



By Praparn Luengphoonlap
Email: eva10409@gmail.com

Docker 101

Docker: 101



Outline

- Docker principle
- Platform of Docker
- Docker Use case: Nodejs
- Docker Network Principle
- Docker Use case: Reverse Proxy with NGINX
- Docker Use case: LINE Notify
- The Brief history of Docker & K8S
- Kubernetes & Swarm Demo

Prerequisite

- Windows (64 bit) / Mac / Linux (64 bit) machine (ubuntu / alpine prefer)
- Tool for editor (vscode etc)
- Tool for shell (putty / terminal etc)
- Tool for transfer file (winscp / scp)
- Basic understand for linux operate
- Basic text editor skill (vim prefer) and linux structure
- Internet for download / upload image

Lab Resource

- Repository for lab

The screenshot shows a web interface for managing Docker repositories. At the top, there is a dark header bar with a logo, 'Explore', 'Help', a search bar containing 'labdocker', and 'Sign up / Log in' buttons. Below the header, the page title 'Repositories (7)' is displayed. A dropdown menu is open, showing the option 'All'. The main content area lists seven repositories in a grid:

Repository	Status	Stars	Pulls	Details
labdocker/alpineweb	public	0	31	DETAILS
labdocker/nginx	public	0	16	DETAILS
labdocker/alpine	public	0	7	DETAILS
...
...
...

Docker: 101



Lab Resource

- Software in lab

The screenshot shows a Windows file explorer window with the path: Computer > DATA (D:) > Docker_Nodejs > Workshop > Workshop_1-11_Registry. Inside this folder, there are three files: 'instruction' (Text Document, 4 KB), 'labdocker.com' (Security Certificate, 2 KB), and 'labdocker.com.key' (KEY File, 2 KB). Below the file explorer is a Notepad window titled 'instruction - Notepad'. The content of the Notepad is as follows:

Link for download:
<https://www.docker.com/products/docker-toolbox>

1. See PDF document for detail to install
2. After finished then run below command for check docker-machine (Command prompt)
 2.1 docker-machine --version ==> check version of docker machine & readiness
 2.2 docker-machine create --driver virtualbox labdocker ==> create new docker-machine for lab
 2.3 docker-machine ls ==> check ip address of new docker-machine

*Remark: default username/password for access docker-machine is docker/tcuser

3. SSH to docker-machine (labdocker)
 3.1 docker-machine ssh labdocker ==> default ssh via command prompt
 3.2 access via putty(windows) to ip address
 3.3 access via shell (mac)
 - Shell ==> New Remote Connection (Service: ssh)

4. Incase Upgrade docker-machine. Please check PDF document (Upgrade_Docker_1.10.pdf)

Lab Resource

- Download on Google Drive
 - <https://goo.gl/Sb16HT>
- Download on GitHub
 - git@github.com:praparn/docker101.git

The screenshot shows a GitHub repository page for 'praparn / docker101'. The repository has 1 commit, 1 branch, 0 releases, and 1 contributor. The latest commit was made a minute ago. The repository contains files: Workshop_1_Download_Install, Workshop_2_Docker_for_Nodejs, Workshop_3_Docker_for_Linenotify, Workshop_4_K8S, and README.md.

File	Created	Last Updated
Workshop_1_Download_Install	20180409230251	44 seconds ago
Workshop_2_Docker_for_Nodejs	20180409230251	44 seconds ago
Workshop_3_Docker_for_Linenotify	20180409230251	44 seconds ago
Workshop_4_K8S	20180409230251	44 seconds ago
README.md	20180409230251	44 seconds ago

Docker: 101



Workshop 1: Download & Install



The screenshot shows the Docker website at <https://www.docker.com/products/docker-toolbox>. The page features the Docker logo and navigation links for Docs, Support, Training, Tech Blog, Blog, Docker Hub, Why Docker?, Products, Partners, Community, Company, Careers, and Open Source. A prominent section titled "Docker Toolbox" includes a cartoon illustration of a red toolbox overflowing with various Docker-related icons like a whale, a brain, and a magnifying glass. Below the illustration, a text block states: "The Docker Toolbox is an installer to quickly and easily install and setup a Docker environment on your computer." Two download buttons are provided: one for Mac (Apple icon) and one for Windows (Windows icon).

- <https://www.docker.com/products/docker-toolbox>
- Install the software
 - Windows : Windows_Install_Docker.pdf
 - Mac OS: Mac_Install_Docker.pdf

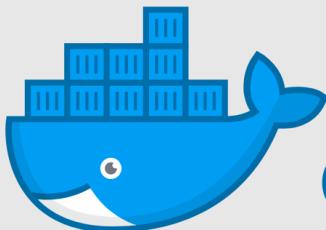
WorkShop

labs.play-with-docker.com

Welcome!

Before starting we need to verify you are a human

✓ ตั้งใจให้ไปง่ายๆ ก็ได้นะนี่! reCAPTCHA



docker

Waiting for labs.play-with-docker.com...
Kubernetes_Training.pptx
Cancelled

03:57:34
CLOSE SESSION
Instances  
+ ADD NEW INSTANCE

eb5f5443_node1

eb5f5443_node1

eb5f5443-b72c-4aa3-b785-5e740f4da0ea#eb5f5443_node1

Kubernetes vagrant Mesos Vue docker NGINX Taiwan Kubernetes PWA_Progressive_V

eb5f5443_node1

IP: 10.0.50.3

Memory: 2.19% (89.52MiB / 3.996GiB)

CPU: 0.19%

DELETE

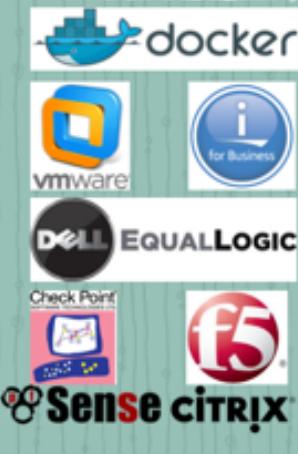
```
#####
# WARNING!!!!
# This is a sandbox environment. Using personal credentials
# is HIGHLY! discouraged. Any consequences of doing so are
# completely the user's responsibilites.
#
# The PWD team.
#####
[node1] (local) root@10.0.50.3 ~
$ docker ps
CONTAINER ID        IMAGE               COMMAND             CREATED            STATUS              NAMES
[node1] (local)   root@10.0.50.3 ~
$
```



Docker: 101

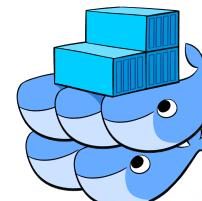


Senior Infrastructure / System Engineer
Compute, Network & Storage System (CNS)
Infrastructure Enterprise Equipment
Virtual Machine for Open System in Prd
Core Application on IBM AS/400 System
Instructor / Consult for Docker's Solution



kubernetes

NGINX



Docker: 101



Landscape of the world now

Cloud Native Landscape

v2.1

See the interactive landscape at landscape.cncf.io

Greyed logos are not open source

App Definition & Development



Streaming



Source Code Management



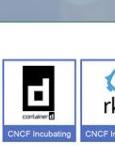
Continuous Integration / Continuous Delivery (CI/CD)



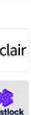
Orchestration & Management



Runtime



Provisioning



Cloud



Public

Private



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.



CLOUD NATIVE
COMPUTING FOUNDATION

Redpoint Amplify PARTNERS

Kubernetes Certified Service Provider

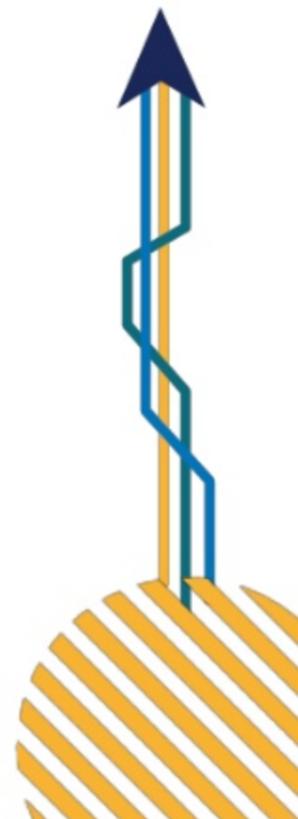
Special



Landscape of the world now

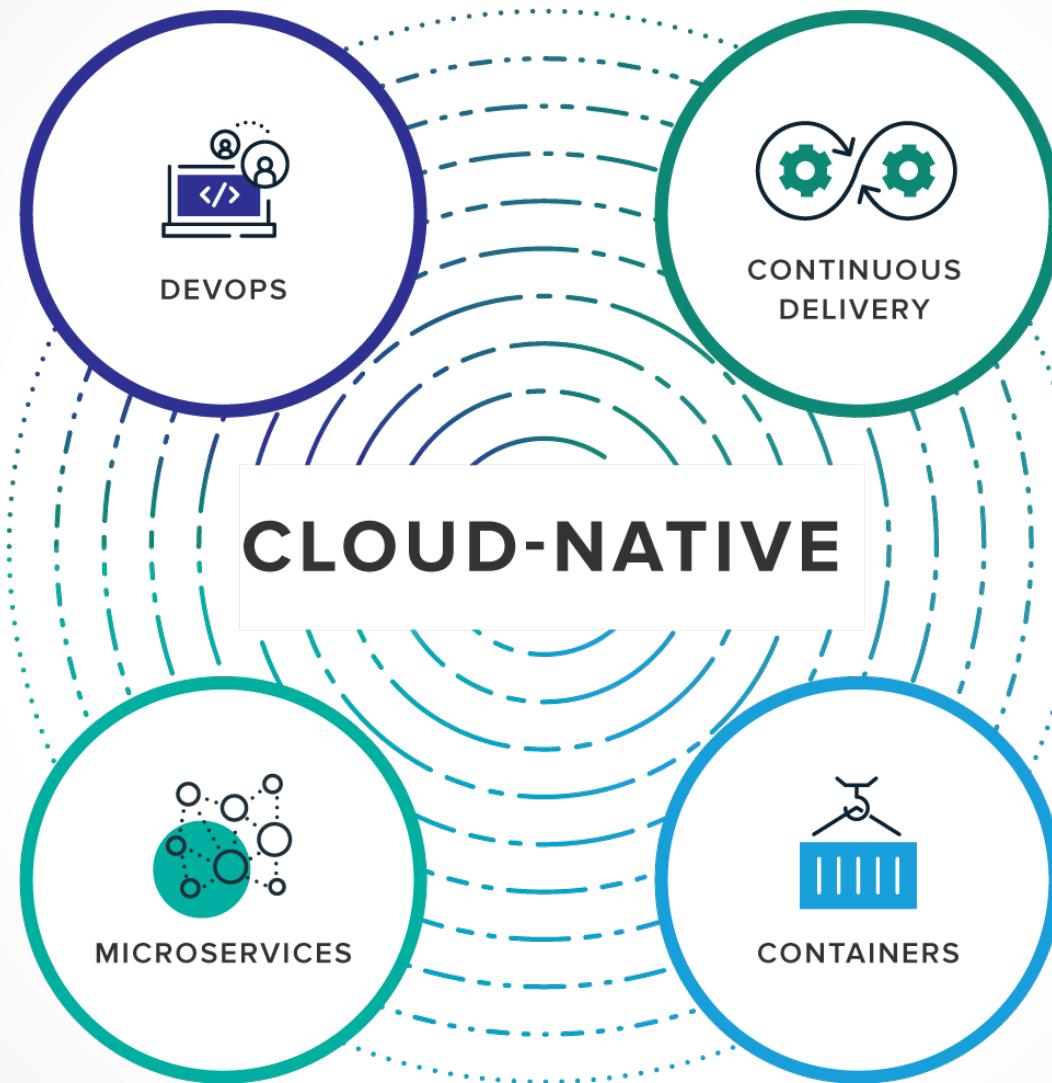


Imagine how the
world should work



 dockercon¹⁷ EU

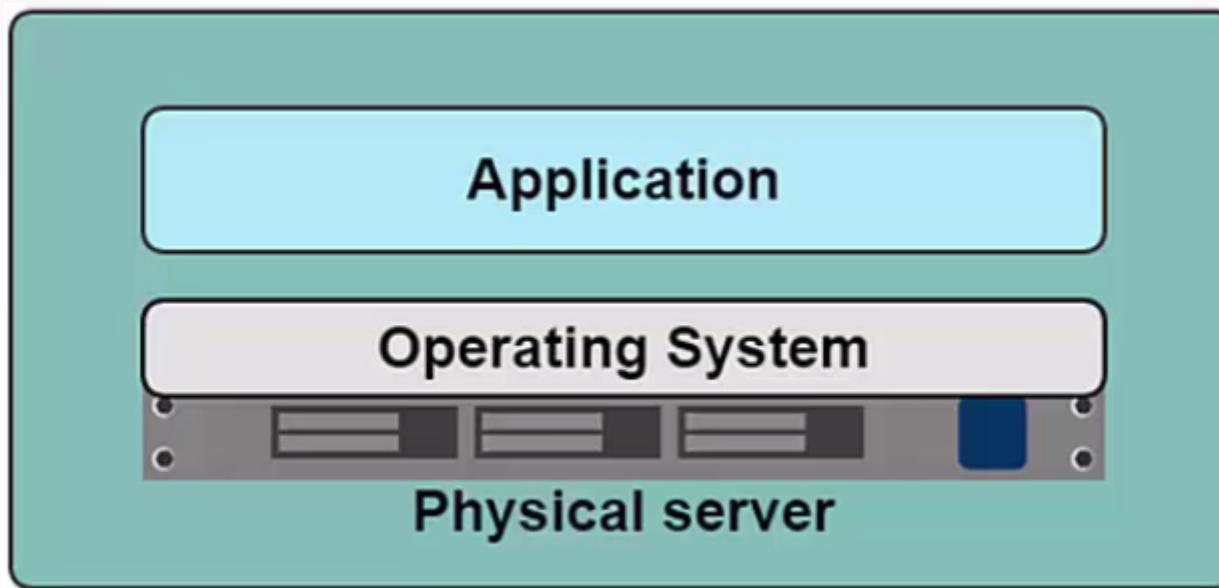
Landscape of the world now



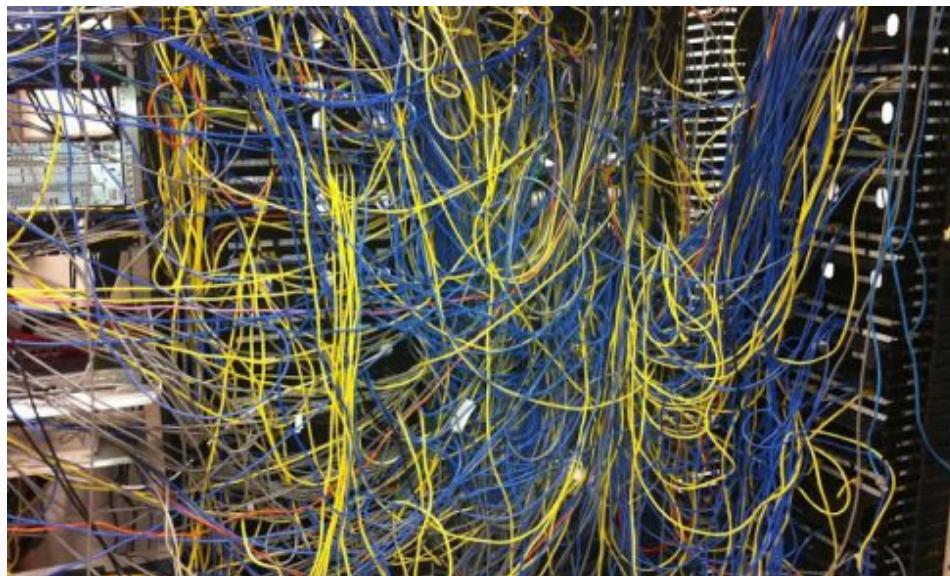
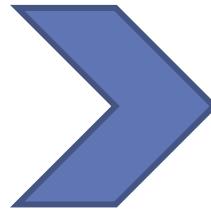
Docker Principle

• • •

What is docker ?



Existing Technology

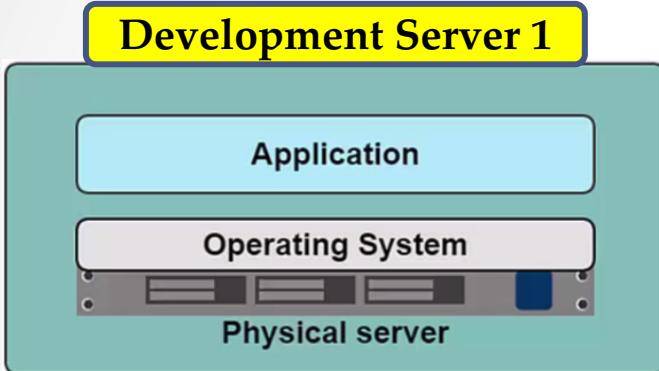


Docker: 101

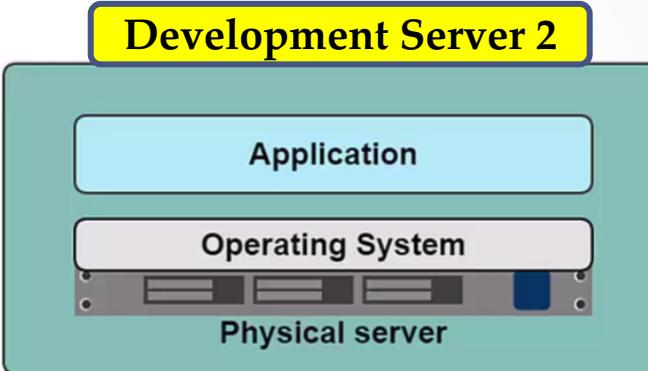


Existing Technology

- Development Environment (Mix everything possible)



- Apache 2.20 Web Server
 - PHP 5.5 Engine
 - Laravel 4.1 Framework
 - MySQL 5.1
 - Etc
- (All-in-One Server)

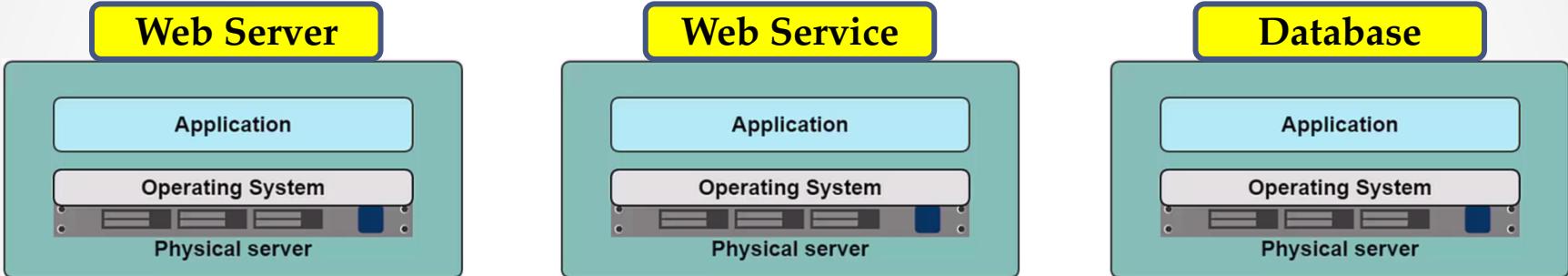


- IIS 8.0
- .Net Framework 3.5
- ASP.NET
- Etc

- Need concern about conflict component
- Survive among legacy dependency
- Lack for environment for fulfill develop & test (module test/integrate test/ UAT test / MOT test etc)
- Unexpected software conflict frequently occur
- Incomplete software's integrated test

Existing Technology

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



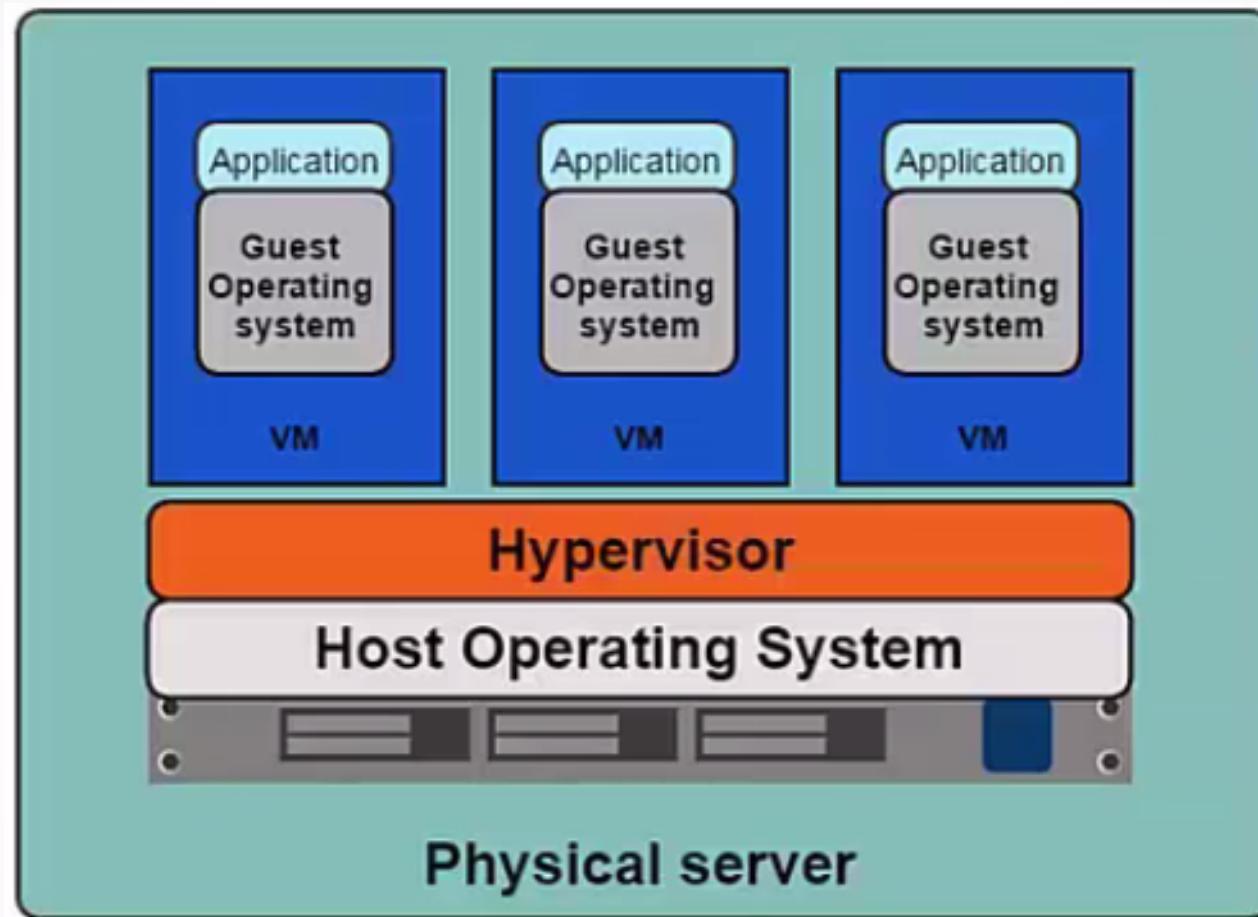
- Apache 2.20 Web Server
- PHP 5.5 Engine
- Laravel 4.1 Framework

- IIS 8
- .Net Framework 3.5

- MariaDB 5.1

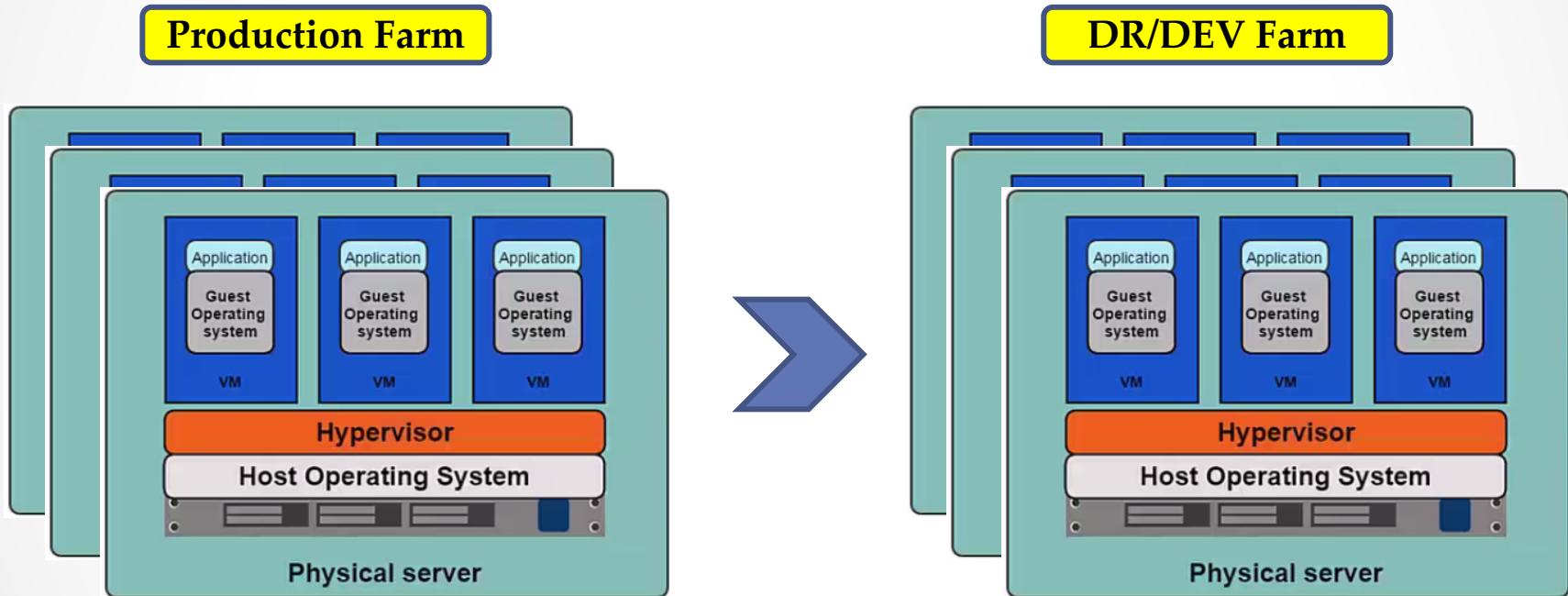
- Day 2: Application 2: Need to implement
 - Need PHP 7.0 ?
 - MariaDB 10.1.14 (Need search feature on 10.1)
- Problem ?
 - Possible to upgrade PHP to 7.0 ? / How to test existing application ?
 - What effect to MariaDB upgrade ?

What is docker ?



Existing Technology

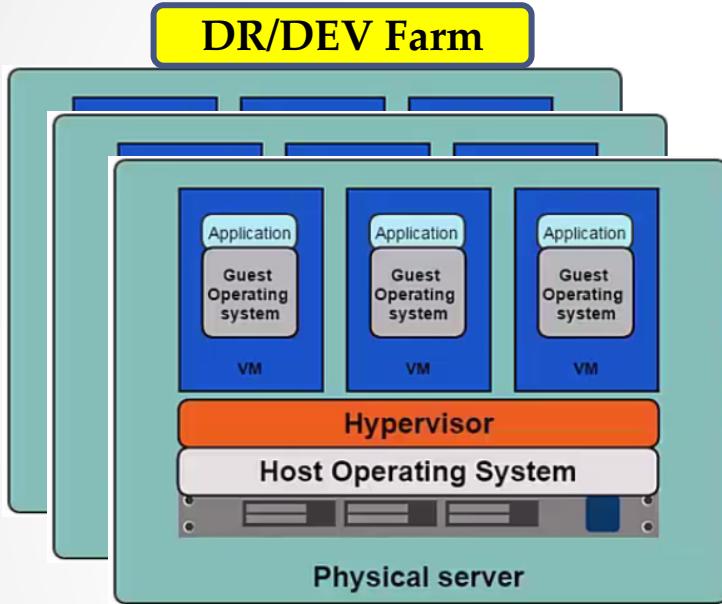
- 1 Physical server : 1 – N VMWare (&OS)
- Virtualize Hardware (CPU, Memory, Disk, Network etc)



- Kernel-base virtual machine (KVM), Vmware, Virtualbox etc

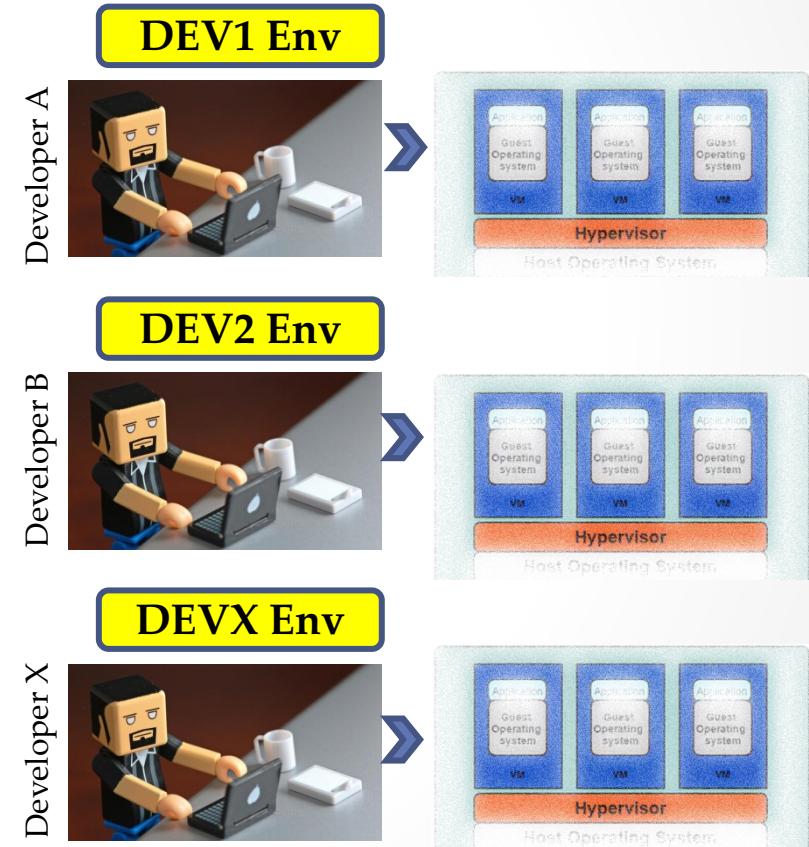
Existing Technology

- Development Environment (Clone from Production)



Virtual Machine reach limit:

- Resource insufficient (Almost from disk)
- Conflict version
- Huge of disk duplication
- Conflict & dependency still exist



Existing Technology

- Production Environment
- Day 1: Application 1: Implement



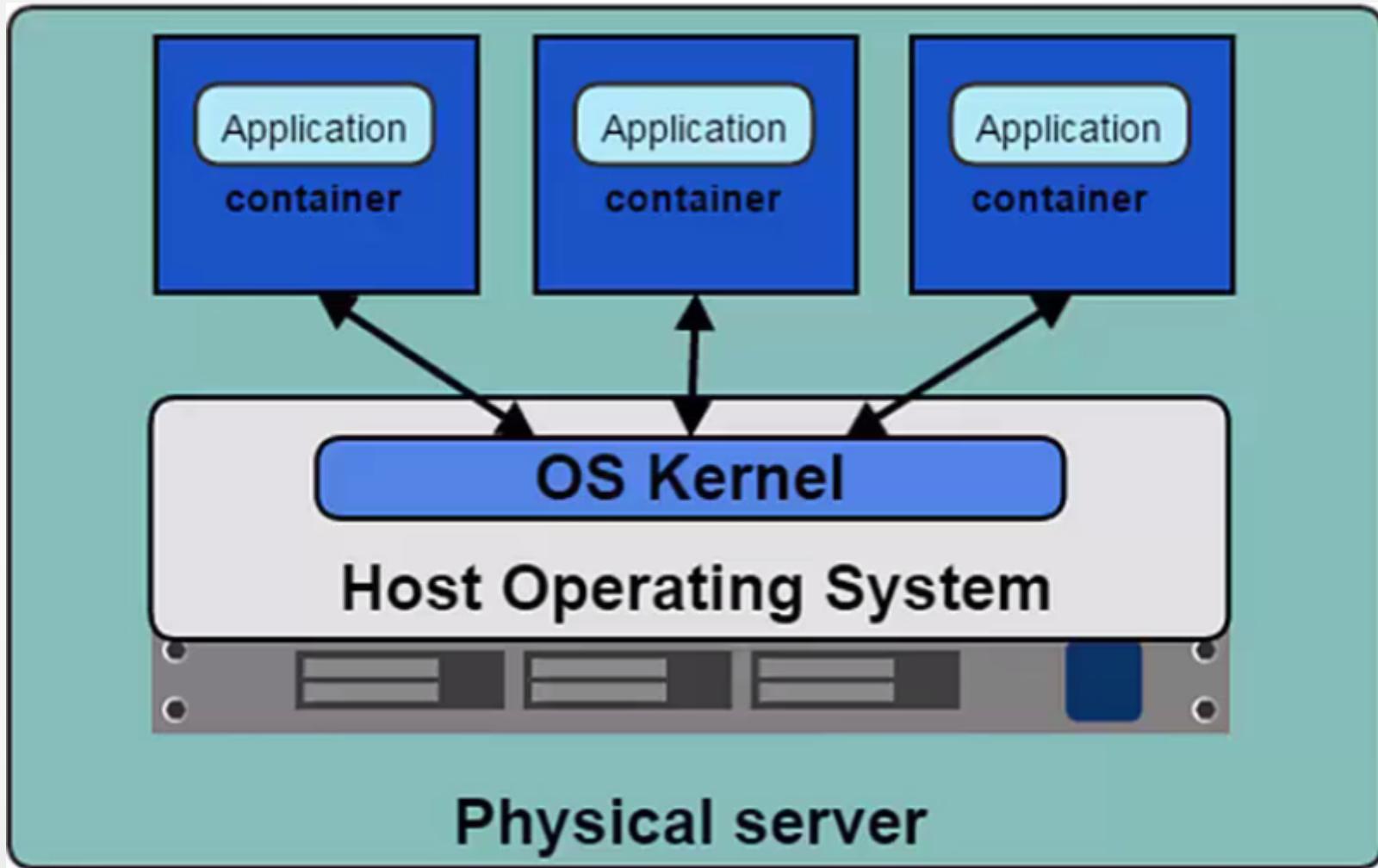
- Apache 2.20 Web Server
- PHP 5.5 Engine
- Laravel 4.1 Framework

- IIS 8
- .Net FrameWork 3.5

- MariaDB 5.1

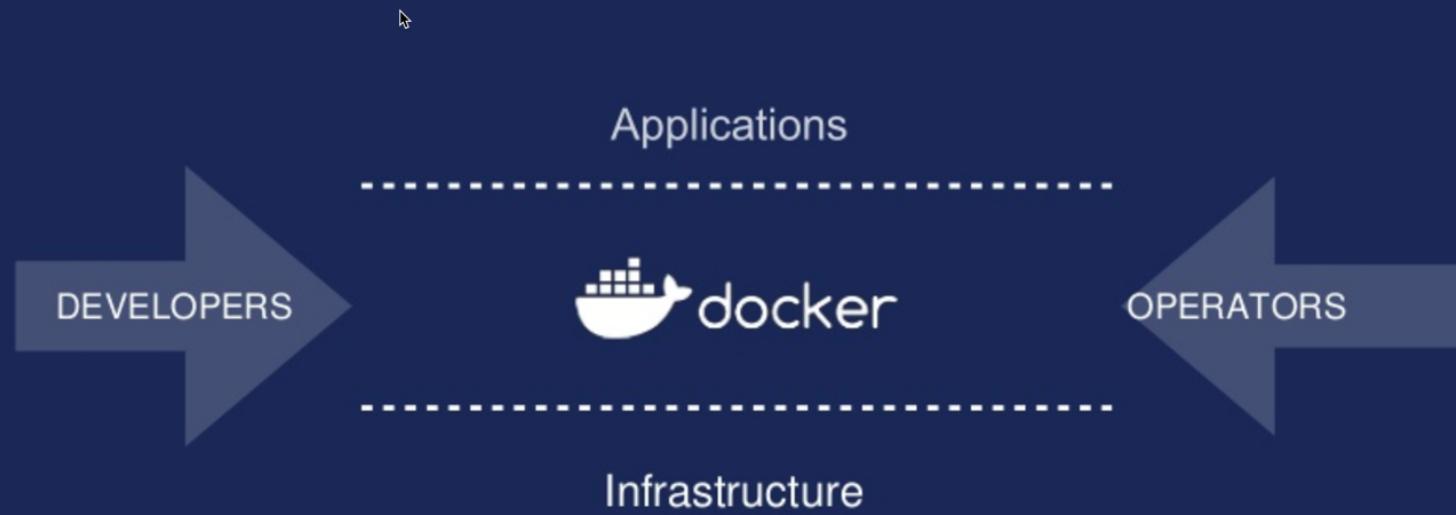
- Day 2: Application 2: Need to implement
 - Need PHP 7.0 ?
 - MariaDB 10.1.14 (Need search feature on 10.1)
- So... The problem still exist.

What is docker ?



What is docker ?

The Docker Platform in a nutshell



What is docker ?

Core Principles of the Docker Platform

INDEPENDENCE



OPENNESS



SIMPLICITY



 dockercon EU

What is docker ?

- Docker คือ open platform solution ที่ทำงานภายใต้คอนเซ็ปต์ของ container virtualize technology (operating -system –level virtualization)
- ผู้ใช้สามารถสร้างสภาพแวดล้อมเพื่อใช้ในพัฒนาโปรแกรมและส่งมอบเพื่อใช้งานในสภาพเดียวกัน
- Build, Ship, Run
- เหมาะสำหรับ Developer, DevOps, Architecture, Engineer
- เขียนด้วยภาษา Go (1.8.3) (Now)
- รองรับการติดตั้งบนบนลินุกซ์ 64 บิต (kernel 3.1.0) (Official)
 - Ubuntu
 - Debian
 - Red Hat Enterprise Linux
 - CentOS
 - Fedora
 - Microsoft Windows Server 2016
 - Suse Linux Enterprise Server
 - Raspberry PI

What is docker ?

Desktop

Platform	Docker CE x86_64	Docker CE ARM	Docker EE
Docker for Mac (macOS)	✓		
Docker for Windows (Microsoft Windows 10)	✓		

Cloud

Platform	Docker CE x86_64	Docker CE ARM	Docker EE
Amazon Web Services	✓		✓
Microsoft Azure	✓		✓

See also [Docker Cloud](#) for setup instructions for Digital Ocean, Packet, SoftLink, or Bring Your Own Cloud.

Server

Platform	Docker CE x86_64	Docker CE ARM	Docker CE System Z (s390x)	Docker EE
CentOS	✓			✓
Debian	✓	✓		
Fedora	✓			
Microsoft Windows Server 2016				✓
Oracle Linux				✓
Red Hat Enterprise Linux				✓
SUSE Linux Enterprise Server				✓
Ubuntu	✓	✓	✓	✓

What is docker ?



Lifecycle

Squaring the circle: Faster releases and better stability



Docker EE Availability

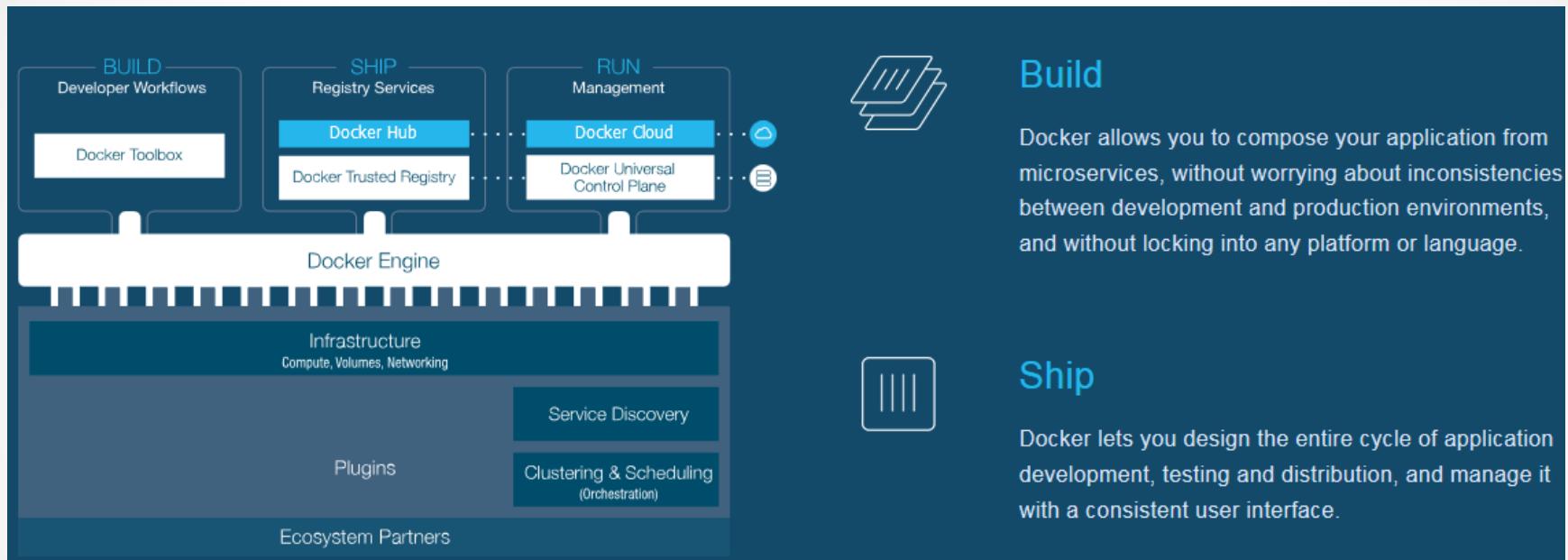


Ref: <https://blog.docker.com/2017/03/docker-online-meetup-recap-docker-enterprise-edition-ee-community-edition-ce/>

Docker: 101



What is docker ?



Operating-system-level virtualization

Mechanism	Operating system	License	Available since/between	Features									
				File system isolation	Copy on Write	Disk quotas	I/O rate limiting	Memory limits	CPU quotas	Network isolation	Nested virtualization	Partition checkpointing and live migration	Root privilege isolation
chroot	most UNIX-like operating systems	varies by operating system	1982	Partial ^[3]	No	No	No	No	No	No	Yes	No	No
Docker Linux-VServer (security context)	Linux ^[7]	Apache License 2.0	2013	Yes	Yes	Not directly	Not directly	Yes	Yes	Yes	Yes	No	No
		GNU GPLv2	2001	Yes	Yes	Yes	Yes ^[8]	Yes	Yes	Partial ^[9]	?	No	Partial ^[9]
lxc LXC	Linux	Apache License 2.0	2013	Yes	Yes	Yes	Yes ^[9]	Yes	Yes	Partial ^[9]	?	No	Partial ^[9]
		GNU GPLv2	2008	Yes ^[9]	Yes	Partial ^[9]	Partial ^[9]	Yes	Yes	Yes	Yes	No	Yes ^[9]
LXD	Linux	Apache License 2.0	2015	Yes	Yes	Partial(see LXC)	Partial(see LXC)	Yes	Yes	Yes	Yes	Partial ^[9]	Yes
OpenVZ	Linux	GNU GPLv2	2005	Yes	No	Yes	Yes ^[10]	Yes	Yes	Yes ^[10]	Partial ^[11]	Yes	Yes ^[10]
Virtuozzo	Linux, Windows	Proprietary	2000 ^[14]	Yes	Yes	Yes	Yes ^[11]	Yes	Yes	Yes ^[11]	Partial ^[12]	Yes	Yes
Solaris Containers (Zones)	illumos (OpenSolaris), Solaris	CDDL, Proprietary	2004	Yes	Yes (ZFS)	Yes	Partial ^[13]	Yes	Yes	Yes ^{[13][17][18]}	Partial ^[13]	Partial ^[13]	Yes ^[8]
FreeBSD jail	FreeBSD	BSD License	2000 ^[20]	Yes	Yes (ZFS)	Yes ^[14]	No	Yes ^[21]	Yes	Yes ^[22]	Yes	No	Yes ^[23]
sysjail	OpenBSD, NetBSD	BSD License	2006–2009 (As of March 3, 2009, it is no longer supported)	Yes	No	No	No	No	No	Yes	No	No	?
WPARs	AIX	Proprietary	2007	Yes	No	Yes	Yes	Yes	Yes	Yes ^[24]	No	Yes ^[25]	?
HP-UX Containers (SRP) ^[9]	HPUX	Proprietary	2007	Yes	No	Partial ^[15]	Yes	Yes	Yes	Yes	?	Yes	?
iCore Virtual Accounts	Windows XP	Proprietary/Freeware	2008	Yes	No	Yes	No	No	No	No	?	No	?
Sandboxie Spoon	Windows	Proprietary/Shareware	2004	Yes	Yes	Partial	No	No	No	Partial	No	No	Yes
		Proprietary	2012	Yes	Yes	No	No	No	No	Yes	No	No	Yes
VMware ThinApp	Windows	Proprietary	2008	Yes	Yes	No	No	No	No	Yes	No	No	Yes

Reference: https://en.wikipedia.org/wiki/Operating-system-level_virtualization

Docker History

- A dotCloud (PaaS provider) project
- Initial commit January 18, 2013
- Docker 0.1.0 released March 25, 2013
- 18,600+ github stars, 3800+ forks, 740 Contributors.... and continues
- dotCloud pivots to docker inc. October 29, 2013



A circular portrait of Solomon Hykes, a man with dark hair and a beard, wearing a black t-shirt. The portrait is set against a white background and has a gold-colored circular border.

Solomon Hykes

CTO and Founder

 dockercon¹⁷ EU

Docker: 101



← → ⌂ 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 มีบัญชีผู้ใช้อื่นแล้ว? 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

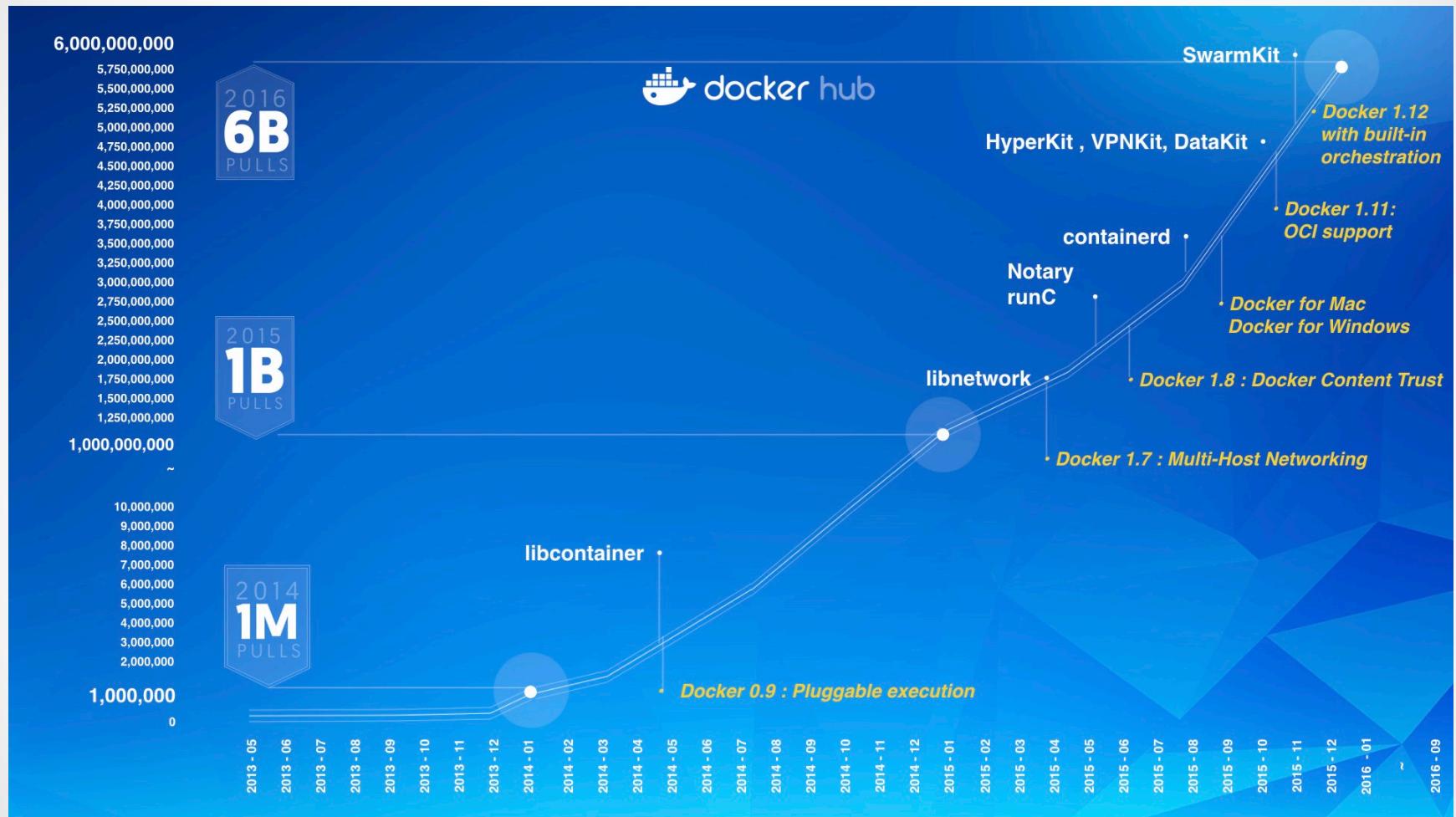
ทวีต ทวีตและการตอบกลับ สื่อ

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...

195 รูปภาพหรือวิดีโอ

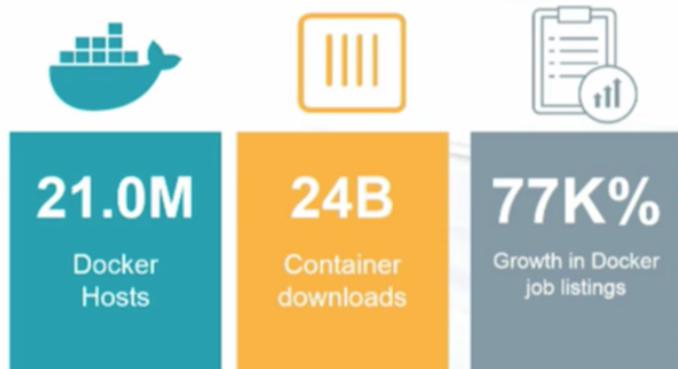
195 รูปภาพหรือวิดีโอ

Growth of Docker (2016 stats)



Growth of Docker (Now)

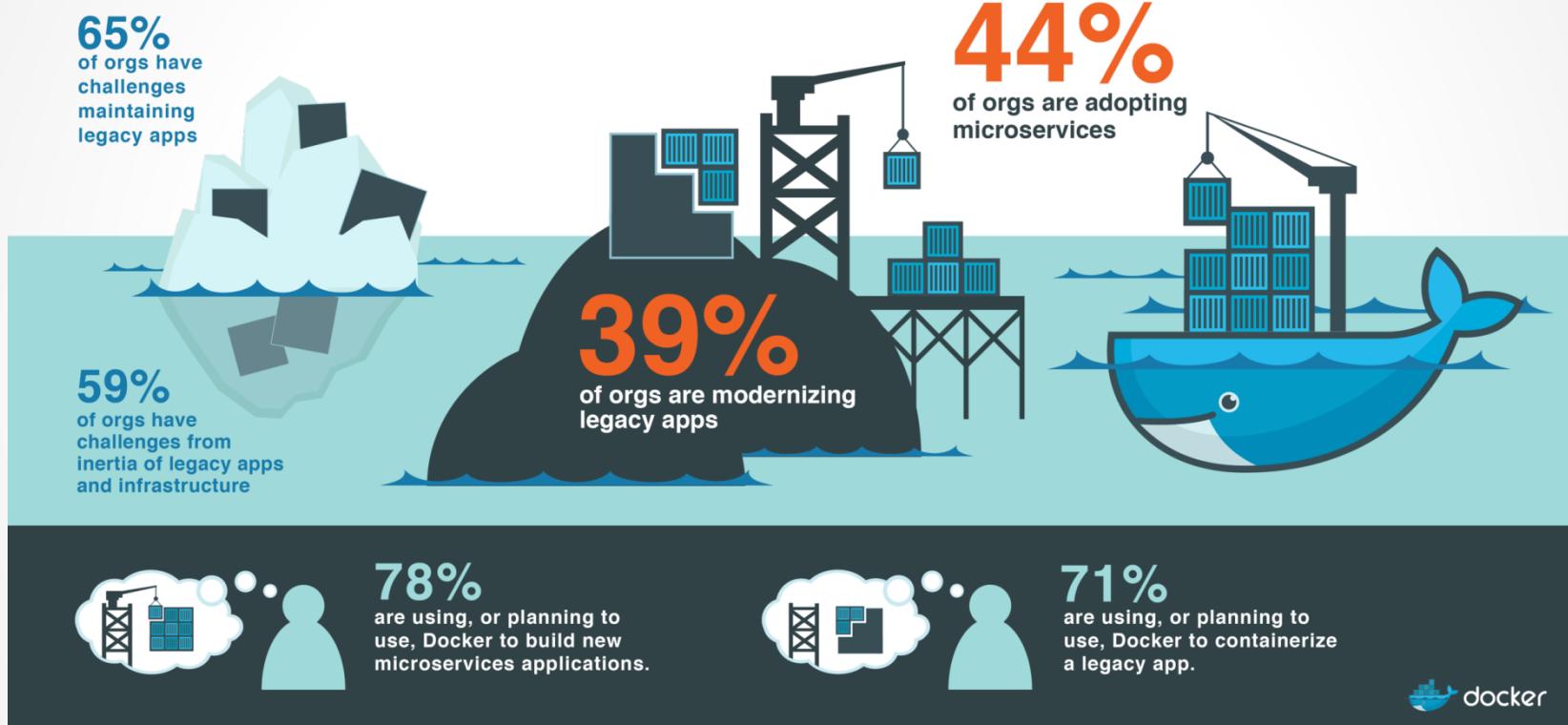
Docker Momentum



Industry Standards



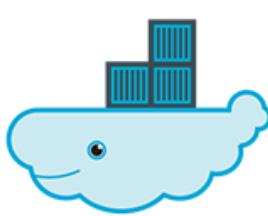
Trend of Modernize Application



Cloud Strategy Survey

80%

say Docker is part
of cloud strategy



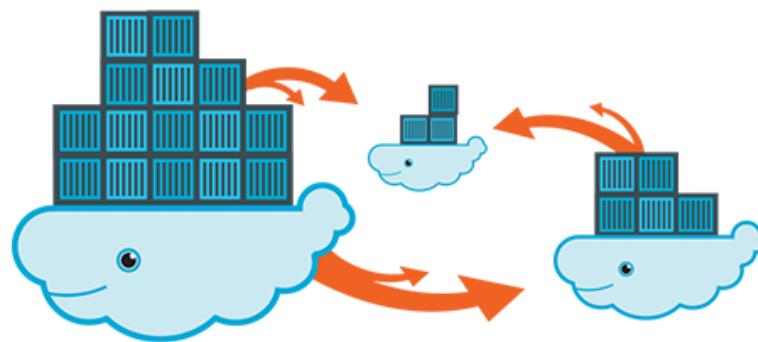
60%

plan to use Docker to
migrate workloads to cloud



41%

want application
portability across
environments



35+%

want to avoid
cloud vendor
lock-in



Docker in Thailand

- 3 Enterprise Communication on Implement Phase
- 2 Bank on R&D

Who use docker now ?

Breaking the
Pattern:
Docker
Enterprise
and
Microservices



Who use docker now ?

Visa Today

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

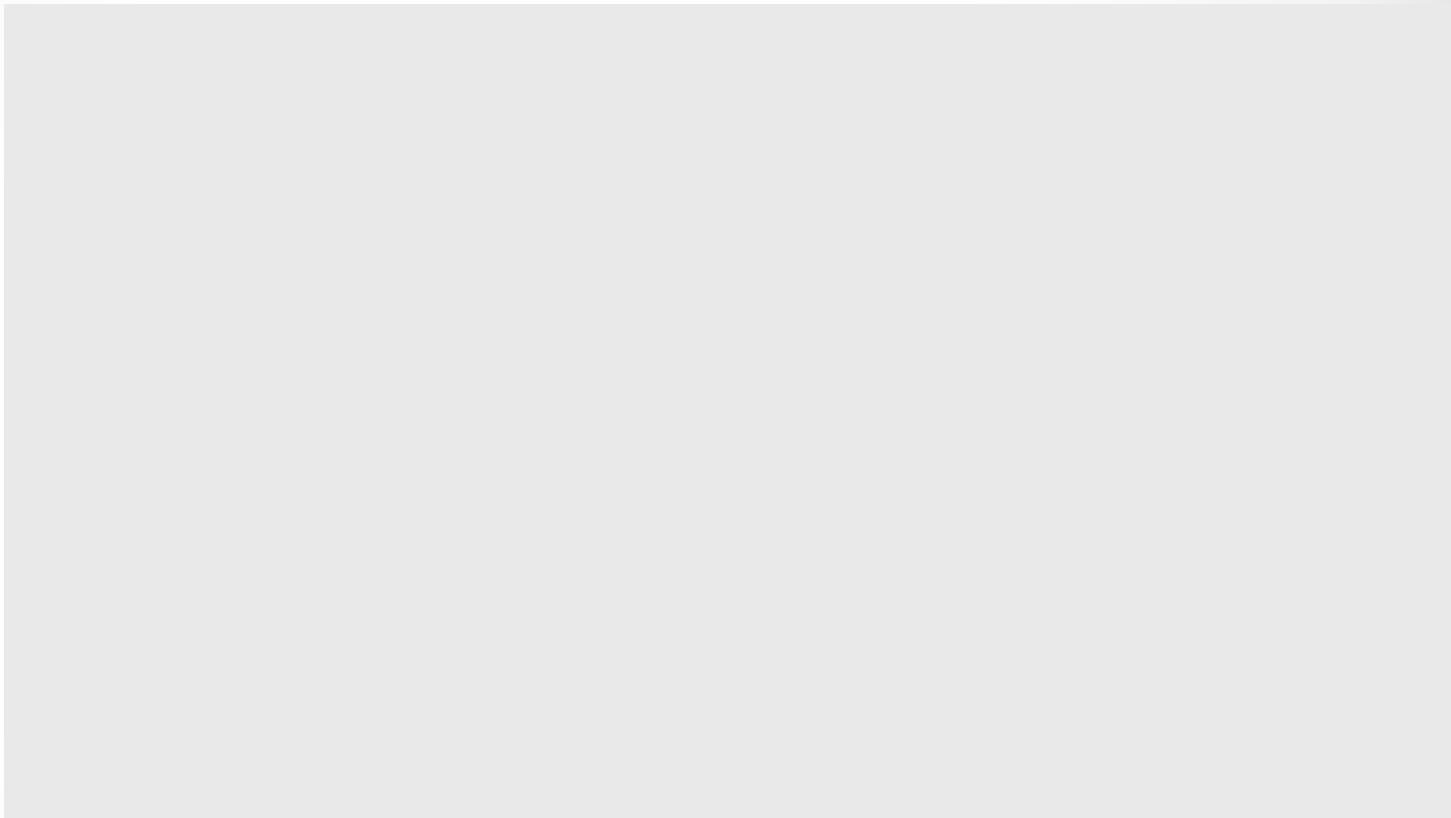


Who use docker now ?

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



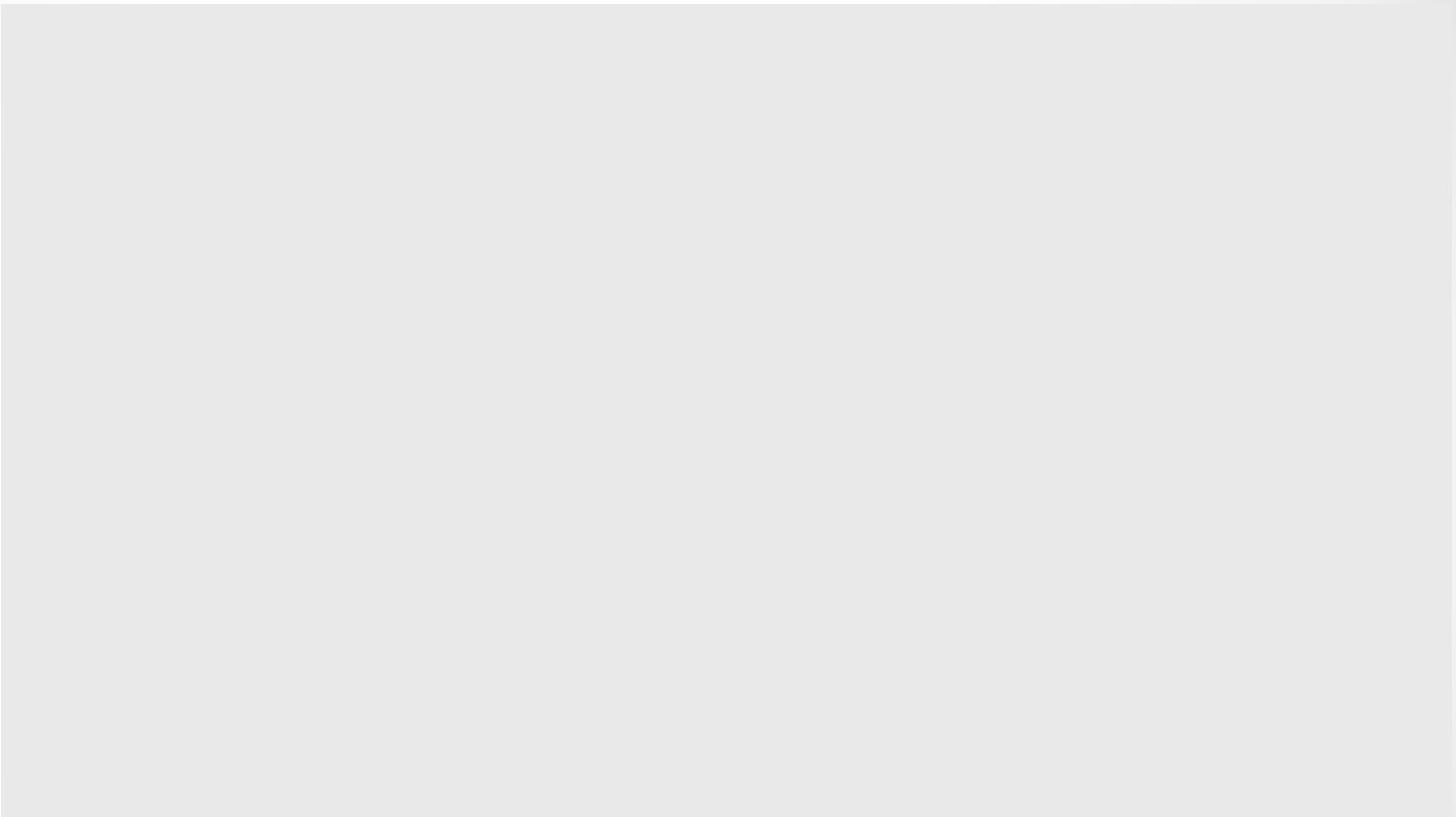
Who use docker now ?



Docker 101



Who use docker now ?



Docker 101



Who use docker now ?



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

Who use docker now ?

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

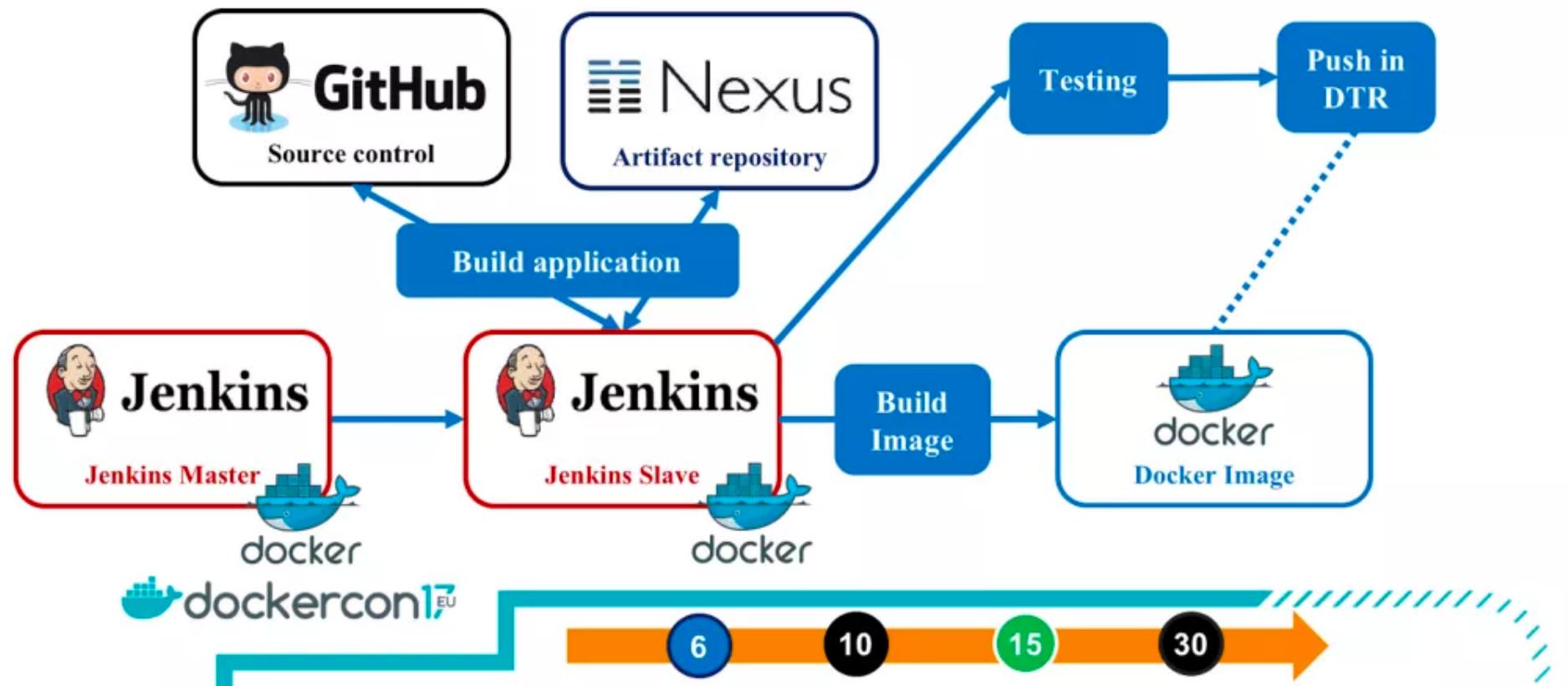
Developer
Freedom

for greenfield apps

Who use docker now ?



Level 1 - Build

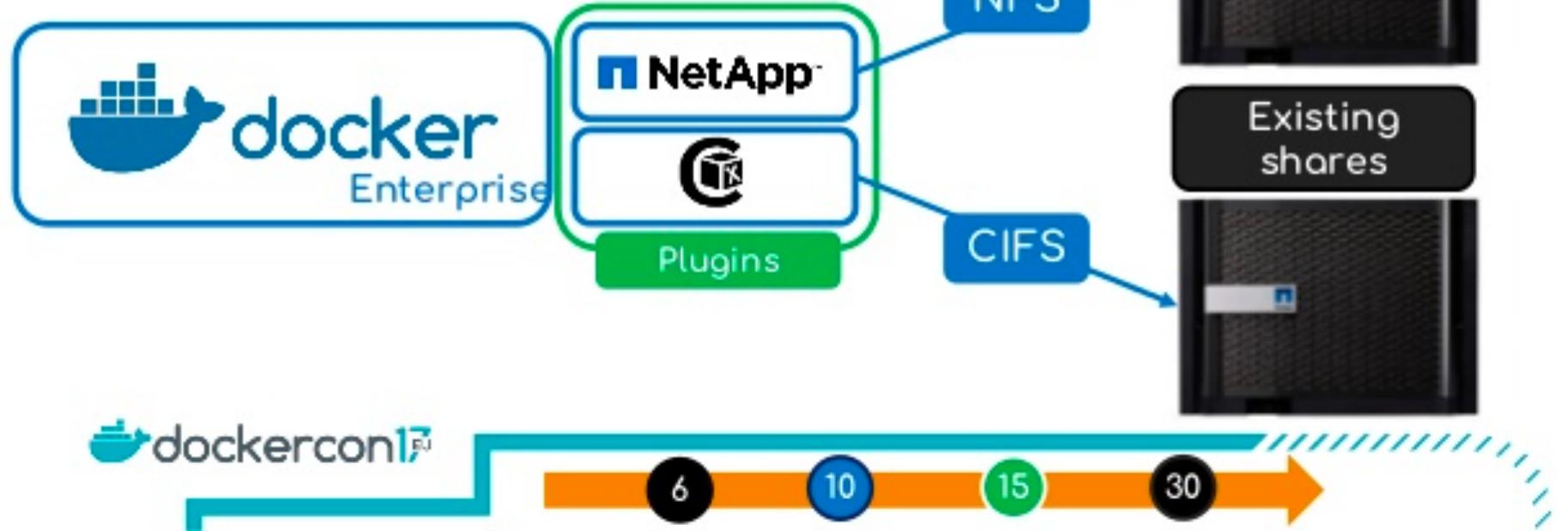


Who use docker now ?



Level 2 - Storage

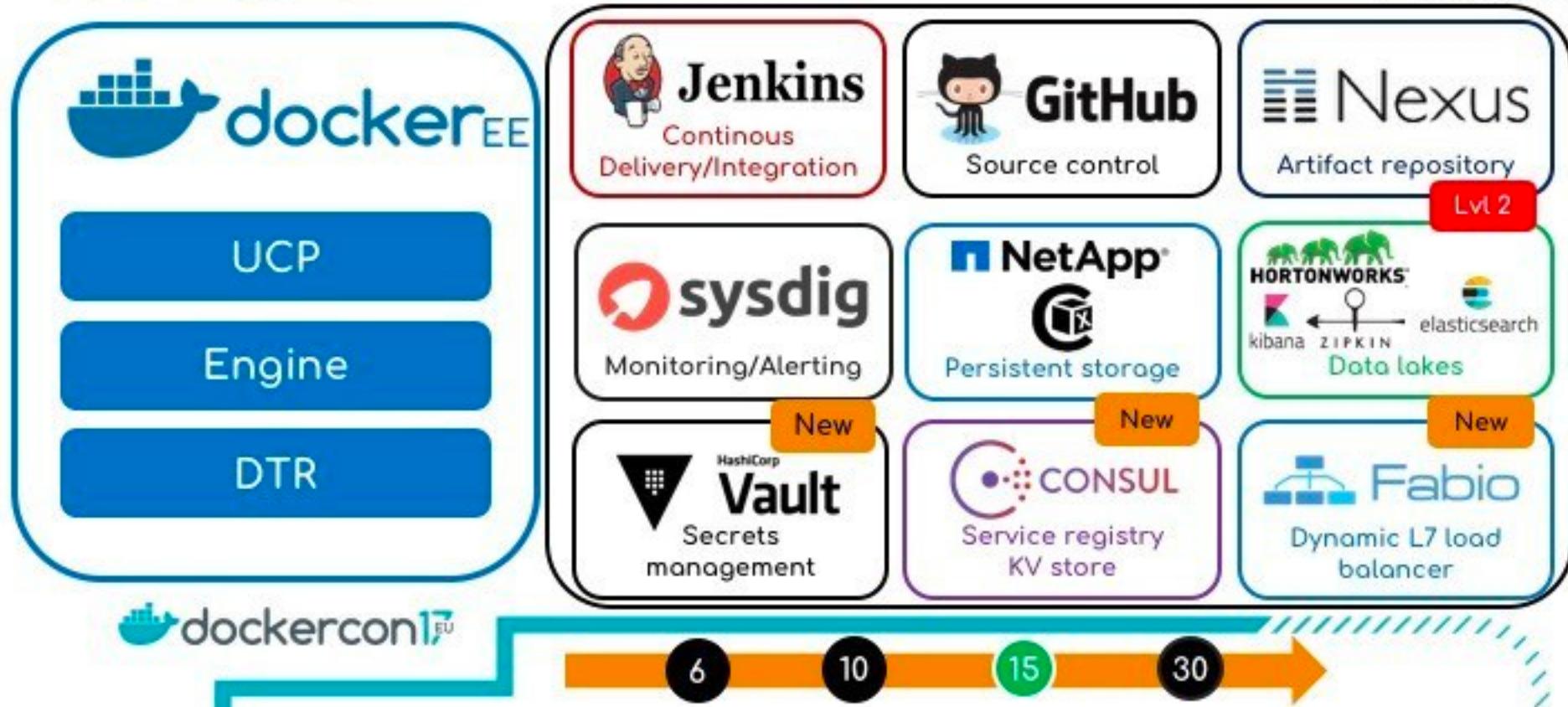
- Integrate with existing storage



Who use docker now ?

Level 3

SOCIETE
GENERALE



Who use docker now ?

About MetLife



Life



P&C



Annuity



Dental



Disability



Health



Legal

- Global Fortune 500® 128 Company
- 100 Million Customers
- \$500 Billion total assets under investment
- \$63 Billion in revenue in 2017



Even Old Elephants Can Dance

- Concept to production in 5 months
- The spark of innovation is spreading at MetLife
- Docker and DevOps change your culture
- Check out Tim Tyler's session

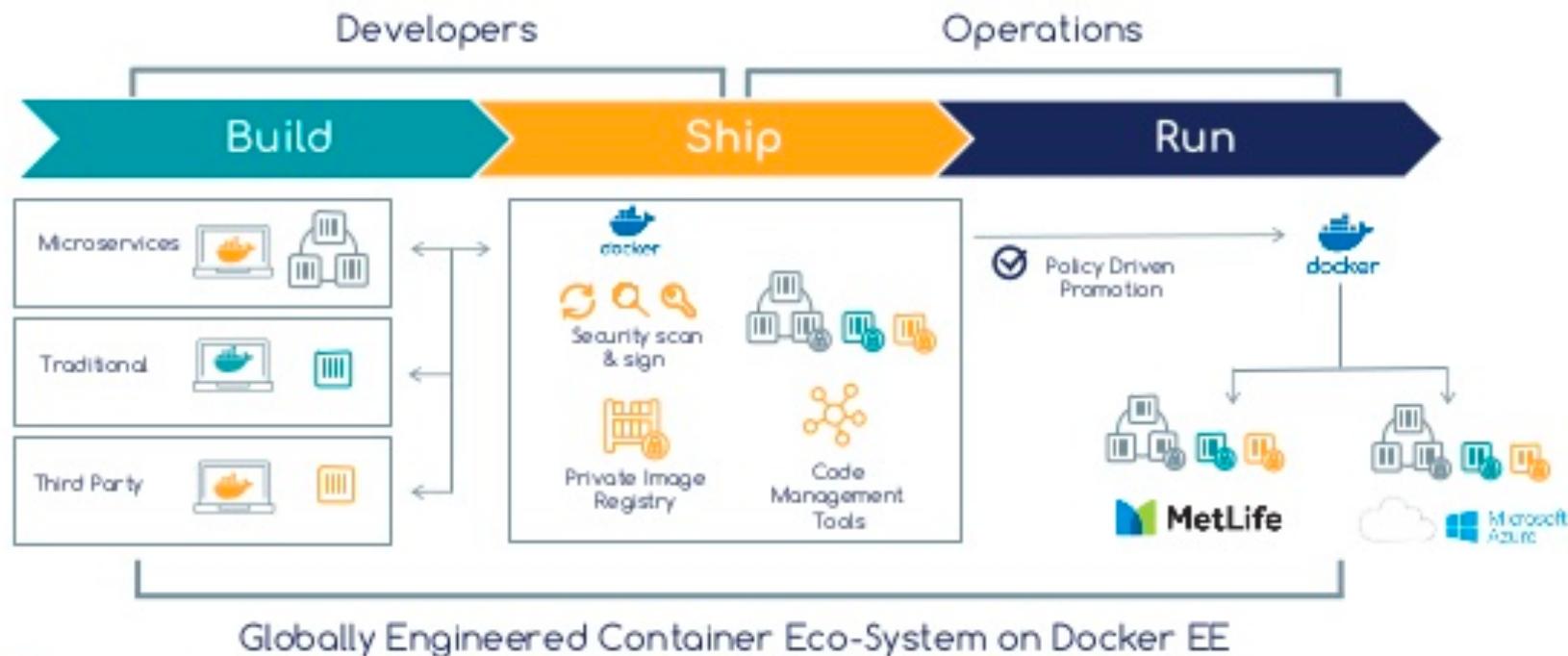
11:15 AM - Docker 0 to 60 in 5 Months:
How a Traditional Fortune 40 Company
Turns on a Dime

docker17
con



Who use docker now ?

The MetLife Platform



Who use docker now ?

US Infrastructure Reduction Forecast

593 Applications

10%

Of the total portfolio

-70%

VMs

+

-67%

Cores

+

10x

Average CPU Utilization

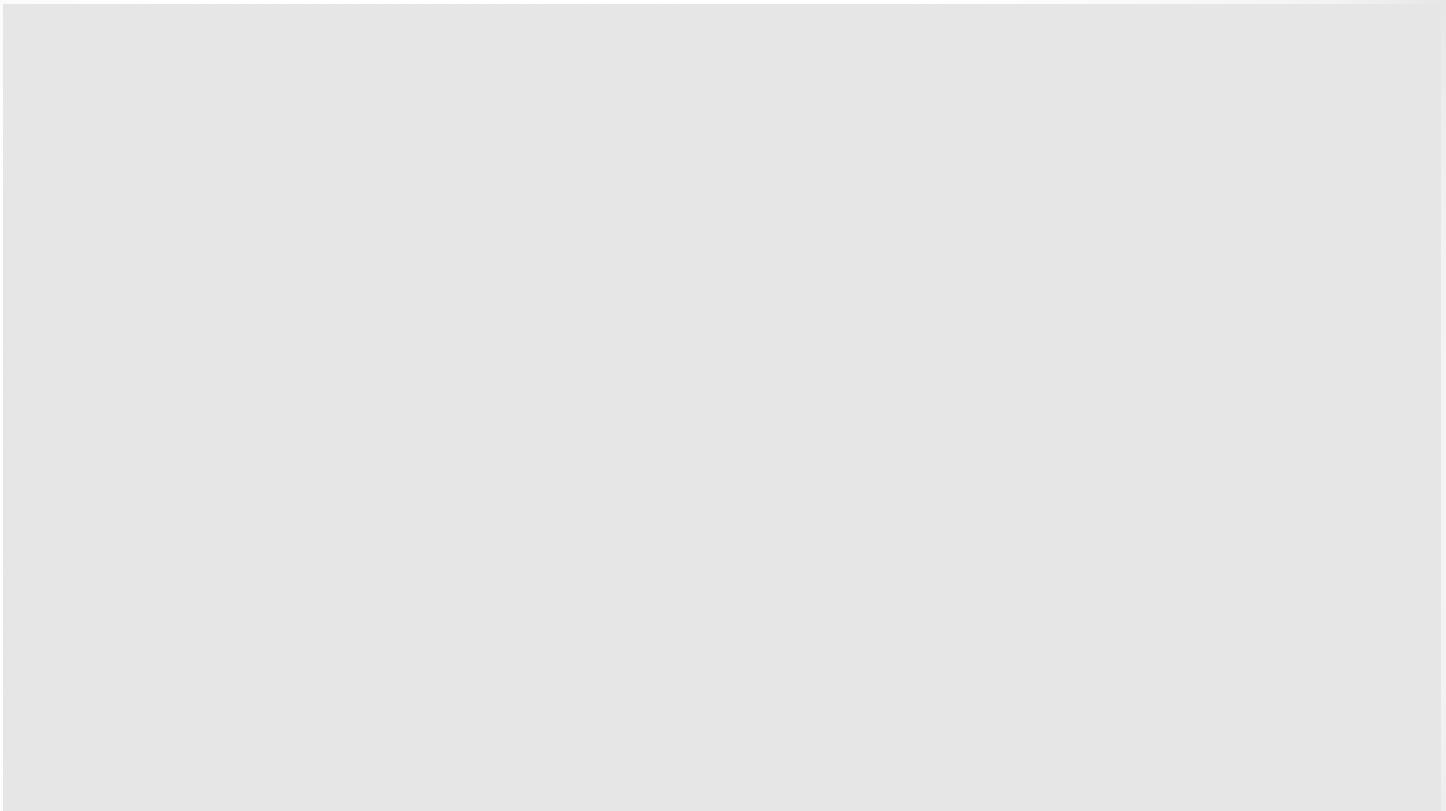
=

-66%

Cost Reduction



Who use docker now ?



U B E R

Docker 101



What Cool of Docker ?

- รวบรวมทุกสิ่งที่จำเป็นต้องใช้ในการรันโปรแกรมไว้ใน container (component, library etc)
- ขนาดไฟล์ container มีขนาดเล็กมาก (เทียบกับขนาดไฟล์ของ virtual machine หรือ os)
- มี overhead ในการรันโปรแกรมทรัพยากรต่ำ
- ลดระยะเวลาในการติดตั้งและทดสอบโปรแกรม
- ส่งมอบโปรแกรมไปทำงานบนเครื่องแม่ข่าย production ได้โดยไม่มีความจำเป็นต้องปรับแต่งระบบใหม่ (zero config)
- สามารถรันโปรแกรมได้บนเครื่องแม่ข่ายทุกๆระบบปฏิบัติการฯ ที่ติดตั้ง docker ได้
- สามารถ scale-out ได้ง่ายในอนาคต
- World open for docker !!!

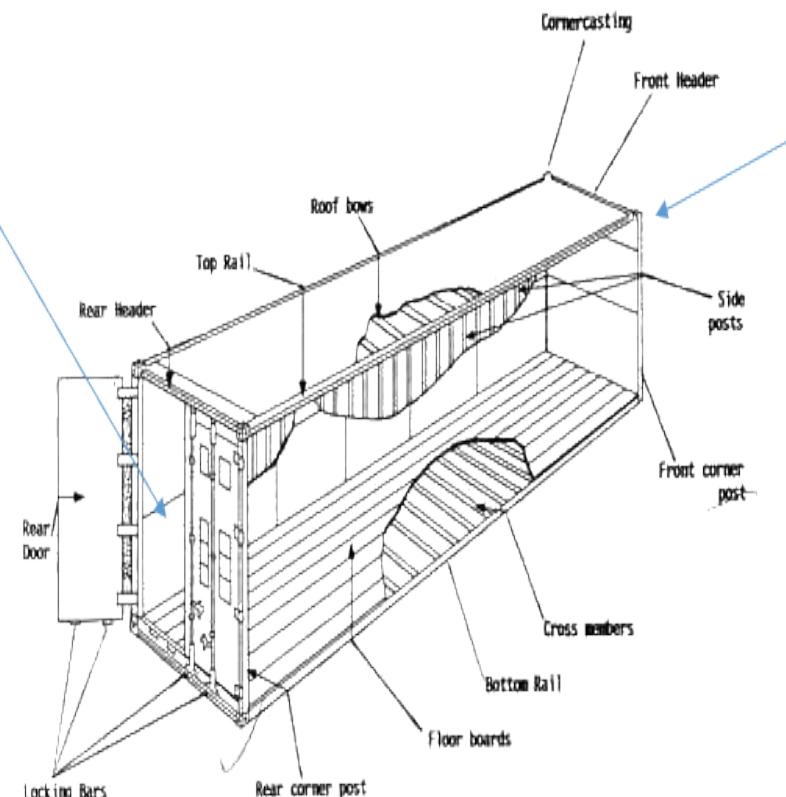
Separate of Concern

Dan the Developer

- Worries about what's "inside" the container
 - His code
 - His Libraries
 - His Package Manager
 - His Apps
 - His Data
- All Linux servers look the same

Oscar the Ops Guy

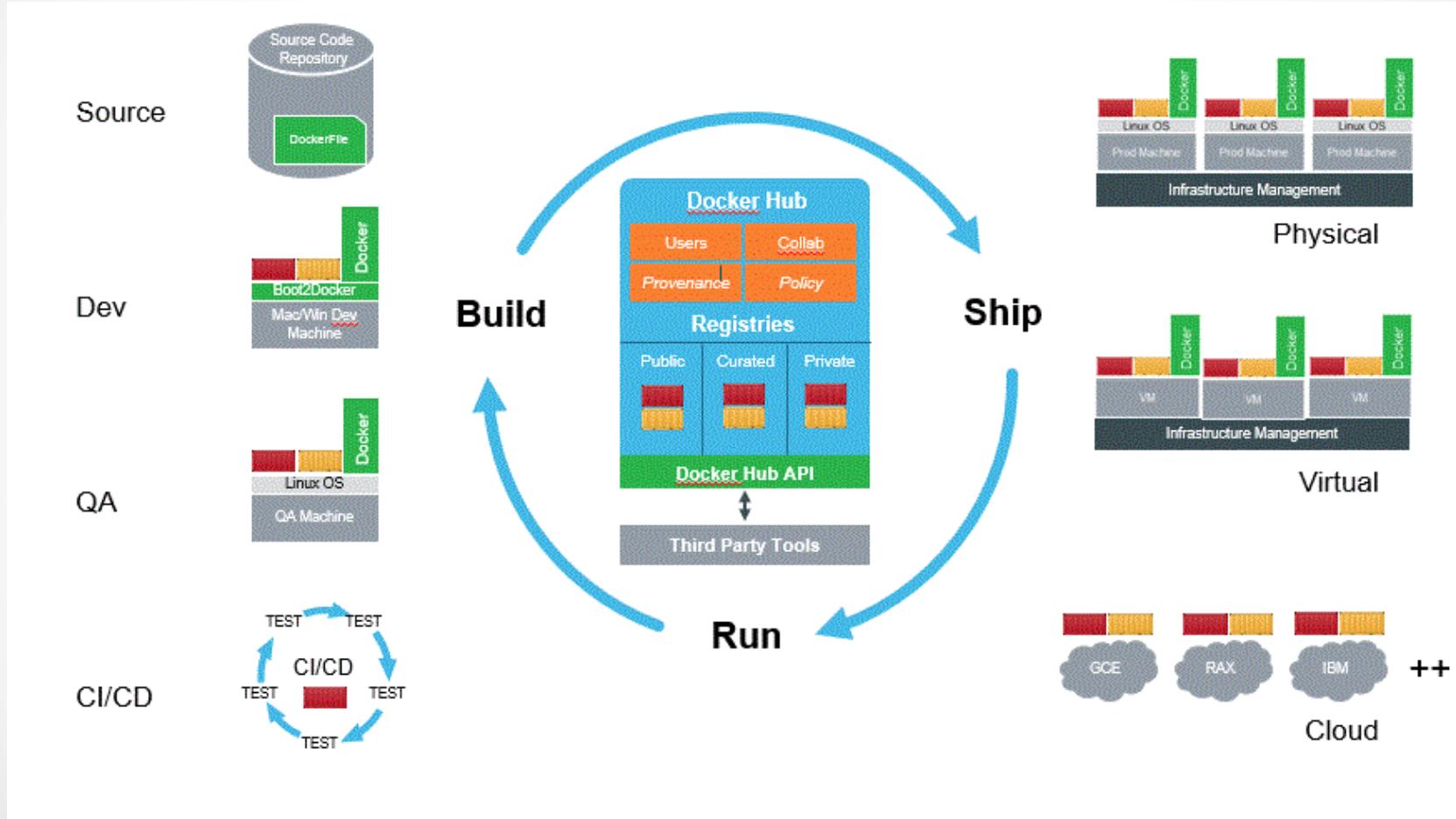
- Worries about what's "outside" the container
 - Logging
 - Remote access
 - Monitoring
 - Network config
- All containers start, stop, copy, attach, migrate, etc. the same way



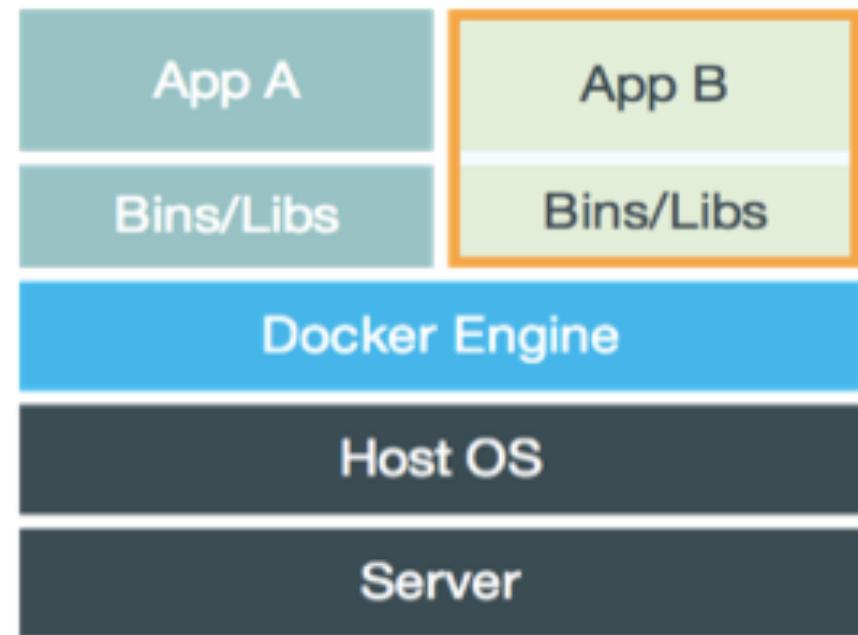
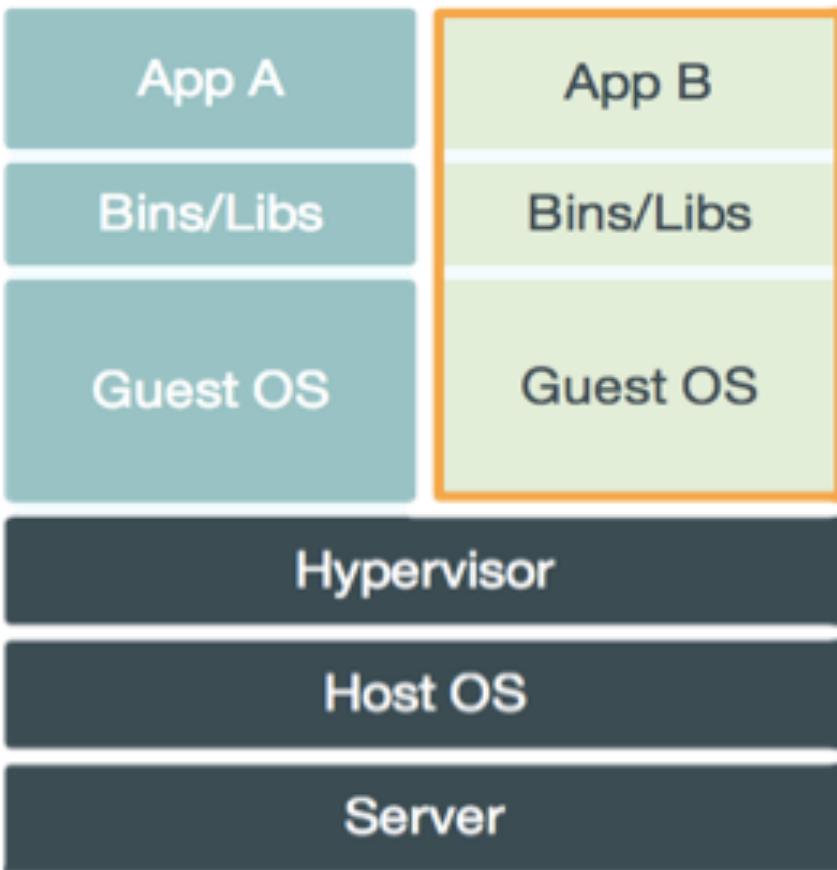
Major components of the container:

Benefit of docker for DevOps

- Build-Ship-Run



Docker vs VMWare



Docker Version

```
ip-192-168-1-112% docker version
Client:
Version: 18.03.0-ce
API version: 1.37
Go version: go1.9.4
Git commit: 0520e24
Built: Wed Mar 21 23:06:22 2018
OS/Arch: darwin/amd64
Experimental: true
Orchestrator: kubernetes

Server:
Engine:
Version: 18.03.0-ce
API version: 1.37 (minimum version 1.12)
Go version: go1.9.4
Git commit: 0520e24
Built: Wed Mar 21 23:14:32 2018
OS/Arch: linux/amd64
Experimental: true
ip-192-168-1-112%
```

```
ip-192-168-1-112% docker info
Containers: 22
Running: 22
Paused: 0
Stopped: 0
Images: 41
Server Version: 18.03.0-ce
Storage Driver: overlay2
Backing Filesystem: extfs
Supports d_type: true
Native Overlay Diff: true
Logging Driver: json-file
Cgroup Driver: cgroupfs
Plugins:
Volume: local
Network: bridge host ipvlan macvlan null overlay
Log: awslogs fluentd gcplogs gelf journalctl json-file logentries splunk syslog
Swarm: inactive
Runtimes: runc
Default Runtime: runc
Init Binary: docker-init
containerd version: cfd04396dc68220d1cecbe686a6cc3aa5ce3667c
runc version: 4fc53a81fb7c994640722ac585fa9ca548971871
init version: 949e6fa
Security Options:
seccomp
Profile: default
Kernel Version: 4.9.87-linuxkit-aufs
Operating System: Docker for Mac
OSType: linux
Architecture: x86_64
CPUs: 4
Total Memory: 1.952GiB
Name: linuxkit-025000000001
ID: WWTW:NOOB:UC5K:TG2P:S3YX:RBCW:GD3J:7HKY:LJDM:6SYT:M7OF:SGXX
Docker Root Dir: /var/lib/docker
Debug Mode (client): false
Debug Mode (server): true
File Descriptors: 140
Goroutines: 132
System Time: 2018-04-09T15:06:53.068007752Z
EventsListeners: 2
HTTP Proxy: docker.for.mac.http.internal:3128
HTTPS Proxy: docker.for.mac.http.internal:3129
Registry: https://index.docker.io/v1/
Labels:
Experimental: true
Insecure Registries:
127.0.0.0/8
Live Restore Enabled: false
ip-192-168-1-112%
```

What's New

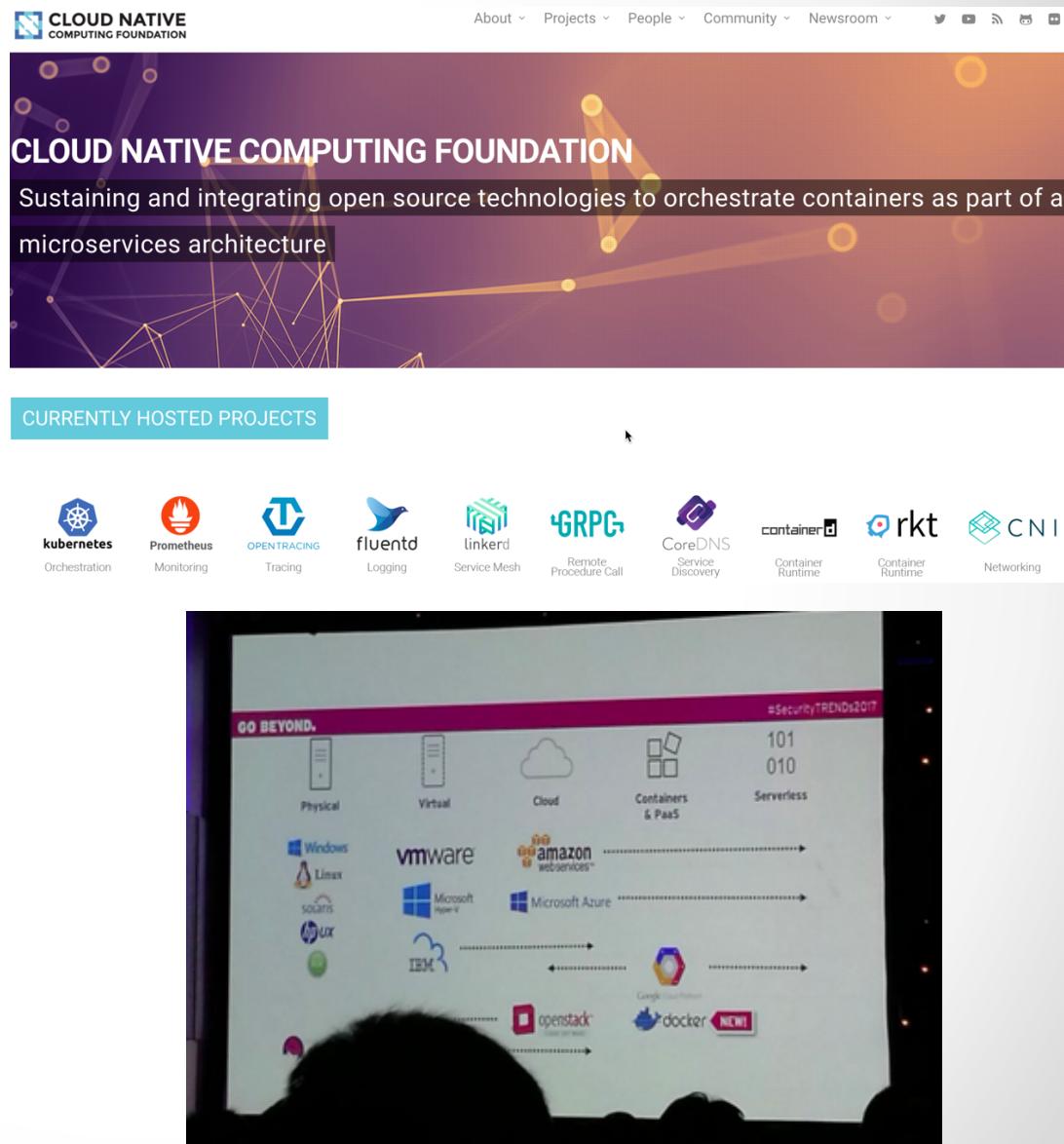


Docker's History of contribution to the OCI

Docker has lead the development of OCI from the initial commit to the donation of runc and Docker V2 Image format as a base for the image format specification.

2014	2015	2016	2017
● FEBRUARY 18 First commit of libcontainer	● APRIL Support for V2 of content addressable Docker Registry API released - groundwork for the OCI Image Specification	● JANUARY Formation of the OCI TOB - 2 Docker maintainers - Michael Crosby and Diogo Monica join effort	● MARCH Docker donates containerd to the CNCF
● FEBRUARY 20 nsinit is developed - the inspiration for runC	● MAY 1 Docker moves to donate its image format and runtime	● APRIL Schema2 of Docker's V2 Image Format support	● APRIL Docker announces project Moby
● MAR 7 Docker 0.9 ships with libcontainer as the default runtime	● MAY/ JUNE Docker works with Inaugural participants and Linux Foundation to donate container format/runtime	● JULY 1.0 of OCI runtime and image format	● DECEMBER Docker spins out containerd
● JUNE 22 Docker announces donation of base container format and runtime, runC*, the cornerstone for the OCI			
● DECEMBER 17 Docker announces containerd, a daemon to manage runc			

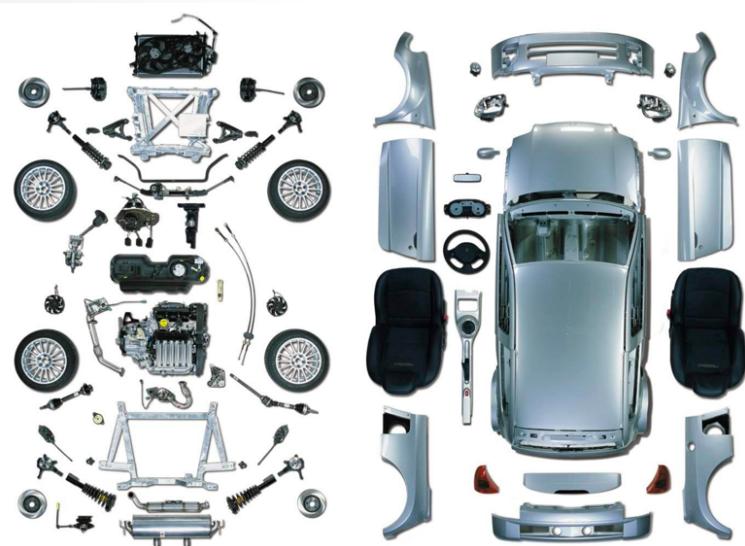
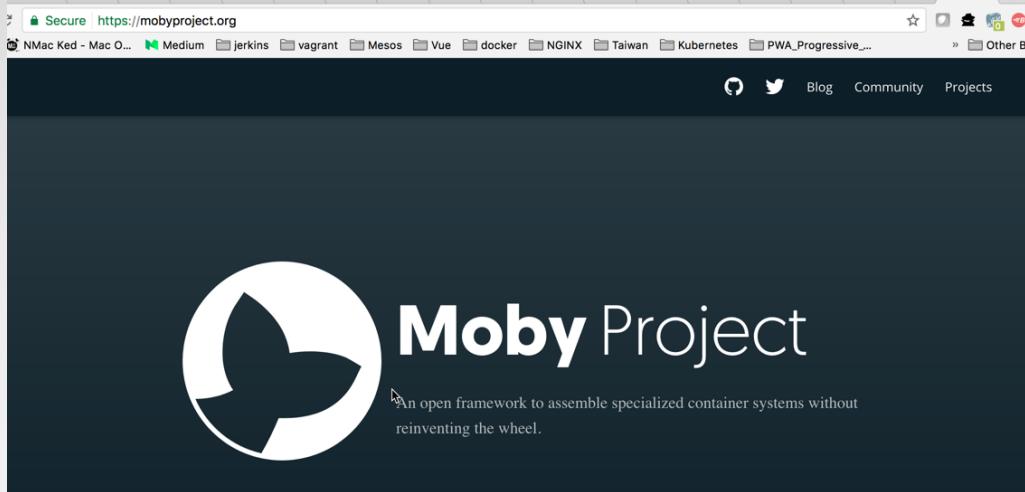
Docker 101



The Cloud Native Computing Foundation (CNCF) homepage features a purple background with abstract geometric shapes. The CNCF logo is at the top left, and a navigation bar with links to About, Projects, People, Community, Newsroom, and social media icons is at the top right. The main title "CLOUD NATIVE COMPUTING FOUNDATION" is in large white letters, followed by a subtitle: "Sustaining and integrating open source technologies to orchestrate containers as part of a microservices architecture". Below this is a section titled "CURRENTLY HOSTED PROJECTS" containing logos for various projects: Kubernetes (Orchestration), Prometheus (Monitoring), OpenTracing (Tracing), fluentd (Logging), linkerd (Service Mesh), gRPC (Remote Procedure Call), CoreDNS (Service Discovery), containerd (Container Runtime), orkt (Container Runtime), and CNI (Networking). At the bottom is a large image of a presentation slide titled "#SecurityTRENDS2017" showing a network diagram with various cloud providers and technologies like Physical, Virtual, Cloud, Containers & PaaS, Serverless, VMware, Amazon Web Services, Microsoft, Microsoft Azure, IBM, OpenStack, and docker.



What's New



Docker 101



What's News

17.06.0-ce (2017-06-28)

Note: Docker 17.06.0 has an issue in the image builder causing a change in the behavior of the `ADD` instruction of Dockerfile when referencing a remote `.tar.gz` file. The issue will be fixed in Docker 17.06.1.

Note: Starting with Docker CE 17.06, Ubuntu packages are also available for IBM z Systems using the s390x architecture.

Note: Docker 17.06 by default disables communication with legacy (v1) registries. If you require interaction with registries that have not yet migrated to the v2 protocol, set the `--disable-legacy-registry=false` daemon option. Interaction with v1 registries will be removed in Docker 17.12.

Server

Platform	Docker CE x86_64	Docker CE ARM	Docker CE System Z (s390x)	Docker EE
CentOS	✓			✓
Debian	✓	✓		
Fedora	✓			
Microsoft Windows Server 2016				✓
Oracle Linux				✓
Red Hat Enterprise Linux			✓	✓
SUSE Linux Enterprise Server				✓
Ubuntu	✓	✓	✓	✓

What's News

Secure | <https://docs.microsoft.com/en-us/sql/linux/quickstart-install-connect-docker>

NMac Ked - Mac OS... Medium Jenkins vagrant Mesos Vue docker NGINX Taiwan Kubernetes PWA_Progressive_W... MYSQL_Cluster GeneralKB Nodejs

 Microsoft Technologies Documentation Resources

Docs Windows Microsoft Azure Visual Studio Office More

Docs / SQL / SQL Server on Linux

Filter

About SQL Server on Linux

- > Overview
- Quickstarts
 - Install & Connect - Red Hat
 - Install & Connect - SUSE
 - Install & Connect - Ubuntu
- Run & Connect - Docker**

> Concepts

> Samples

> Resources

Download PDF

Run the SQL Server 2017 container image with Docker

2017-7-17 • 7 min to read • Contributors 

In this quick start tutorial, you use Docker to pull and run the SQL Server 2017 RC1 container image, [mssql-server-linux](#). Then connect with **sqlcmd** to create your first database and run queries.

This image consists of SQL Server running on Linux based on Ubuntu 16.04. It can be used with the Docker Engine 1.8+ on Linux or on Docker for Mac/Windows.

 **Note**

This quick start specifically focuses on using the mssql-server-linux image. The Windows image is not covered, but you can learn more about it on the [mssql-server-windows Docker Hub page](#).

Prerequisites

- Docker Engine 1.8+ on any supported Linux distribution or Docker for Mac/Windows.
- Minimum of 4 GB of disk space
- Minimum of 4 GB of RAM
- [System requirements for SQL Server on Linux](#).



Platform of Docker

• • •

Platform of Docker

- Docker Toolbox (Legacy) for Desktop (Docker CE)
 - Install docker toolbox on machine will Create VM (Oracle VirtualBox)
 - Dockertoolbox for MAC
 - Dockertoolbox for Windows (7,8,10)
- Docker Native for Desktop (Docker CE)
 - Docker for MAC (Moby Linux (xhyve engine))
 - Docker for Windows (Moby Linux (hyper-v engine))
- Docker Native for Server (Docker CE/EE)
 - Docker for Windows 2016 (EE)
 - Docker for Ubuntu,CENTOS (CE/EE)
 - Docker for Debian (EE/ARM)
 - Docker for Red Hat Enterprise/SUSE/Oracle Linux (EE)
 - Docker for Fedora (CE)

Platform of Docker

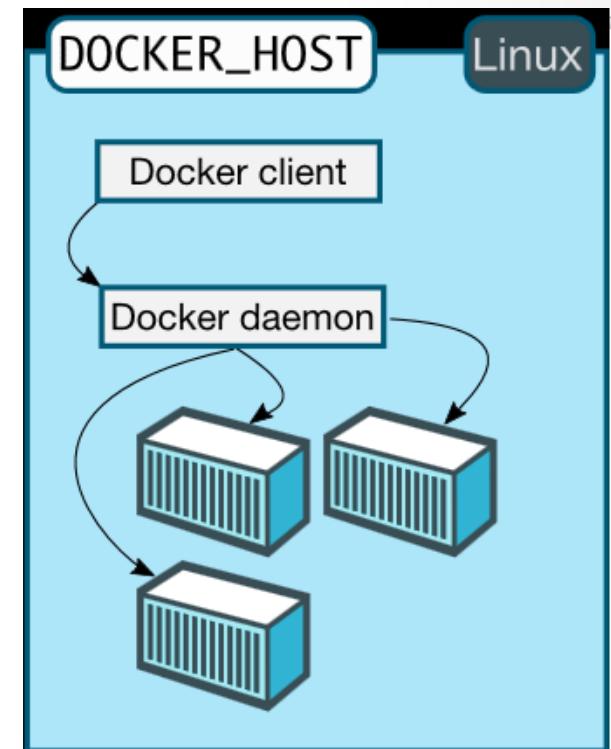
- Pros and Cons.
- Docker Toolbox
 - Pros
 - Compatible for All (Windows 7, 8, 10, MAC OS X)
 - Limit result for operate
 - Suitable for simulate multiple node or swarm
 - Cons
 - Multiple step for transfer source code and debug
 - Overhead for VirtualBox

Platform of Docker

- Pros and Cons.
- Docker for Windows or MAC
 - Pros
 - Full integrate with Machine (No need virtualbox)
 - Auto-update
 - File sharing (Host $\leftarrow \rightarrow$ Container)
 - Http-proxy
 - Custom docker-registry
 - Cons
 - Single Node all case
 - Limit for

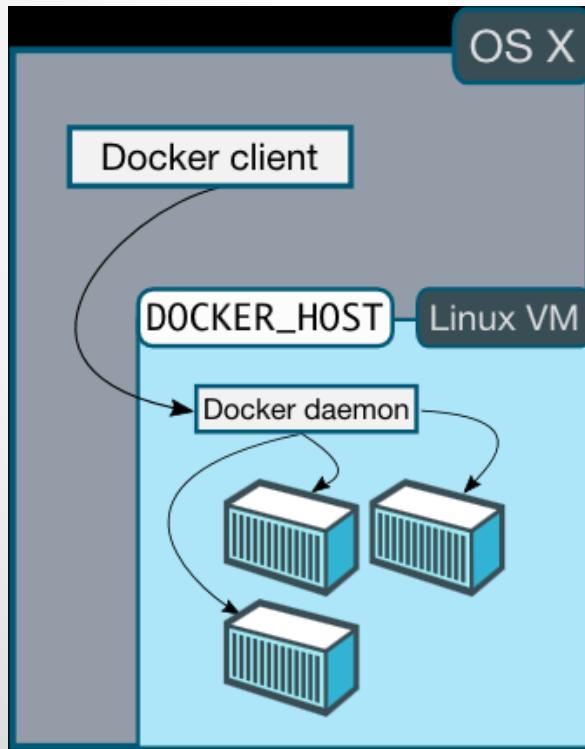
Platform of Docker

- **Docker Native for Desktop/Server**
- การติดตั้ง docker ลงบนเครื่อง linux server / windows (non-toolbox) / mac (non-toolbox) เมื่อเกิดการรัน container ขึ้น docker daemon จะให้บริการบนเครื่อง linux server ที่ทำการติดตั้งโดยตรง ซึ่งเราสามารถเรียกใช้งานโปรแกรมบนเครื่องได้ทันที

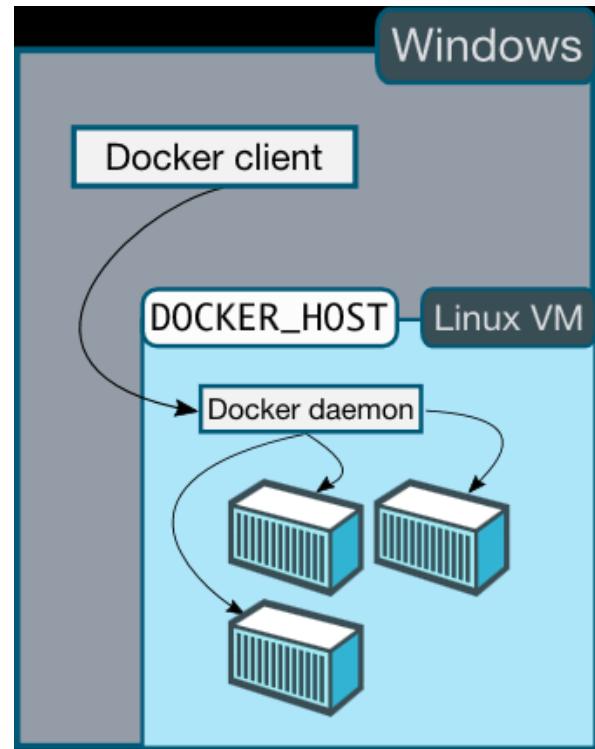


Linux vs Windows vs MAC OS

- Docker ToolBox
- การติดตั้ง docker ลงบนเครื่อง windows หรือ mac os จะมีการติดตั้ง virtual machine ใหม่ขึ้นบนเครื่องเพื่อให้บริการสำหรับ docker โดยเฉพาะดังนั้นมีการสั่งรันจึงเกิดการ remote run ไปยังเครื่อง virtual machine (docker-machine)



Docker 101



WorkShop

• • •



WorkShop

Container: Nodejs
Port: 3000



Container: Python
Port: 5000



Container: Mysql
Port: 3306



Container: XXXXX
Port: XXXX

Docker Server

Map Port
(3000:3000 → Nodejs)
(5000:5000 → Python)



WorkShop

- Part1: NODEJS

Container Name: NODEJS
IP Address: X.X.X.X (Port: 3000:3000)



Docker-Server:
Path: /Workshop_2_Docker_for_Nodejs/app



Server-Side request jade to compile
(HTML+Javascript)



Default.jade

Menu HTML

WorkShop

Server.js

```
var express = require('express')
, logger = require('morgan')
, app = express()
, template = require('jade').compileFile(__dirname + '/source/templates/homepage.jade')

app.use(logger('dev'))
app.use(express.static(__dirname + '/static'))

app.get('/', function (req, res, next) {
  try {
    var html = template({ title: 'Home' })
    res.send(html)
  } catch (e) {
    next(e)
  }
})

app.listen(process.env.PORT || 3000, function () {
  console.log('Listening on The Node.js logo features the word "node" in a lowercase, sans-serif font where each letter is a different color: a dark gray 'n', a green 'o', a light gray 'd', and a black 'e'. Below it is a smaller, stylized green hexagon containing the letters "JS". A small registered trademark symbol \(®\) is located at the top right of the hexagon.
```

WorkShop

Homepage.jade



```
extend default

block content
p.
  Welcome to Example "Nodejs Thailand WebPage" that build from Nodejs, Express under
  Docker for NodeJS version 1.0. This example webpage is come from container technology (Docker)

p.
  Docker and Container Technology is formally call "OS level virtualization" and also
  known as containerization, refers to an operating system feature in which the kernel
  allows the existence of multiple isolated user-space instances. Such instances,
  called containers,[1] partitions, virtualization engines (VEs) or jails
  (FreeBSD jail or chroot jail), may look like real computers from the point of view
  of programs running in them. A computer program running on an ordinary person's
  computer's operating system can see all resources (connected devices, files and folders
  , network shares, CPU power, quantifiable hardware capabilities) of that computer.
  However, programs running inside a container can only see the container's contents
  and devices assigned to the container.
```

Docker 101



WorkShop

Default.jade



```
doctype html
html
  head
    link(rel='stylesheet', href='/css/index.css')
    title Nodejs Thailand
  body
    .header
      h1.page-title Nodejs Thailand

    ul.nav
      li: a(href='') Home
      li: a(href='https://www.docker.com') Docker Website
      li: a(href='https://nodejs.org/en/') NodeJS Website
      li: a(href='https://www.facebook.com/praparn.lungpoonlap') About Me

    .main-content
      block content

    .footer
      p Copyright © 2018 Praparn Luangphoonlap. Labdockerthailand ® is a Registered Trademark of Docker (un)Ltd. in the Earth and other
```

Docker 101



WorkShop

Dockerfile

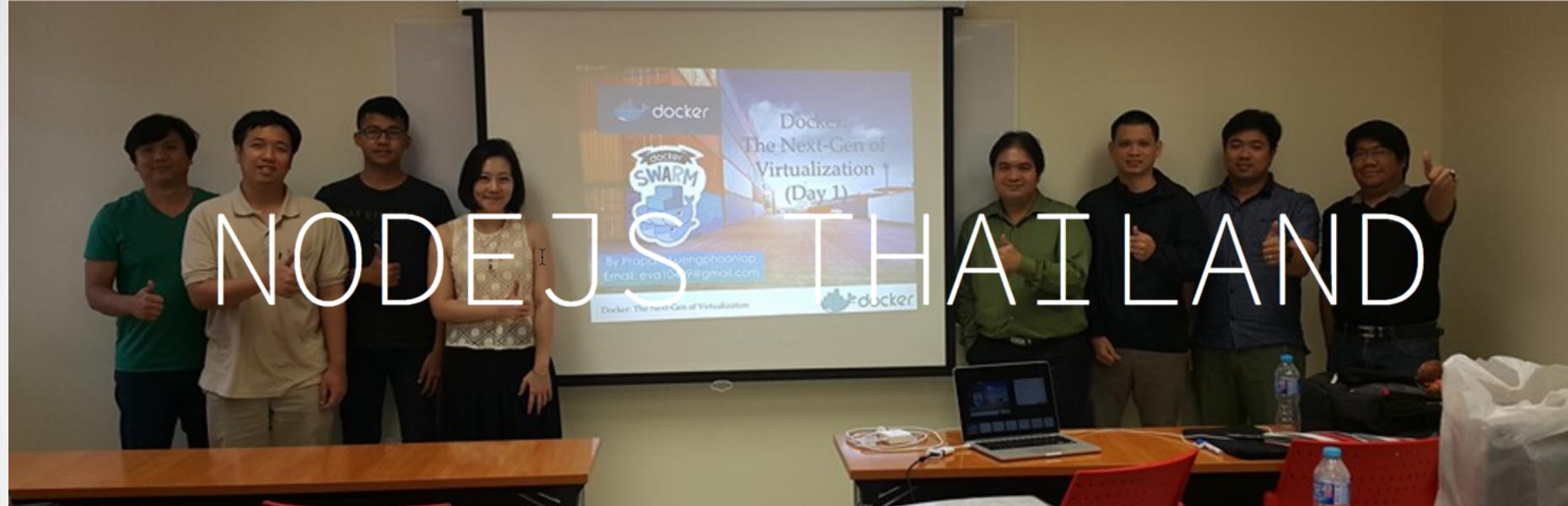
```
FROM bitnami/node:latest
MAINTAINER Praparn Lueangphoonlap (eva10409@gmail.com)
LABEL Description="NodeJS Workshop" Version="1.0"
ENV NODE_VERSION=v6.12.2 NPM_VERSION=3.10.10
RUN mkdir /nodejs
COPY ./app/. /app/
WORKDIR /app
RUN npm i && \
    npm run build
ENTRYPOINT ["node","server.js"]
EXPOSE 3000
```

Docker Run

```
[docker@labdocker:~$ docker pull labdocker/nodejsbitnami:latest
latest: Pulling from labdocker/nodejsbitnami
Digest: sha256:f1fd52b61b2a32bae96723b8aea907f58aa18412aae37aa1f517536af3a8b8ba
Status: Image is up to date for labdocker/nodejsbitnami:latest
[docker@labdocker:~$ docker run -dt --name nodejs -p 3000:3000 labdocker/nodejsbitnami:latest
0b5ed4ad6802711825337e744d89fb62c79159e4fa545b783bdbc49ad1170ce3
[docker@labdocker:~$ docker ps
CONTAINER ID        IMAGE               COMMAND             CREATED            STATUS              PORTS               NAMES
0b5ed4ad6802        labdocker/nodejsbitnami:latest   "node server.js"   2 seconds ago     Up 2 seconds          0.0.0.0:3000->3000/tcp   nodejs
[docker@labdocker:~$ ]
```

WorkShop

Output



Welcome to Example "Nodejs Thailand WebPage" that build from Nodejs, Express under Docker for NodeJS version 1.0. This example webpage is come from container technology (Docker)

WorkShop

- This example nodejs images from “Bitnami”

PUBLIC | AUTOMATED BUILD

bitnami/node ☆

Last pushed: 10 days ago

Repo Info Tags Dockerfile Build Details

Short Description

Bitnami Node.js Docker Image

Full Description

ciircled ci passing
slack join chat →

What is Node.js?

Node.js is a platform built on Chrome's JavaScript runtime for easily building fast, scalable network applications. Node.js uses an event-driven, non-blocking I/O model that makes it lightweight and efficient, perfect for data-intensive real-time applications that run across distributed devices.

nodejs.org

Docker Pull Command

`docker pull bitnami/node`

Owner

 bitnami

Source Repository

 [bitnami/bitnami-docker-node](https://github.com/bitnami/bitnami-docker-node)

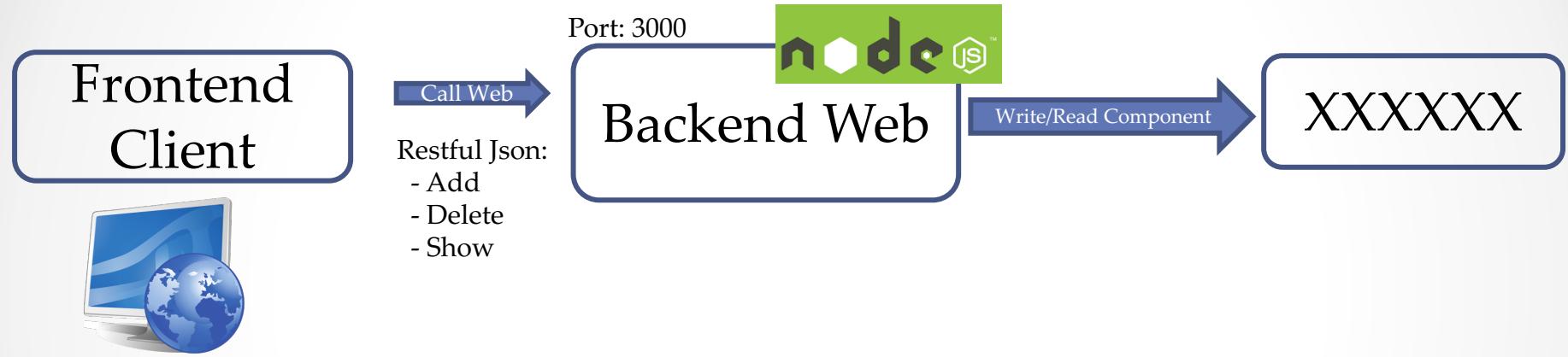
<https://github.com/bitnami/bitnami-docker-node>

WorkShop

- Why “bitnami” ?
 - Bitnami closely tracks upstream source changes and promptly publishes new versions of this image using our automated systems.
 - With Bitnami images the latest bug fixes and features are available as soon as possible.
 - Bitnami containers, virtual machines and cloud images use the same components and configuration approach - making it easy to switch between formats based on your project needs.
 - Bitnami images are built on CircleCI and automatically pushed to the Docker Hub.
 - **All our images are based on [minideb](#) a minimalist Debian based container image which gives you a small base container image and the familiarity of a leading linux distribution**

WorkShop

- Basic Component

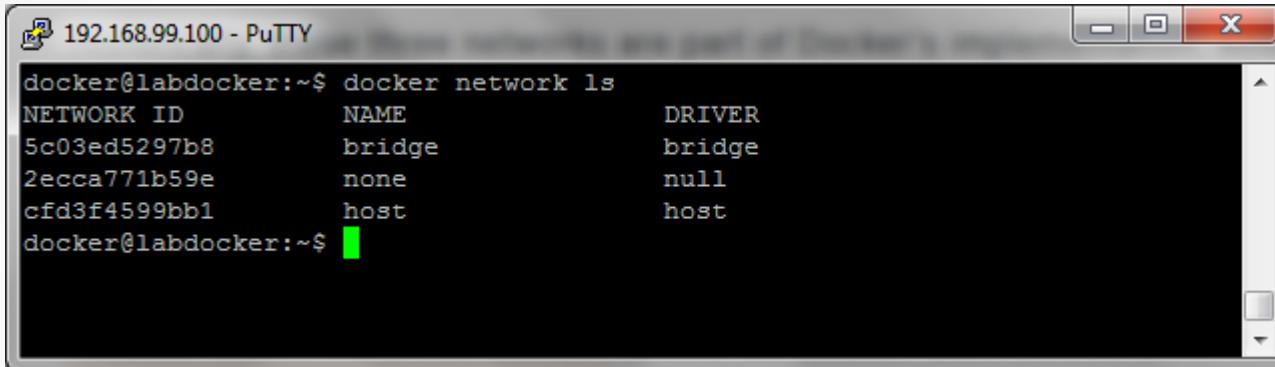


Docker Network

• • •

Docker Network

- Software define network by design (virtual switch)
- default docker จะจัดเตรียม network มาให้สามรูปแบบ

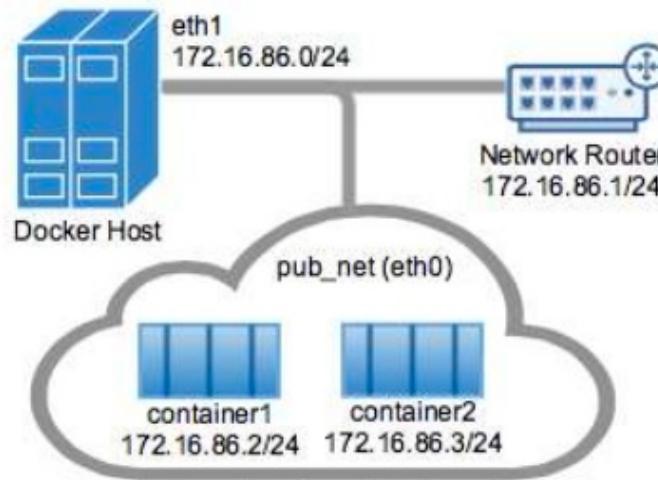


```
192.168.99.100 - PuTTY
docker@labdocker:~$ docker network ls
NETWORK ID      NAME      DRIVER
5c03ed5297b8    bridge    bridge
2ecca771b59e    none     null
cf3f4599bb1     host     host
docker@labdocker:~$
```

- **bridge** คือ default network สำหรับให้ container เชื่อมต่อออกสู่โลกภายนอกผ่าน virtual switch “docker0” โดยใช้ network stack ของ container เองในการเชื่อมต่อ (route mode)
- **none** คือ network loopback (127.0.0.1) สำหรับ container ที่ไม่มีการเชื่อมต่อออกไปด้านนอก
- **host** คือ network host ที่ container ใช้งาน host network stack ในการทำงาน
(ใช้ในกรณี ต้องการ network performance สูงสุด) (security concern)

Docker Network

- Additional network type
- **MACVLAN**

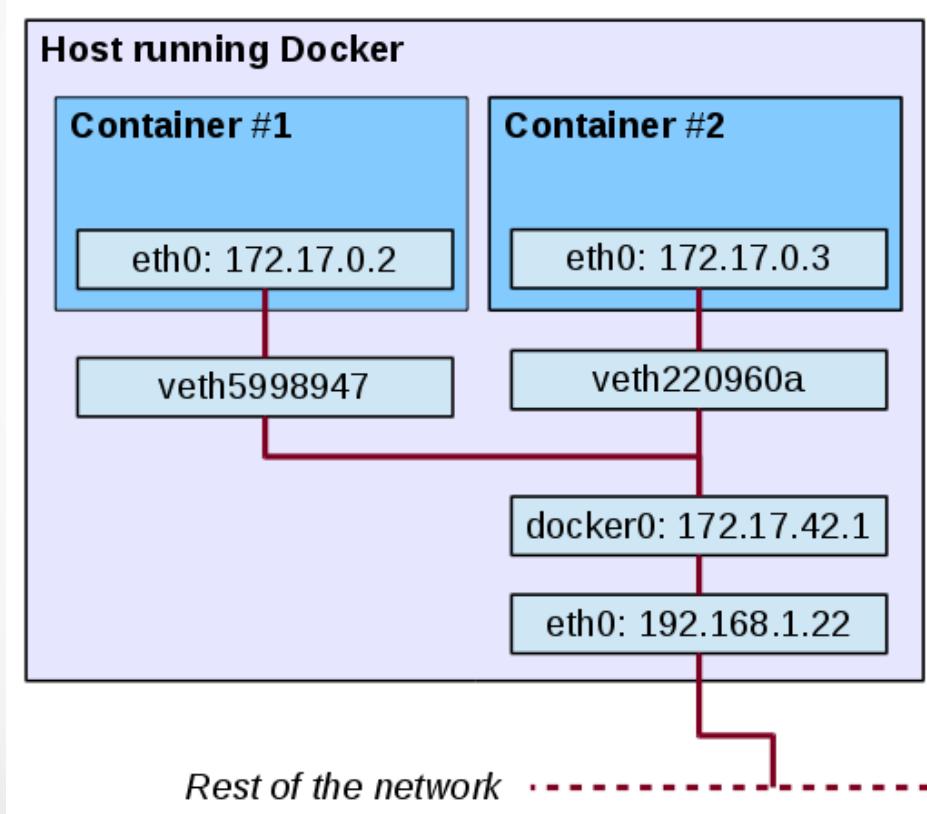


```
docker network create -d macvlan \
    --subnet=172.16.86.0/24 \
    --gateway=172.16.86.1 \
    --parent=eth1 pub_net
```

- **Overlay (Swarm Mode)**
- **3rd Party Network Plugin (Swarm Mode)**

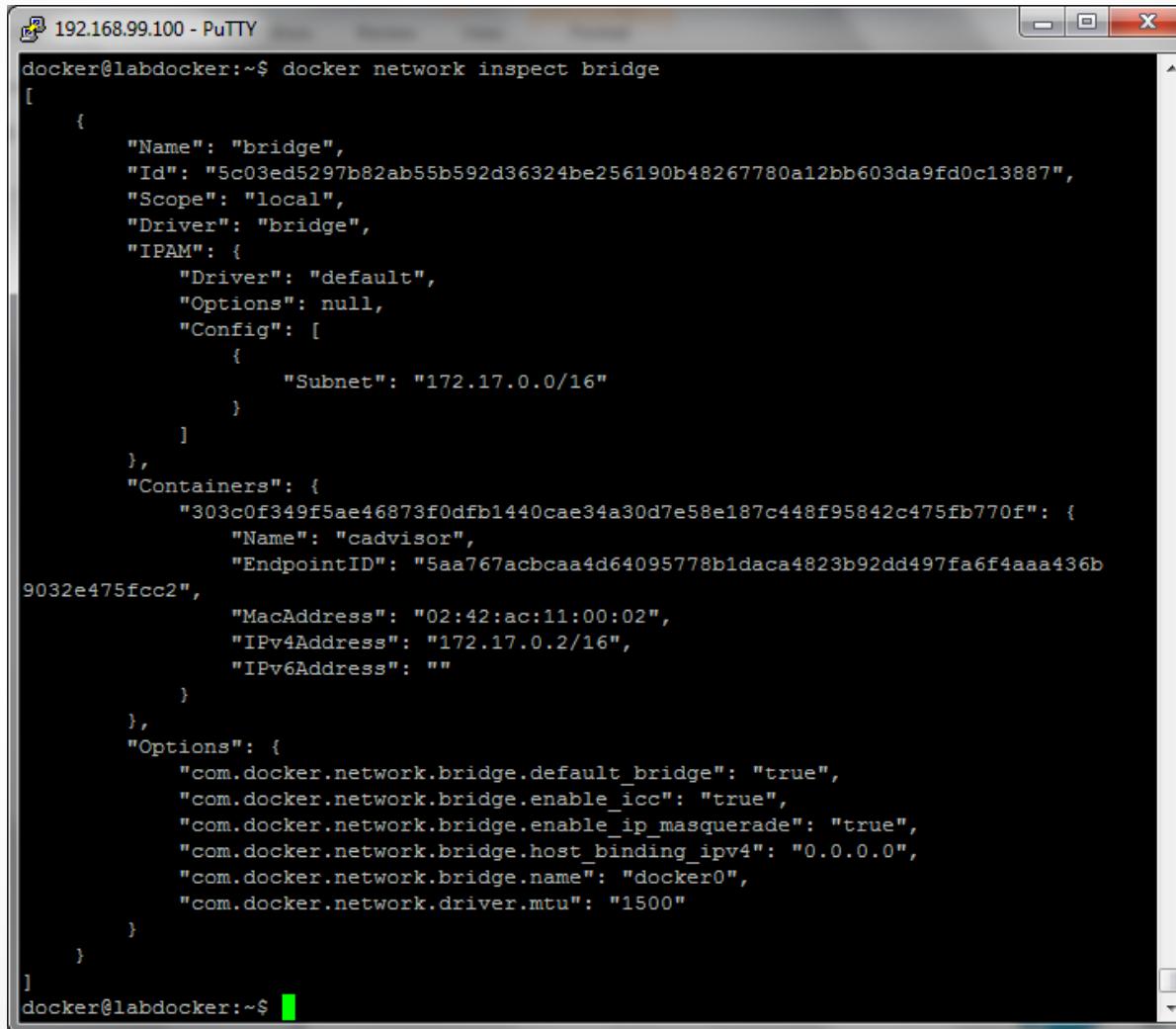
Docker Network

- bridge (default) เมื่อสั่งรัน container ในระบบโดยไม่ได้ระบุ network ใดๆ container จะถูกเพิ่ม ipaddress, subnet เข้าสู่ bridge network โดยอัตโนมัติ



Docker Network

- docker network inspect bridge



A screenshot of a PuTTY terminal window titled "192.168.99.100 - PuTTY". The window displays the output of the command "docker network inspect bridge". The output is a JSON object representing the bridge network configuration. Key details include the network's name ("bridge"), ID ("5c03ed5297b82ab55b592d36324be256190b48267780a12bb603da9fd0c13887"), driver ("bridge"), IPAM settings (subnet "172.17.0.0/16"), and a container ("cadvisor") connected to it with specific IP and MAC addresses.

```
192.168.99.100 - PuTTY
docker@labdocker:~$ docker network inspect bridge
[
    {
        "Name": "bridge",
        "Id": "5c03ed5297b82ab55b592d36324be256190b48267780a12bb603da9fd0c13887",
        "Scope": "local",
        "Driver": "bridge",
        "IPAM": {
            "Driver": "default",
            "Options": null,
            "Config": [
                {
                    "Subnet": "172.17.0.0/16"
                }
            ]
        },
        "Containers": {
            "303c0f349f5ae46873f0dfb1440cae34a30d7e58e187c448f95842c475fb770f": {
                "Name": "cadvisor",
                "EndpointID": "5aa767acbc当地4d64095778b1daca4823b92dd497fa6f4aaa436b9032e475fcc2",
                "MacAddress": "02:42:ac:11:00:02",
                "IPv4Address": "172.17.0.2/16",
                "IPv6Address": ""
            }
        },
        "Options": {
            "com.docker.network.bridge.default_bridge": "true",
            "com.docker.network.bridge.enable_icc": "true",
            "com.docker.network.bridge.enable_ip_masquerade": "true",
            "com.docker.network.bridge.host_binding_ipv4": "0.0.0.0",
            "com.docker.network.bridge.name": "docker0",
            "com.docker.network.driver.mtu": "1500"
        }
    }
]
docker@labdocker:~$
```

Docker Network

- Option เกี่ยวกับ network สำหรับสั่ง run container

--dns= x.x.x.x (Default --name,--net-alias,--link
also dns internal docker)
--net="<bridge/none/host/custom>"
--net-alias = "xxxx" <new in 1.11>
--add-host="xxxx"
--mac-address="xx:xx:xx:xx"
--ip="x.x.x.x"
--ipv6="xx:xx:xx:xx" <new in 1.11>
-p, --publish = 9999:9999
-P, --publish-all → Auto map network

```
docker run -i -t --rm --name nodejs -p 3000:3000 \
labdocker/alpineweb:latest node nodejs/hello.js
```

Docker Network

- สร้าง custom virtual switch เพื่อจัดระเบียบและแบ่งแยก network ของ application ออกจากกัน (Recommend for Production)

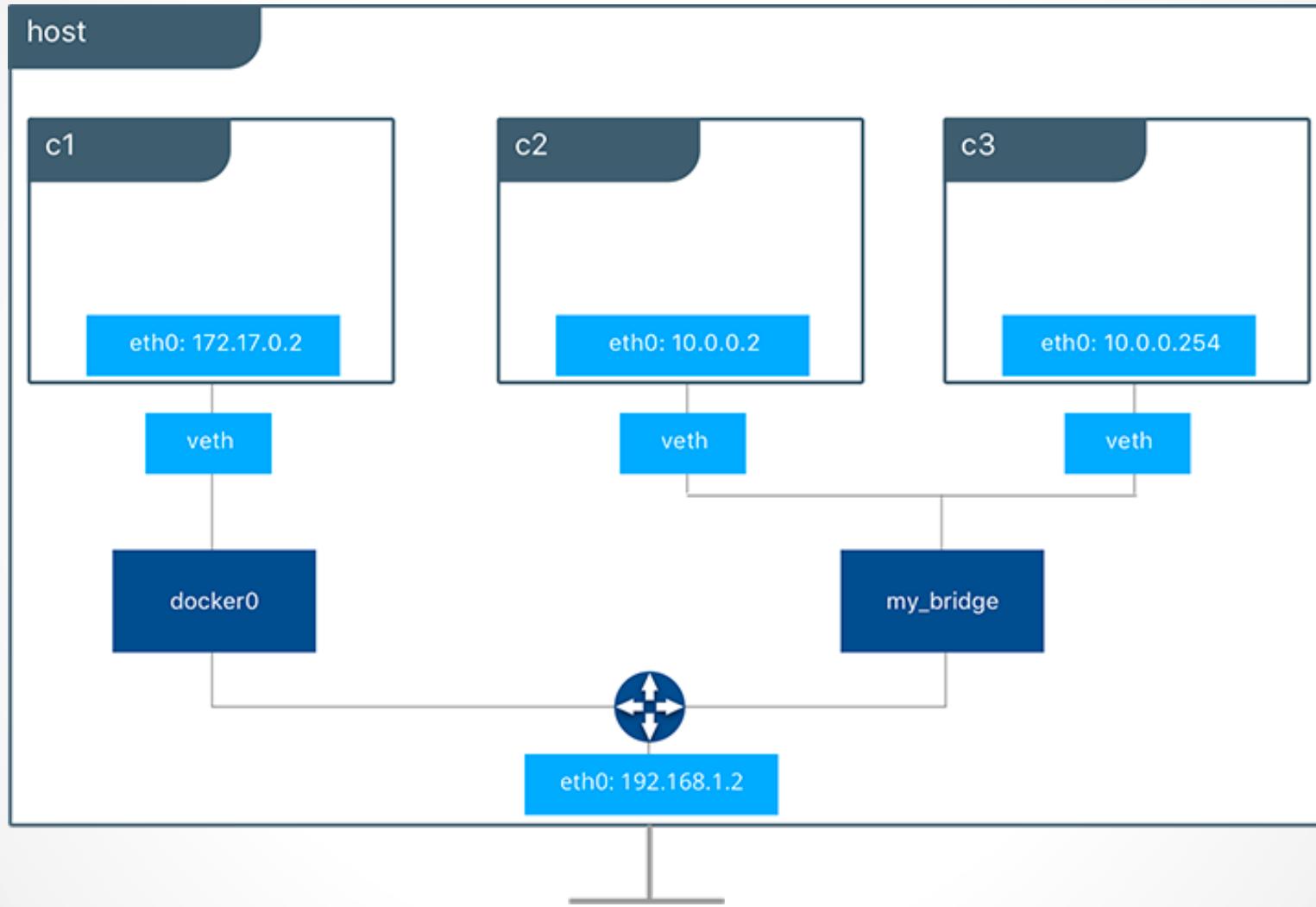
```
docker network create --driver <default/null/host> name
```

Ex: docker network create --driver default netweb

Ex: docker network rm netweb

- สามารถเพิ่มเติม option ในการสร้าง virtual switch เพิ่มเติม
 - d macvlan -o parent = กำหนด vlan tagging
 - subnet = xx (ระบุ ip address ของ virtual switch (Ex: 192.168.100.0/24))
 - ip-range = xx (ระบุ ip address ที่จะส่งให้ container ทำงาน) (Ex: 192.168.100.128/25)
 - gateway = xx (ระบุ ip address gateway) (Ex: 192.168.100.5)
 - opt = custom option
 - opt="com.docker.network.mtu"="1500"
 - opt="com.docker.network.bridge.host_binding_ipv4"=x.x.x.x

Docker Network



Docker Network

- การเพิ่ม network ใน container

```
docker network connect <network> <container>
```

```
Ex: docker network connect webnet web1
```

- การลด network ใน container

```
docker network disconnect <network> <container>
```

```
Ex: docker network disconnect webnet web1
```

Docker Network

- Link container (add on /etc/hosts)

```
docker -dt --name web1 labdocker/alpineweb sh
```

```
docker -dt --name web2 --link web1:webmaster \  
labdocker/alpineweb sh
```

- DNS resolve on docker network (Recommend for Production)

Docker Network

- Reverse Proxy Network (DNS)
- Purpose: ทำการ Deploy nodejs webserver ด้วย solution proxy ของ nginx (load balance / reverse proxy)
- **public network** เพื่อให้บุคคลภายนอกเข้าใช้งานผ่าน nginx server (reverse proxy)
 - IP Address: 192.168.100.0/24
 - Range Address: 192.168.100.128 – 192.168.100.256
 - MTU: 1500
- **private network** เพื่อให้ nginx server เข้าเรียกใช้งาน nodejs web server
 - IP Address: 192.168.101.0/24
 - Range Address: 192.168.101.128 – 192.168.101.256
 - MTU: 9000

Docker Network

Reverse Proxy to Node.js Application



Docker Network

WEB1 (No Map)

DNS Name: web1 (Private)

WEB2 (No Map)

DNS Name: web2 (Private)

vSwitch: Webinternal

IP Address: 192.168.101.0/24

NGINX (Map Port:80:8080)

Join Network: Webinternal

Join Network: Webpublic

vSwitch: Webpublic

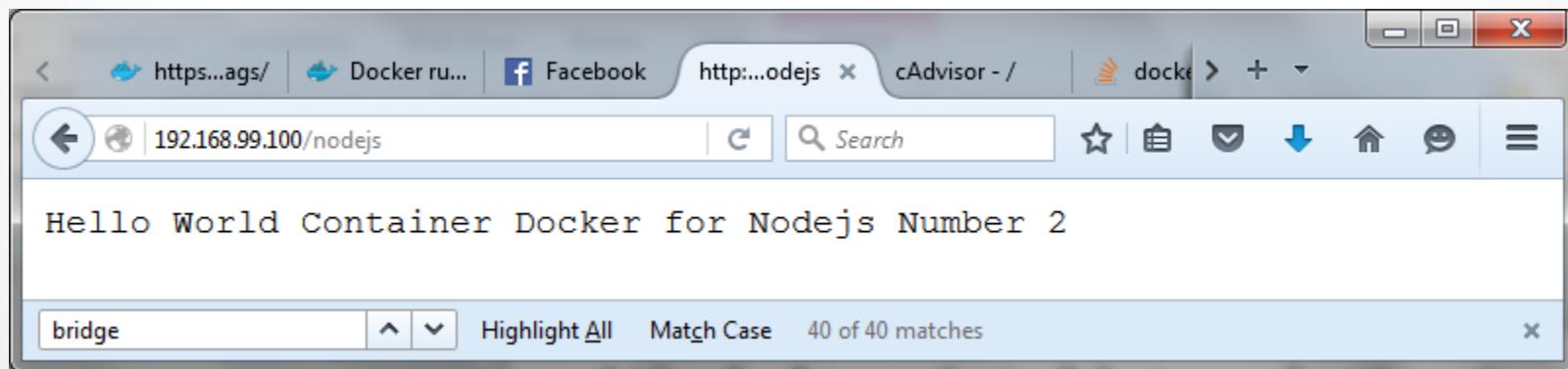
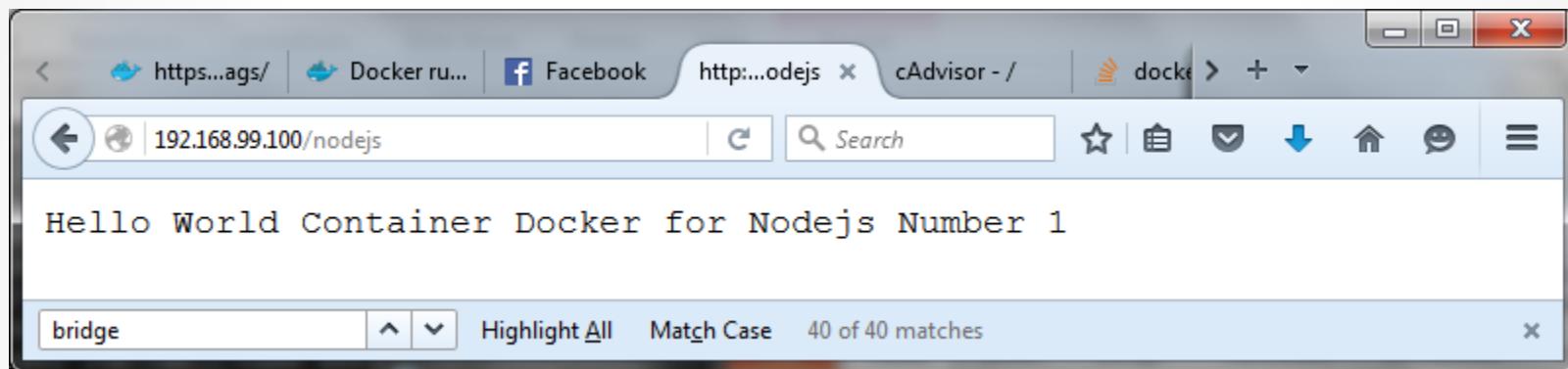
IP Address: 192.168.100.0/24

Map Port
(80:8080)

Public
Network

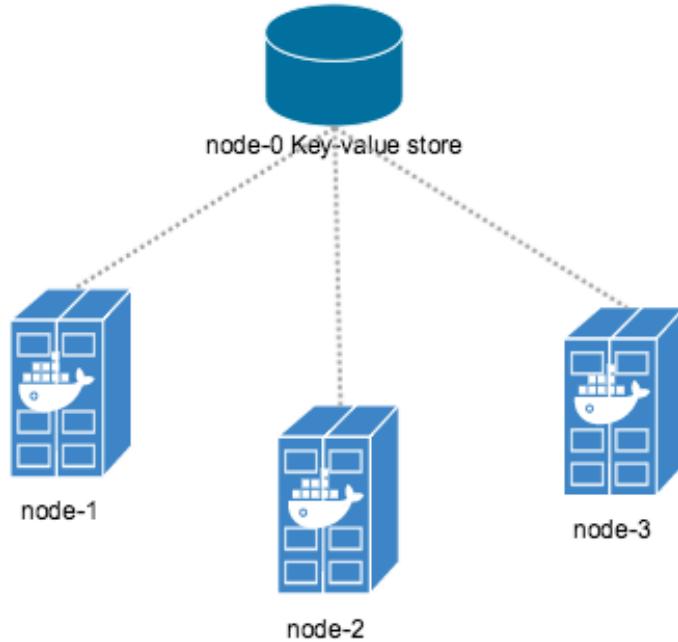
Docker Network

- url: http://<ip address of docker-machine>/nodejs



Docker Network

- Network across host (Overlay Network)



Docker Network

- 3rd Party Network Plugin

Driver	Description
contiv	An open source network plugin led by Cisco Systems to provide infrastructure and security policies for multi-tenant microservices deployments. Contiv also provides integration for non-container workloads and with physical networks, such as ACI. Contiv implements remote network and IPAM drivers.
weave	A network plugin that creates a virtual network that connects Docker containers across multiple hosts or clouds. Weave provides automatic discovery of applications, can operate on partially connected networks, does not require an external cluster store, and is operations friendly.
calico	An open source solution for virtual networking in cloud datacenters. It targets datacenters where most of the workloads (VMs, containers, or bare metal servers) only require IP connectivity. Calico provides this connectivity using standard IP routing. Isolation between workloads – whether according to tenant ownership or any finer grained policy – is achieved via iptables programming on the servers hosting the source and destination workloads.
kuryr	A network plugin developed as part of the OpenStack Kuryr project. It implements the Docker networking (libnetwork) remote driver API by utilizing Neutron, the OpenStack networking service. Kuryr includes an IPAM driver as well.

Docker Remote IPAM Drivers

Community and vendor created IPAM drivers can also be used to provide integrations with existing systems or special capabilities.

Driver	Description
infoblox	An open source IPAM plugin that provides integration with existing Infoblox tools.

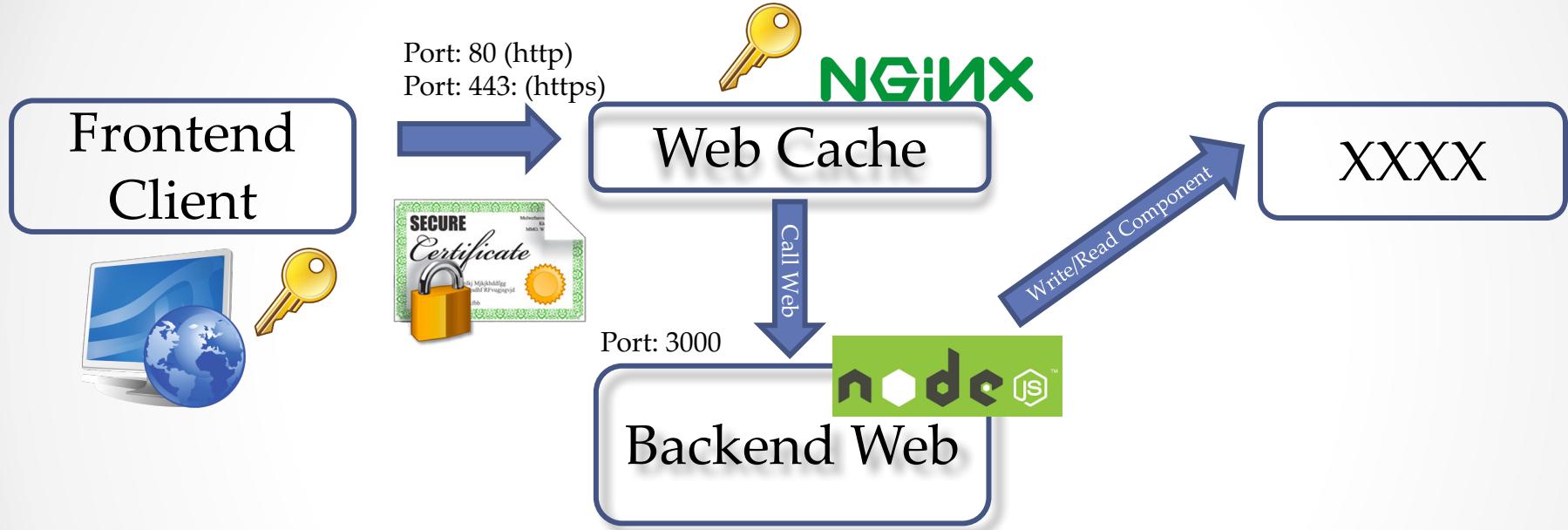
Docker Network

Reverse Proxy to Node.js Application



WorkShop

- Optimize for WorkLoad



WorkShop

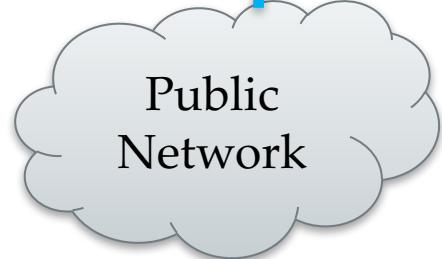
- NODEJS with NGINX

Container Name: nodejs
DNS Name: "nodejs"
Join Network: nodejs_new

Container Name: nginx
(Map Port: 80:8080, 443: 8443)
Join Network: nodejs_new

vSwitch: nodejs_net
IP Address: 192.168.100.0/24

Port: 80:8080 (http)
Port: 443:8443: (https)



WorkShop

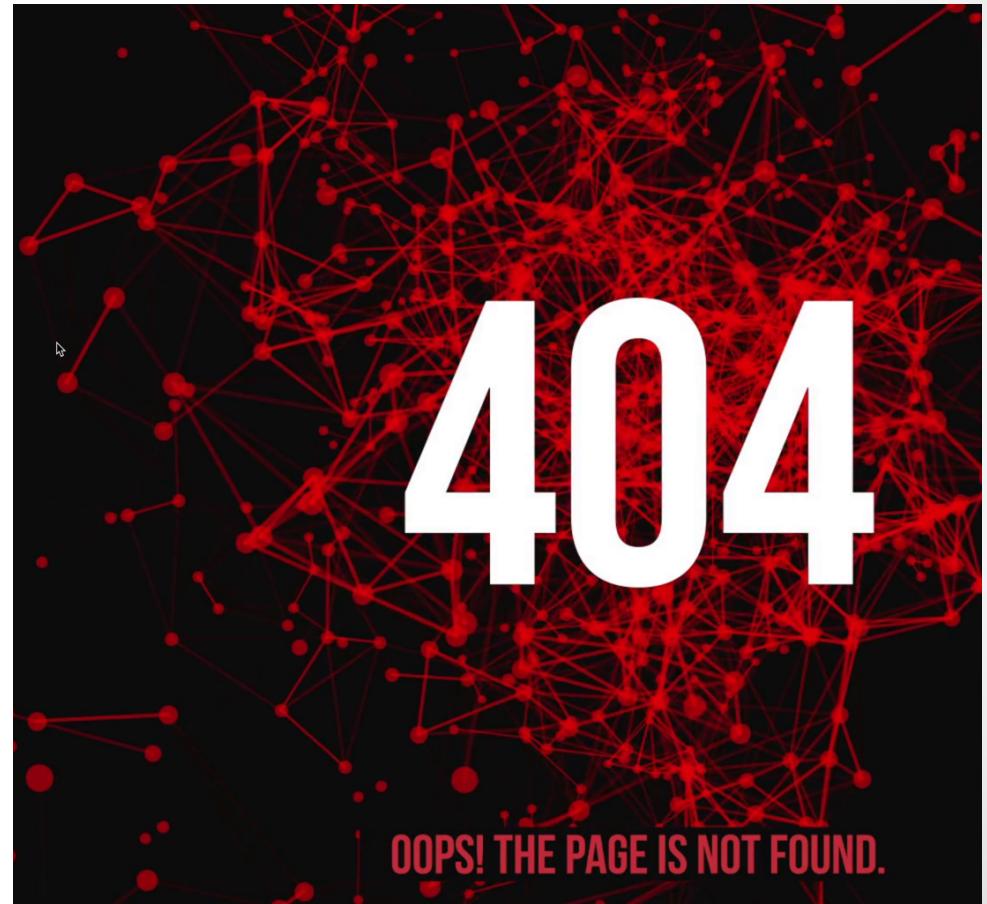
vhost.conf

```
server {
    listen 8080;
    listen 8443 ssl;
    ssl_certificate      /bitnami/nginx/conf/bitnami/certs/server.crt;
    ssl_certificate_key  /bitnami/nginx/conf/bitnami/certs/server.key;
    client_body_buffer_size 100M;
    index index.html      index.htm;
    location / {
        #root /opt/bitnami/nginx/conf/vhosts/404
        #index 404.html
    }
}
server {
    listen 8080;
    listen 8443 ssl;
    server_name www.nodejsthailand.com;
    ssl_certificate      /bitnami/nginx/conf/bitnami/certs/www.nodejsthailand.com.crt;
    ssl_certificate_key  /bitnami/nginx/conf/bitnami/certs/www.nodejsthailand.com.key;
    client_body_buffer_size 100M;
    index index.html      index.htm;
    location / {
        proxy_pass http://nodejs:3000;
        proxy_next_upstream error timeout invalid_header http_500 http_502 http_503 http_504;
        proxy_redirect off;
        proxy_buffering off;
        proxy_set_header Host $host;
        proxy_set_header X-Real-IP $remote_addr;
        proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
        proxy_set_header X-Forwarded-Proto $scheme;
    }
}
```

WorkShop

For 404 default

Case: Access to other name (Denied Access)



WorkShop

dockerfile

```
FROM bitnami/nginx:latest
MAINTAINER Praparn Lueangphoonlap (eva10409@gmail.com)
LABEL Description="NGINX Reverse Proxy" Version="1.0"
COPY /conf/. /bitnami/nginx/conf
COPY /404/. /app
EXPOSE 8080 8443
|
```

WorkShop

Dockerfile

```
FROM bitnami/node:latest
MAINTAINER Praparn Lueangphoonlap (eva10409@gmail.com)
LABEL Description="NodeJS Workshop" Version="1.0"
ENV NODE_VERSION=v6.12.2 NPM_VERSION=3.10.10
RUN mkdir /nodejs
COPY ./app/. /app/
WORKDIR /app
RUN npm i && \
    npm run build
ENTRYPOINT ["node","server.js"]
EXPOSE 3000
```

Docker Run

```
docker@labdocker:~$ docker run -dt --name nodejs --net nodejs_net \
> --net-alias nodejs labdocker/nodejsbitnami:latest
19c2df807044b78b077a3b2ad7c6b517b67c2635278f0712d1c9023fe9b4e2f6
docker@labdocker:~$ docker run -dt --name nginx -p 80:8080 -p 443:8443 --net nodejs_net \
> --net-alias nginx labdocker/nginxbitnami:latest
63f2f5067342e43dfeb6a6e9fdb10c4dfa8a2f8b7acfe54135770a2f40d33af
docker@labdocker:~$ docker ps
CONTAINER ID        IMAGE               COMMAND                  CREATED             STATUS              PORTS                 NAMES
63f2f5067342        labdocker/nginxbitnami:latest   "/app-entrypoint.sh ..."   6 seconds ago       Up 5 seconds          0.0.0.0:80->8080/tcp, 0.0.0.0:443->8443/tcp   nginx
19c2df807044        labdocker/nodejsbitnami:latest   "node server.js"         12 seconds ago      Up 11 seconds         3000/tcp              nodejs
docker@labdocker:~$
```

WorkShop

Output: <https://www.nodejsthailand.com>

The screenshot shows a web browser displaying the Node.js Thailand website at <https://www.nodejsthailand.com>. The page features a large "NODEJS THAILAND" logo overlaid on a photograph of a workshop group. Below the logo, there are links for "Home", "Docker Website", and "NodeJS Website". The main content area contains text about Docker and Container Technology, mentioning Node.js, Express, and Docker for NodeJS version 1. To the right of the main content, a Docker container status window is open, showing details like IP address, port mapping, and MAC address. A small Docker logo is visible in the bottom right corner.

Nodejs Thailand

https://www.nodejsthailand.com

NODEJS THAILAND

Home Docker Website NodeJS Website

Welcome to Example "Nodejs Thailand WebPage" that Nodejs, Express under Docker for NodeJS version 1 example webpage is come from container technology

Docker and Container Technology is formally called "virtualization" and also known as containerization, an operating system feature in which the kernel allows the existence of multiple isolated user-space instances, called containers, [1] partitions, virtual engines (VEs) or jails (FreeBSD jail or chroot jail), like real computers from the point of view of programs in them. A computer program running on an ordinary computer's operating system can see all resources (connect

ชื่อ URL: https://www.nodejsthailand.com/
ที่มาของ: เว็บไซต์นี้ไม่มีข้อมูลเจ้าของเว็บ
ชื่อยืนยันโดย: CN=www.nodejsthailand.com
หมดอายุเมื่อ: 28 ธันวาคม 2570

ตัวดูในบันรอง: "www.nodejsthailand.com"

ที่มาของ: www.nodejsthailand.com
ที่มาของ: <ไม่ใช่ส่วนหนึ่งของในบันรอง>
ที่มาของ: <ไม่ใช่ส่วนหนึ่งของในบันรอง>
หมายเลขบุกรุก: 00:E7:68:41:A1:55:56:47:F9

รายการเบื้องต้น:
การเชื่อมต่อ:
ที่มาของ: www.nodejsthailand.com
ที่มาของ: <ไม่ใช่ส่วนหนึ่งของในบันรอง>
ที่มาของ: <ไม่ใช่ส่วนหนึ่งของในบันรอง>
หมายเลขบุกรุก: 00:E7:68:41:A1:55:56:47:F9

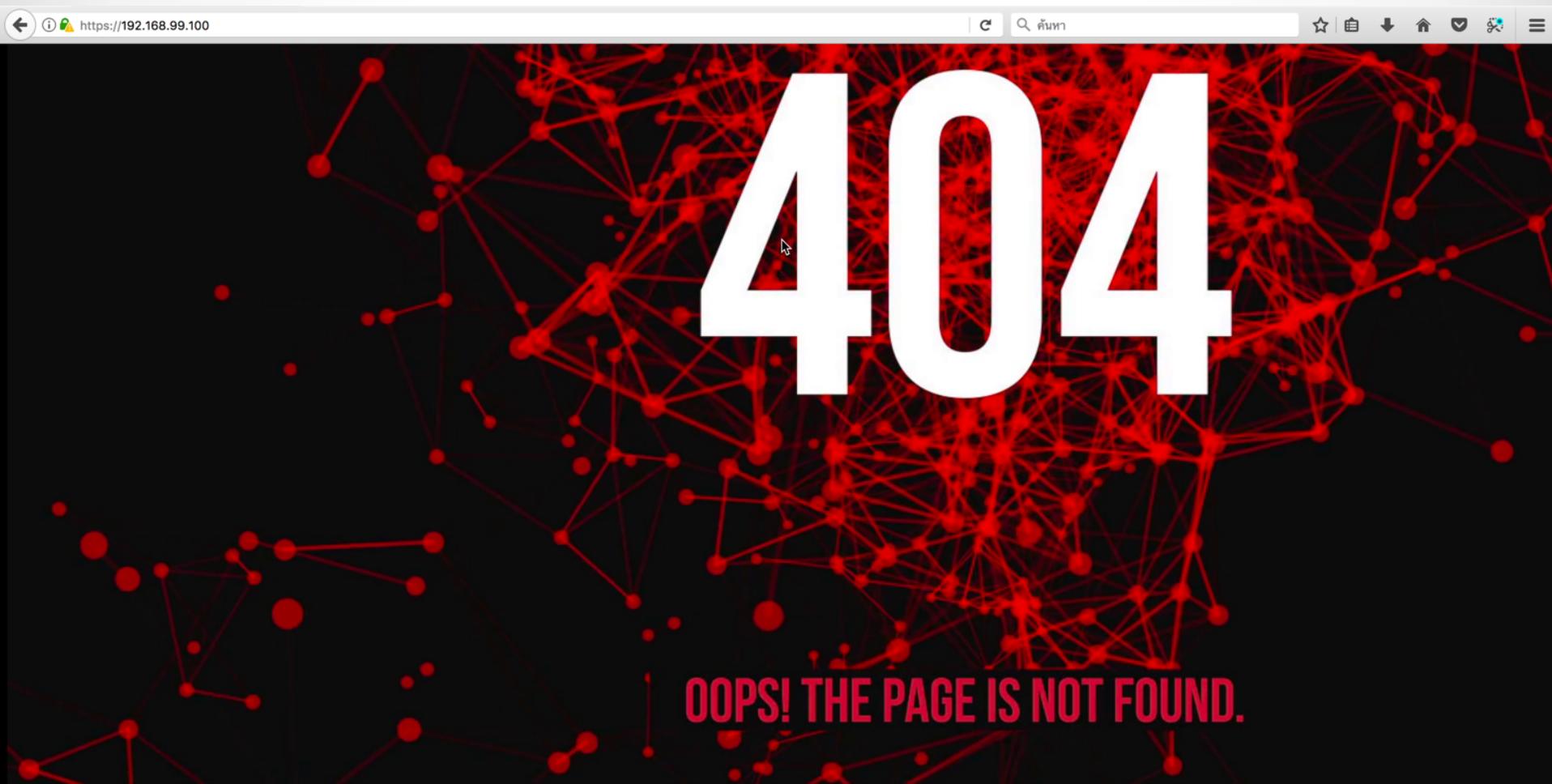
รายการเบื้องต้น:
การเชื่อมต่อ:
ที่มาของ: www.nodejsthailand.com
ที่มาของ: <ไม่ใช่ส่วนหนึ่งของในบันรอง>
ที่มาของ: <ไม่ใช่ส่วนหนึ่งของในบันรอง>
หมายเลขบุกรุก: 30 ธันวาคม 2560
หมดอายุเมื่อ: 28 ธันวาคม 2570

ลายเซ็น:
ลายเซ็น SHA-256 01:14:4A:B8:C7:C6:C8:DD:9D:F1:DC:7E:D4:D4:9A:FC:9A:
B4:00:8E:CD:DR:R7:46:30:FR:R0:C6:5A:16:CD:5R:0D

Docker 101

WorkShop

Output: <https://localhost> , https://x.x.x.x



WorkShop

- DAEMON LINE BOT



ธนาคารแห่งประเทศไทย

ข้อมูลที่นำไปใช้ของ API

ประเภทของ API :	REST
ประเภทของข้อมูล :	JSON
ความปลอดภัย :	PUBLIC

คำแนะนำที่อยู่ของ API

https://iapi.bot.or.th/Stat/Stat-ReferenceRate/DAILY_REF_RATE_V1/

พารามิเตอร์ใน Header

api-key : xk3h6ash-ahw5l3mch6hl3-2hhg6l4mng2346l34l6

ข้อมูลพารามิเตอร์สำหรับการใช้งาน API

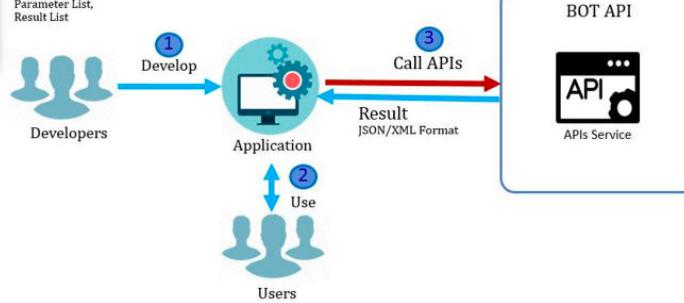
ชื่อตัวแปร	ประเภท	ค่าอธิบาย
start_period	Datetime	วันที่เริ่มคุณสมบัติข้อมูล (YYYY-MM-DD) เช่น 2017-06-30
end_period	Datetime	วันที่สิ้นสุดคุณสมบัติ (YYYY-MM-DD) เช่น 2017-06-30

ข้อมูลพารามิเตอร์ที่จะได้รับ

ชื่อ	ประเภท	ค่าอธิบาย
report_name_eng	String	ชื่อรายงาน ภาษาอังกฤษ
rereport_name_th	String	ชื่อรายงาน ภาษาไทย
report_uoq_name_eng	String	หน่วย ภาษาอังกฤษ
report_uoq_name_th	String	หน่วย ภาษาไทย
report_source_of_data	String	แหล่งมาของข้อมูล
report_remark_eng	String	หมายเหตุ รายงานภาษาอังกฤษ
report_remark_th	String	หมายเหตุ รายงานภาษาไทย
last_updated	String	วันที่แก้ไขข้อมูลล่าสุด
period	String	วันที่รวมข้อมูล
rate	String	อัตรา



BOT API Services Process

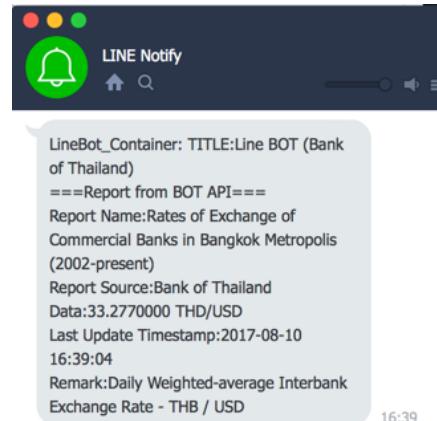


WorkShop

- DAEMON LINE BOT

```
[docker@labdocker:~$ docker container run --rm --name linebot -e "TITLE=Line BOT (Bank of Thailand)" -e "TOKEN=E1eqyillzkzpCo1ROrebZTcGbIyzDZhP0jcd0t6CX" labdocker/linenotify:bot_v1
null
{"statusCode":200,"body":"{\\"status\\":200,\\"message\\":\\"ok\\","headers":{"server":"nginx","date":"Sat, 20 Jan 2018 03:12:22 GMT","content-type":"application/json; charset=UTF-8","transfer-encoding":"chunked","connection":"keep-alive","keep-alive":"timeout=3","x-ratelimit-limit": "1000", "x-ratelimit-imagelimit": "50", "x-ratelimit-remaining": "998", "x-ratelimit-imageremain": "50", "x-ratelimit-reset": "1516421345"}, "request": {"uri": {"protocol": "https:", "slashes": true, "auth": null, "host": "notify-api.line.me", "port": 443, "hostname": "notify-api.line.me", "hash": null, "search": null, "query": null, "pathname": "/api/notify", "path": "/api/notify", "href": "https://notify-api.line.me/api/notify"}, "method": "POST", "headers": {"Content-Type": "application/x-www-form-urlencoded", "authorization": "Bearer E1eqyillzkzpCo1ROrebZTcGbIyzDZhP0jcd0t6CX", "content-length": 466}}}
{\\"status\\":200,\\"message\\":\\"ok\\"
docker@labdocker:~$ ]
```

```
1 const request = require('request');
2 const qs = require('querystring');
3 const TITLE=process.env.TITLE;
4 const KEY =process.env.KEY; //Input API-KEY to Header
5 //const KEY="U9G1L457H60CugT7VmBaEacbHV9RX0Py5005cYaGsm";
6 const URL =process.env.URL; //Input URL for Request
7 //const URL="https://japi.bot.or.th/Stat/Stat-ReferenceRate/DAILY_REF_RATE_V1/?";
8 //const TOKEN = 'E1eqyillzkzpCo1ROrebZTcGbIyzDZhP0jcd0t6CX';
9 const TOKEN=process.env.TOKEN; //Input TOKEN LINE
10 //Set Date
11 var dateFormat = require('dateformat');
12 var daynow = new Date(); // Today!
13 daynow.setDate(daynow.getDate() - 1); // Yesterday!
14 var daynow=dateFormat(daynow, "yyyy-mm-dd");
15 var DAYSTARTPERIOD=daynow;
16 var DAYENDPERIOD=daynow;
17 //Set Date
18 var stickerPkg=2; //stickerPackageId
19 var stickerId=161; //stickerId
20 var Message="";
21 //Setup QueryString
22 var queryString = qs.stringify({
23   start_period: DAYSTARTPERIOD,
24   end_period: DAYENDPERIOD
25 });
26 //Setup QueryString
27 function callback(error, response, body) {
28   //console.log(URL+queryString);
29   //console.log('Response Code: '+ response.statusCode);
30   //console.log(body)
31   if (!error && response.statusCode == 200) {
32     var info = JSON.parse(body);
```



16:39



WorkShop

- DAEMON LINE BOT
- สั่งรัน container แบบ Daemon โดยตั้งชื่อว่า linebot เพื่อทำการดึงข้อมูลจาก API ของธนาคารแห่งประเทศไทยอุปกรณ์แล้วส่ง LINE Notify ไปหา Token Key ที่กำหนด

```
docker container run -it --rm --name linebot \
-e TITLE="Line BOT (Bank of Thailand)" \
-e "TOKEN=<LINE TOKEN>" \
labdocker/linenotify:bot_v1
```

The brief history of Docker & K8S

• • •

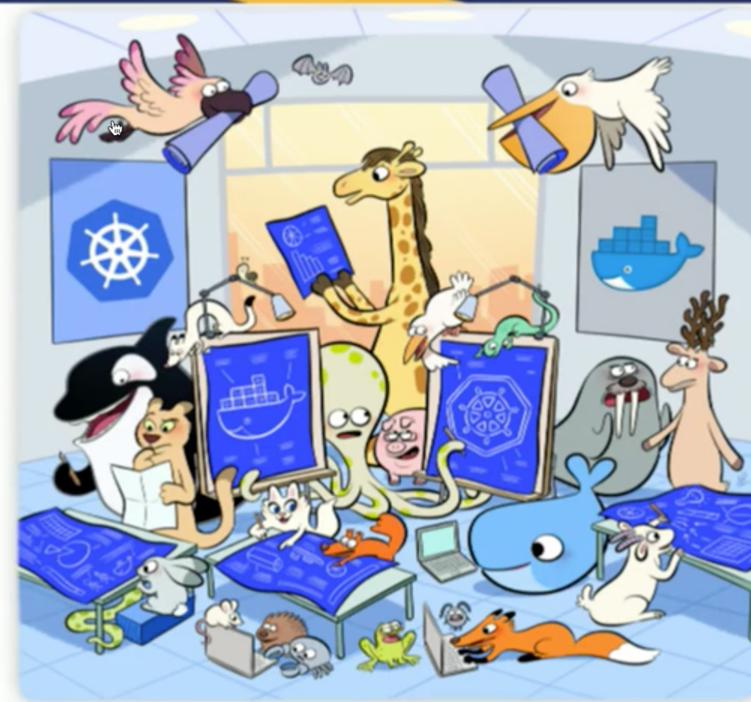
The brief history of Docker & K8S



The brief history of Docker & K8S

- “We are one big community”

WE ARE
ONE BIG
COMMUNITY



docker
con 17 EU

The brief history of Docker & K8S

- Official support orchestrator both K8S and Swarm

Docker with Swarm and Kubernetes

1 →

The best enterprise
container security and
management

Docker Enterprise Edition

Docker Community Edition



2 ←

The best container
development workflow

3 →

Native Kubernetes
integration provides full
ecosystem
compatibility

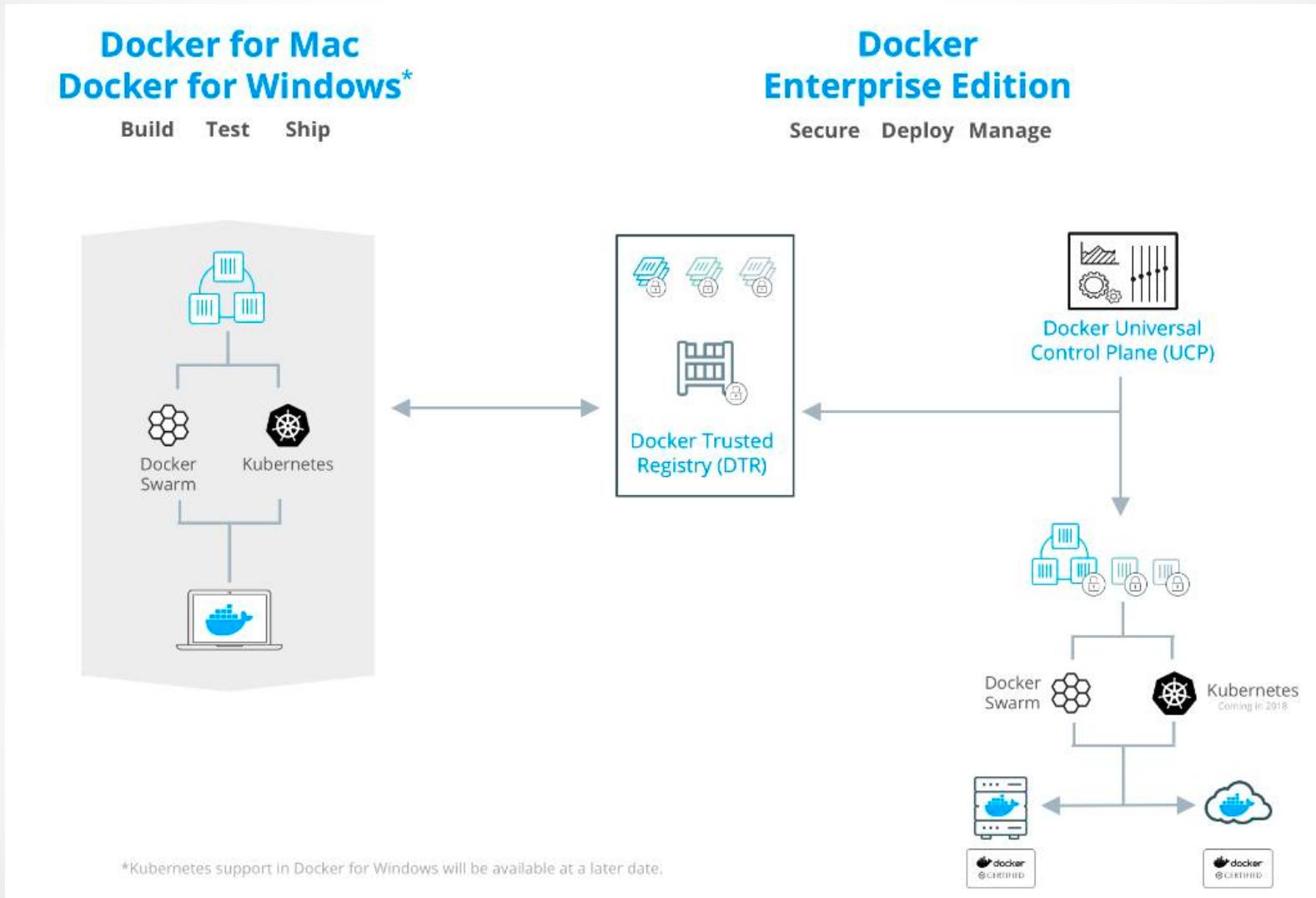
containerd

4 ←

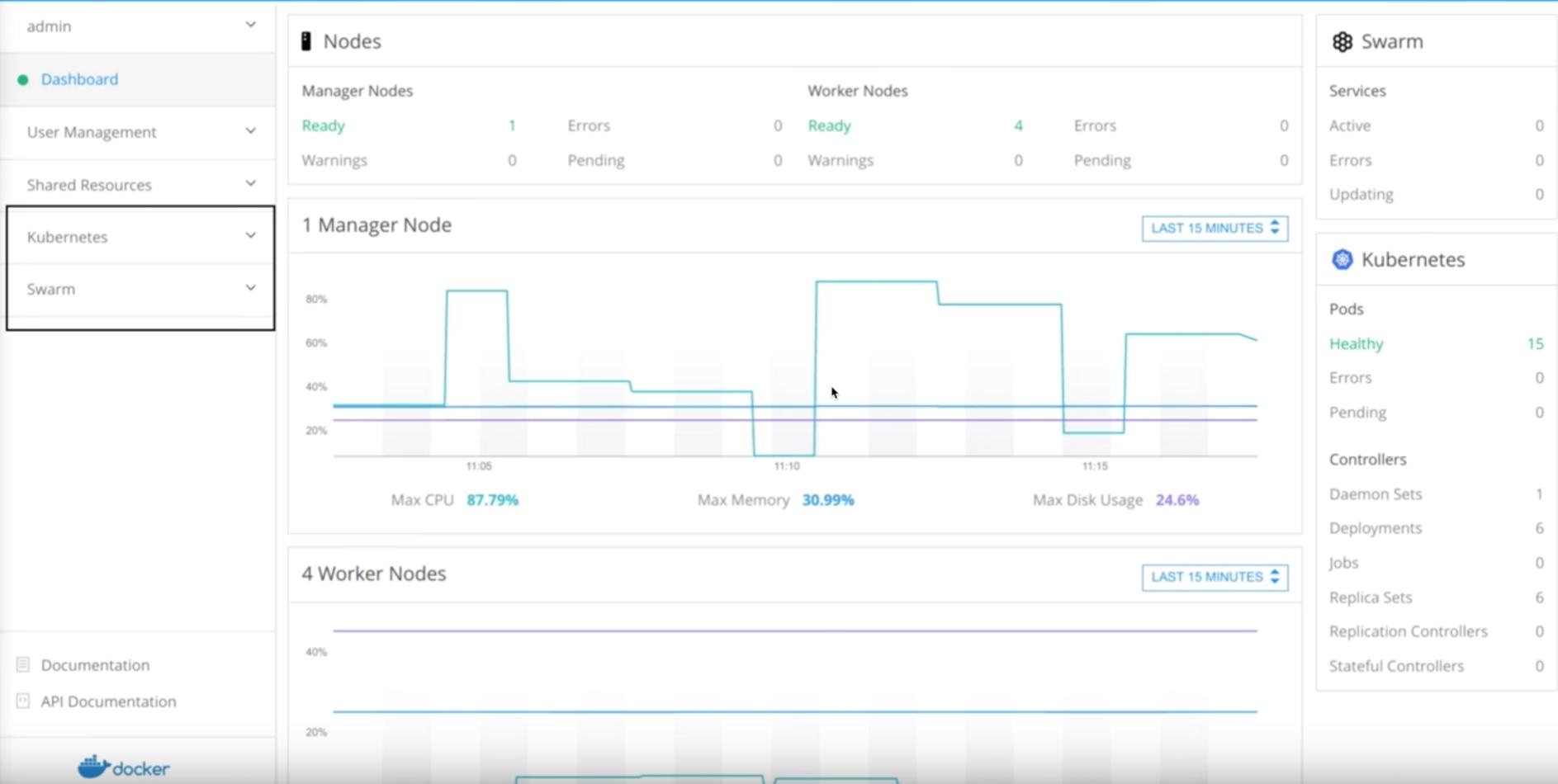
Industry-standard
container runtime

dockercon¹⁷ EU

The brief history of Docker & K8S

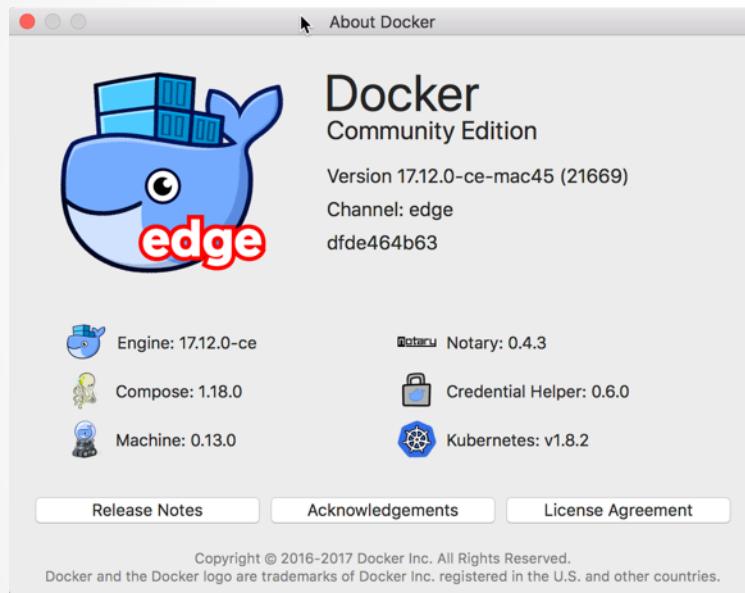


The brief history of Docker & K8S



The brief history of Docker & K8S

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



The brief history of Docker & K8S

What are the best Docker orchestration tools?

11

OPTIONS CONSIDERED

77

RECOMMENDATIONS

Mar 13, 2018

LAST UPDATED

THE BEST 1 OF 11 OPTIONS i 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?

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

Docker 101





By Praparn Luengphoonlap
Email: eva10409@gmail.com

Q&A

Docker 101

