config file.
praparns-MacBook-Pro% minikube status
minikube: Running
cluster: Running
kubectl: Correctly Configured: pointing to minikube-vm at 192.168.99.10
praparns-MacBook-Pro% kubectl config use-context minikube
Switched to context "minikube".
praparns-MacBook-Pro% kubectl get nodes
NAME      STATUS    ROLES   AGE     VERSION
minikube  Ready     <none>  29d    v1.9.0
praparns-MacBook-Pro%
```

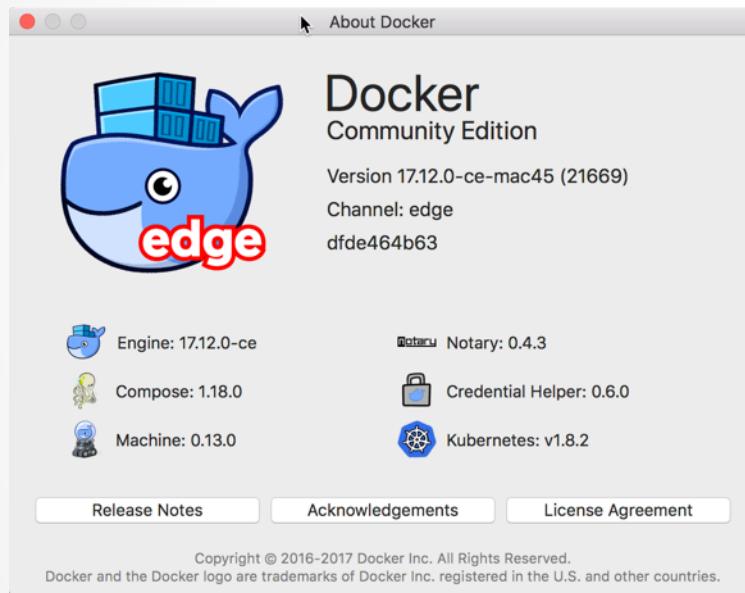


minikube

The screenshot shows the Kubernetes dashboard running on port 192.168.99.100. The left sidebar has tabs for Overview, Namespaces, Nodes, Persistent Volumes, Roles, Storage Classes, and a dropdown for Namespace (set to default). The main area has two sections: 'Cluster' and 'Discovery and Load Balancing'. Under 'Cluster', there's a table for 'Services' with one entry: 'kubernetes' (Cluster IP: 10.96.0.1, Internal endpoints: kubernetes:443 TCP, External endpoints: kubernetes:0 TCP, Age: 29 days). Under 'Discovery and Load Balancing', there's a table for 'Secrets' with one entry: 'default-token-Inhmd' (Type: kubernetes.io/service-account-token, Age: 29 days).

How to acquire K8S to your machine

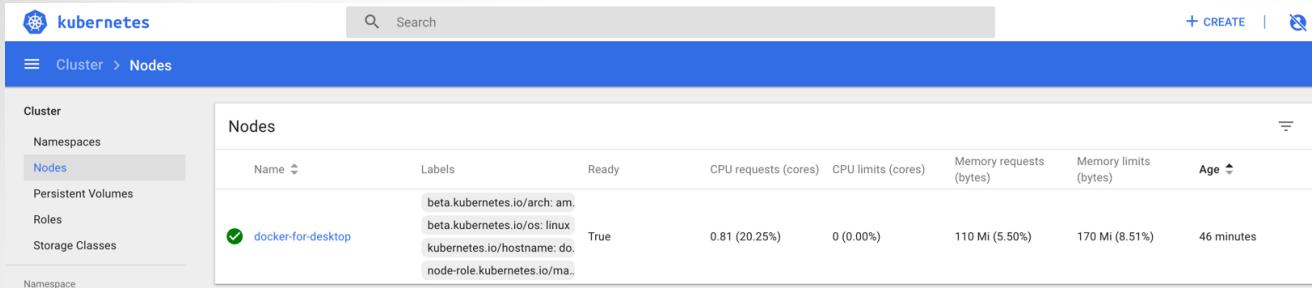
- Way2 : Docker Community Edition (CE)
 - Docker for Windows (**Available Now with Edge Edition !!!**)
 - Docker for MAC (**Available Now with Edge Edition !!!**)



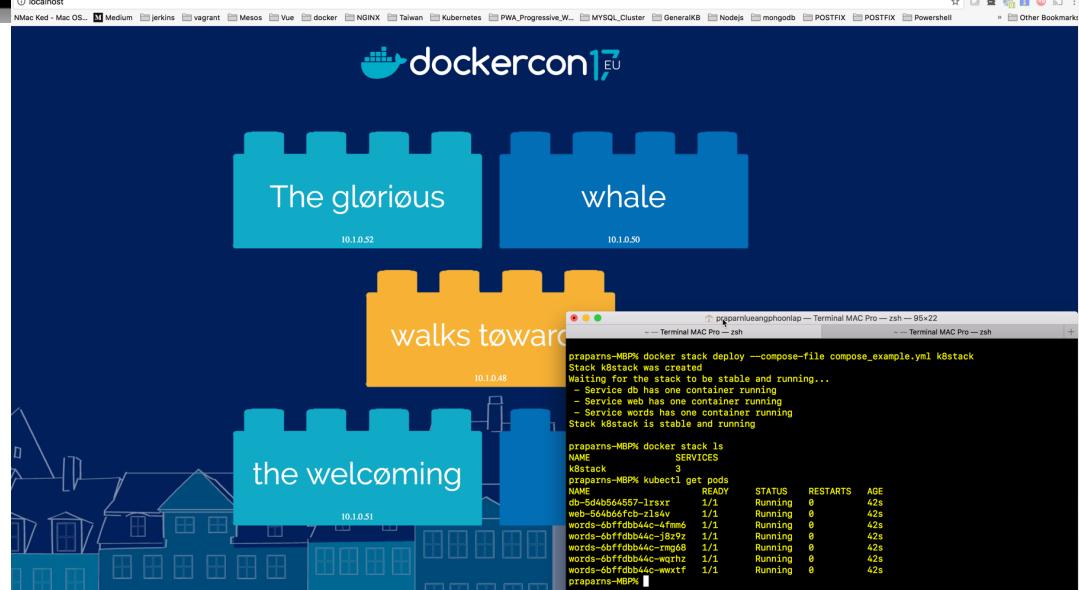
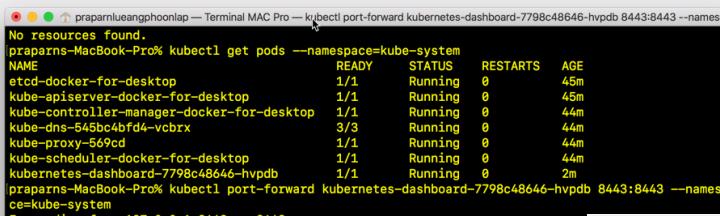
How to acquire K8S to your machine

- Docker CE's Benefit
 - Reduce complexity of K8S setup in this platform (Initial Farm, HA Setup etc)
 - Utilization host resource from mix workload for Swarm and K8S (Docker EE)
 - Developer feel free for testing both Swarm/K8S environment on their machine (Docker CE)
 - Multiple operate support in same farm
 - Docker native command (docker run, docker ps etc)
 - Kuberntest native command (kubectl create, kubectl get pods etc)
 - Compose support both orchestrator
 - Big community of sharing in container world

Benefit of K8S in docker platform



```
praparns-MacBook-Pro% kubectl get pods --namespace=kube-system
No resources found.
praparns-MacBook-Pro% kubectl port-forward kubernetes-dashboard-7798c48646-hvpdb 8443:8443 --namespace=kube-system
Forwarding from 127.0.0.1:8443 -> 8443
```



```
praparns-MBP% docker stack deploy --compose-file compose.example.yml k8stack
Stack k8stack was created
Waiting for the stack to be stable and running...
- Service web has one container running
- Service words has one container running
Stack k8stack is stable and running
praparns-MBP% docker stack ls
NAME          SERVICES
k8stack        3
praparns-MBP% kubectl get pods
NAME          READY   STATUS    RESTARTS   AGE
ce-4dd05445b7-lrxvr  1/1   Running   0          42s
web-564bb6fbcb-21s4v  1/1   Running   0          42s
words-abffdbb44c-4fm6  1/1   Running   0          42s
words-abffdbb44c-j8r9  1/1   Running   0          42s
words-abffdbb44c-rmg08 1/1   Running   0          42s
words-abffdbb44c-wqhz  1/1   Running   0          42s
words-abffdbb44c-wxstf 1/1   Running   0          42s
```

Demo Session





ເໜີງ ລັສຊົ່ງ
ໄມ້ໂລນ ກົດໆໄດ້

By Praparn Luengphoonlap
Email: eva10409@gmail.com

Q&A