



Docker:Zero to Hero (Day 1)

By Praparn Luengphoonlap
Email: praparn@opcellent.com

Docker: The Next-Gen of Virtualization



1

Outline Day 1

- Docker principle
- Docker machine
- Image, Repository & Tag, container
- CPU, Memory and I/O
- Network
- Volume
- Inspect and Log
- Commit
- Kitematic (Alpha)

Docker: The Next-Gen of Virtualization



2

Outline Day 2

- Dockerfile and Build
- Compose
- Docker Security
- Registry
- Swarm Mode
 - Conceptual of Swarm
 - Swarm Mode Architecture
 - Swarm init/join cluster system
 - Swarm service
 - Orchestrator Assignment
 - Config and Secret
 - Network Overlay and Ingress
 - HA Manager Role
 - Docker Stack Deploy (Compose Swarm Mode)
- portainer for Docker
- Q&A

Docker: The Next-Gen of Virtualization



3

Prerequisite

- Windows (64 bit) / Mac / Linux (64 bit) machine (ubuntu / alpine prefer)
- 1 email address (For register "hub.docker.com") / hub.docker.com account
- Line Notify Token (<https://notify-bot.line.me>)
- 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

Docker: The Next-Gen of Virtualization



4

Prerequisite

- Generate LINE Token
 - <https://notify-bot.line.me>

The screenshot shows the LINE Notify API documentation page. At the top, there's a 'Connect LINE with Everything' button. Below it, a large text area says 'รับการแจ้งเตือนจากเว็บไซต์ของ LINE'. To the right, there's a 'ออก Access Token (สำหรับผู้พัฒนา)' button. The URL in the browser is https://notify-bot.line.me/doc.

Docker: The Next-Gen of Virtualization



5

Prerequisite

- Generate LINE Token
 - <https://notify-bot.line.me>

The screenshot shows the 'Token ที่ออก' (Generated Token) section of the LINE Notify API interface. It displays a token string: zHOlcJCcpIIS8mEedn. Below it, a message from a Docker container named 'LineBot_Container' is shown, sent to 'praparn' at 2017.07.19 23:42. The message contains two attachments: a gear icon and a sunset image.

Docker: The Next-Gen of Virtualization



6

Lab Resource

- Repository for lab

Docker: The Next-Gen of Virtualization



7

Lab Resource

- Software in lab

Docker: The Next-Gen of Virtualization



8

Lab Resource

- Download on Google Drive
 - <https://tinyurl.com/yxb2fefo>
- Download on GitHub
 - `git clone https://github.com/praparn/docker-workshop-072020.git`

Docker: The Next-Gen of Virtualization

9

Lab Resource

- <https://tinyurl.com/y9svjua2>

Docker: The Next-Gen of Virtualization

10



11

The slide features a blue header with "DockerCon 2019" and a photo of a stage with a speaker. Below the header, text provides event details: "Registration and Welcome Reception: April 29", "Full Conference: April 30 – May 2", and "Location: Moscone Center, San Francisco, CA". To the right is a photo of six people standing on stage, and at the bottom right is the Docker logo.

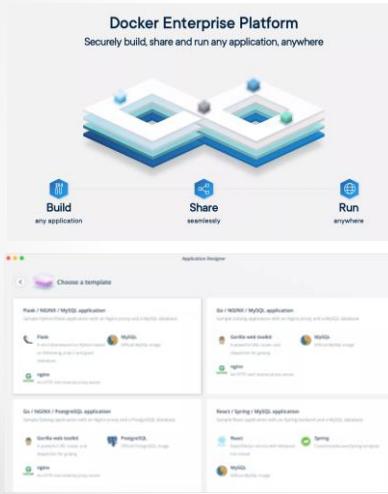
[Ref: https://www.docker.com/customers/innovation-awards?utm_campaign=IT+Pro&utm_content=1560297378&utm_medium=social&utm_source=Organic&fbclid=IwAR0a7HfuForGzX-o3ziYGnBd02ijjRfV6_Hil-sdiX23HtPXBqXoo](https://www.docker.com/customers/innovation-awards?utm_campaign=IT+Pro&utm_content=1560297378&utm_medium=social&utm_source=Organic&fbclid=IwAR0a7HfuForGzX-o3ziYGnBd02ijjRfV6_Hil-sdiX23HtPXBqXoo)

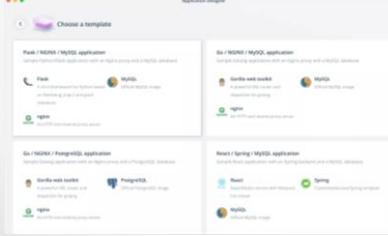
Docker: The Next-Gen of Virtualization

12

DockerCon2019

- Docker Enterprise (Acquired by Mirantis)






Docker: The Next-Gen of Virtualization

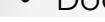
Integrated Kubernetes from Desktop to Production



13

DockerCon2019

- Docker Enterprise (Acquired by Mirantis)




What We Announced Today and Why it Matters

Adrian Ionel - November 13, 2019 - containers | docker | k8s | kubernetes | mirantis

Today we announced that we have acquired the Docker Enterprise platform business from Docker, Inc. including its industry leading Docker Enterprise and 750 customers.

Why Docker Enterprise, and Why Now?

Docker led the container revolution and changed the world towards a better way to build, share and run software. No infrastructure company has had a bigger impact on developers in the last decade. Docker Enterprise is the only independent container platform that enables developers to seamlessly build, share and safely run their applications anywhere – from public cloud, to hybrid cloud to the edge. One-third of Fortune 100 and one-fifth of Global 500 companies use Docker Enterprise. Two years ago Docker Enterprise started to ship Kubernetes as part of its Universal Control Plane and many of its customers are using it today or plan to use it in the near future.

This acquisition will accelerate Mirantis' vision to deliver Kubernetes-as-a-Service with a consistent experience for developers on any cloud and on-prem infrastructure.

[Ref: https://www.mirantis.com/blog/mirantis-acquires-docker-enterprise-platform-business/](https://www.mirantis.com/blog/mirantis-acquires-docker-enterprise-platform-business/)

Docker: The Next-Gen of Virtualization



14

DockerCon2019

- Docker Enterprise (Acquired by Mirantis)

Docker Enterprise

Estimated reading time: 6 minutes

This topic applies to Docker Enterprise.

The Docker Enterprise platform business, including products, customers, and employees, has been acquired by Mirantis, Inc., effective 13-November-2019. For more information on the acquisition and how it may affect you and your business, refer to the Docker Enterprise Customer FAQ.

Docker Enterprise platform

The Docker Enterprise platform is the leading container platform for continuous, high-velocity innovation. Docker Enterprise is the only independent container platform that enables developers to seamlessly build and share any application — from legacy to modern — and operators to securely run them anywhere - from hybrid cloud to the edge.

Docker Enterprise enables deploying highly available workloads using either the Docker Kubernetes Service or Docker Swarm. Docker Enterprise automates many of the tasks that orchestration requires, like provisioning pods, containers, and cluster resources. Self-healing components ensure that Docker Enterprise clusters remain highly available.

Role-based access control (RBAC) applies to Kubernetes and Swarm orchestrators, and communication within the cluster is secured with TLS. Docker Content Trust is enforced for images on all of the orchestrators.

Docker Enterprise includes Docker Universal Control Plane (UCP), the cluster management solution from Docker. UCP can be installed on-premises or in your public cloud of choice, and helps manage your cluster and applications through a single interface.

Docker: The Next-Gen of Virtualization



15

DockerCon2019

- Docker Foundation

Docker Foundation

Transforming the world through inclusive access to innovative technology and education.

Pledge 1%

Docker joins Pledge 1%, a global philanthropy movement dedicated to giving back to those in need.

We are committed to donating our time, funding and resources to ensure inclusive access and opportunity for all those who wish to participate in the digital economy.

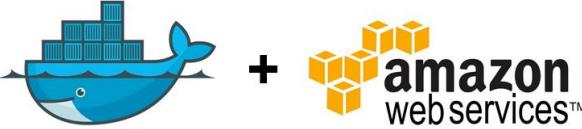
**PLEDGE
1%**

Docker: The Next-Gen of Virtualization



16

Workshop: Access Docker on AWS



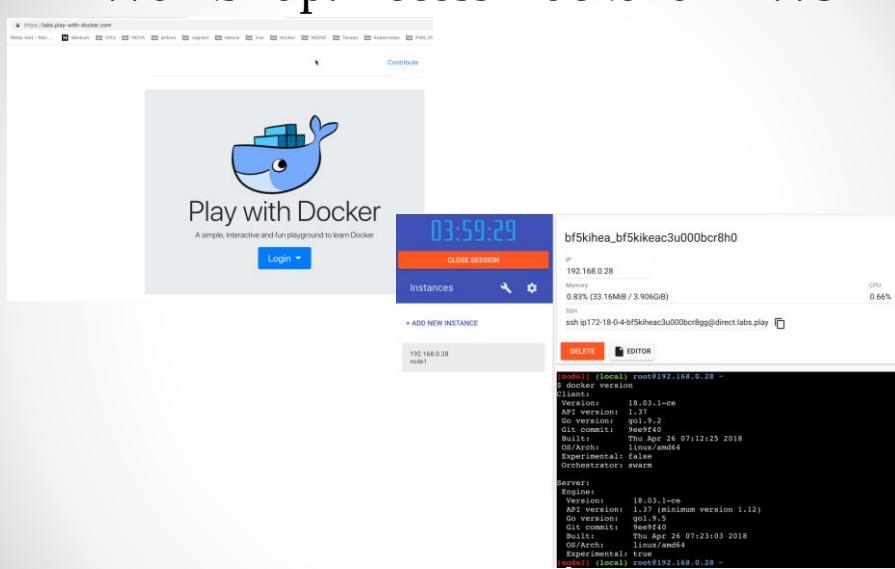
Docker on AWS

Docker: The Next-Gen of Virtualization



17

Workshop: Access Docker on AWS



Docker: The Next-Gen of Virtualization



18

Who are we ? (Opcellent)

Kubernetes
ໝາກໜີຂອບບັນດາການຈັດທາ
ໃຫ້ຄົນເວົ້າໄດ້ ແລະ ຄຸນທີ່ມາຈັດກັນ

Paparn Lungpoonlap
CEO Opcellent

- ...
- ...
- Is it necessary
- Can I solve
- ...

DevOpsCon 2018

aws datavow Google Cloud ODDS Opcellent

Docker: The Next-Gen of Virtualization

19

Landscape of the world now

CONTAINERS ARE NOW MAINSTREAM AND USAGE IS ONLY GROWING.

130B
Total Pulls on Hub

8B
Pulls in the past month*

6M
Repositories on Hub

5M
Hub Users

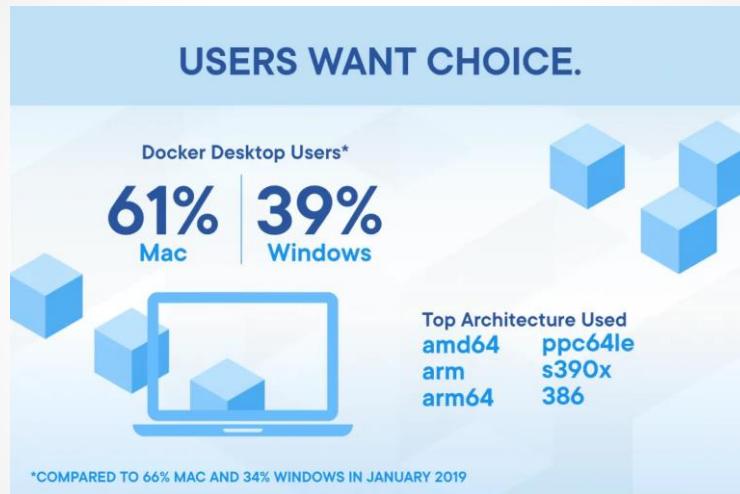
2.4M
Desktop Installations

*UP FROM 5.5 B A YEAR AGO

Docker: The Next-Gen of Virtualization

20

Landscape of the world now

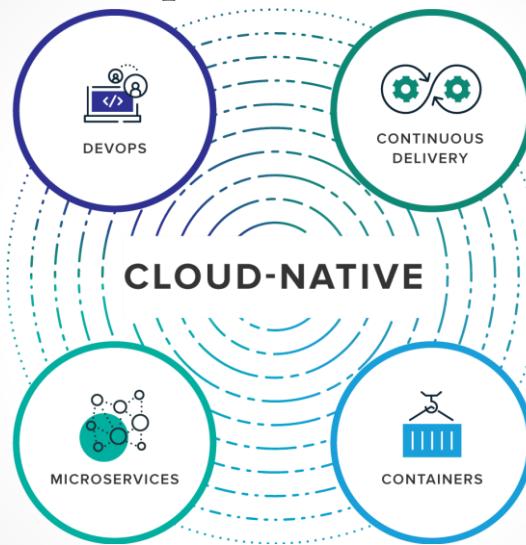


Docker: The Next-Gen of Virtualization



21

Landscape of the world now



Docker: The Next-Gen of Virtualization



22

Landscape of the world now

 sysdig

Key 2019 Insights

52% of containers live 5 minutes or less	2x the number of containers alive for 10 seconds or less	100% increase in container density year over year
 Go and Node.js overtake Java as top cloud app frameworks 		
 Prometheus rises to lead custom metric solution	 Red Hat OpenShift is top choice for secure, on-prem Kubernetes	Containers frequently run as root and in privileged mode

2019 Container Usage Report

Five minute container life highlights need for specific security controls

Ref: https://sysdig.com/blog/sysdig-2019-container-usage-report/?fbclid=IwAR06qOb-M7imoS9FLmQOPy5VQZIXxQYu2htTz9meL_XLdB5yn8sVuYwACg



23

Landscape of the world now

Orchestrators

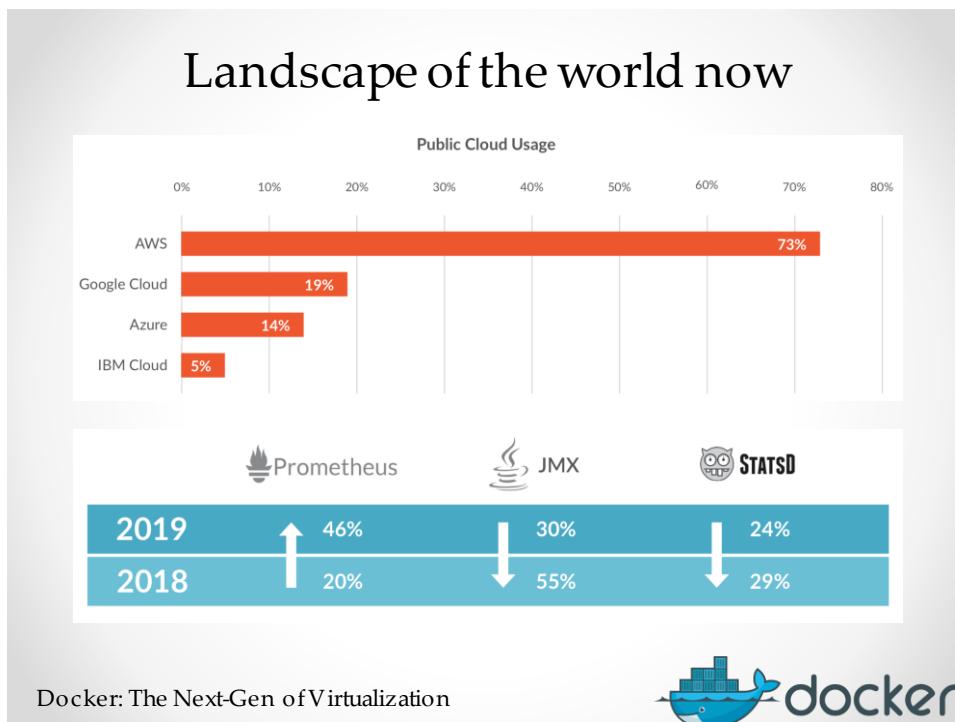
Orchestrator	Percentage
Kubernetes	77%
OpenShift	9%
Swarm	5%
Mesos	4%
Rancher	3%
Amazon ECS	2%

Container Registries

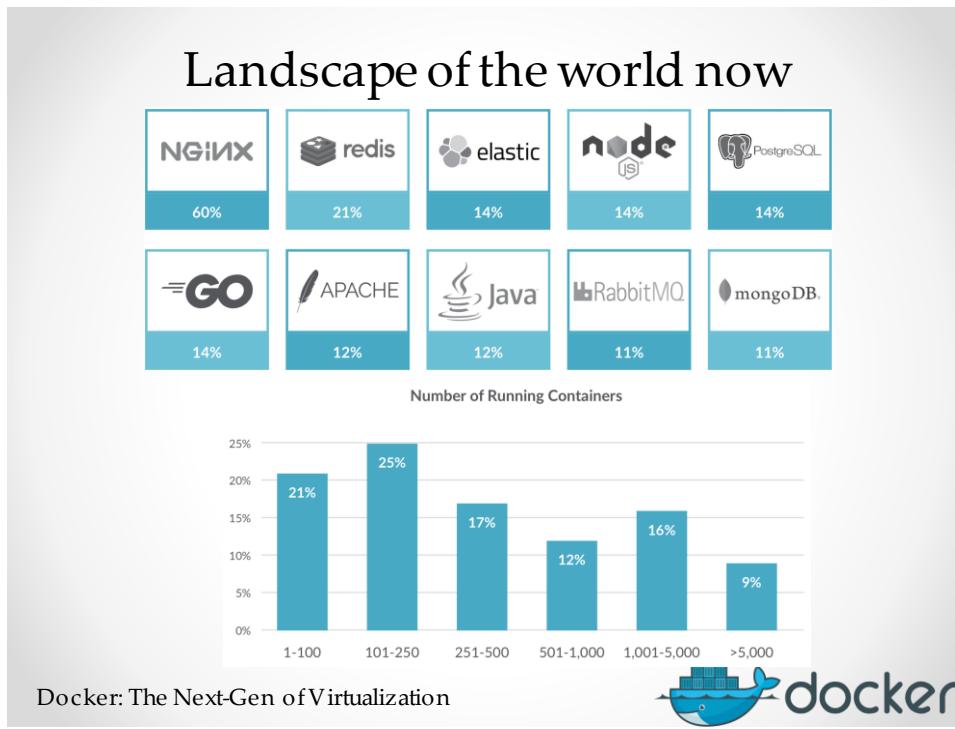
Registry	Percentage
Docker	34%
Google's GCR	28%
Quay	14%
AWS ECR	9%
IBM Cloud ICR	6%
Azure ACR	4%
Other	5%

Docker: The Next-Gen of Virtualization 

24

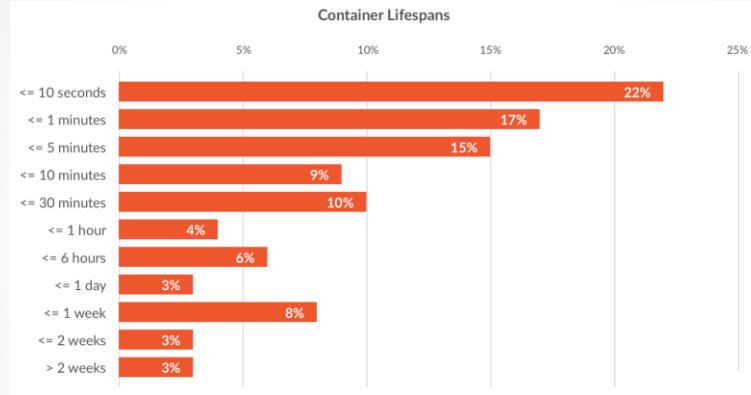


25



26

Landscape of the world now

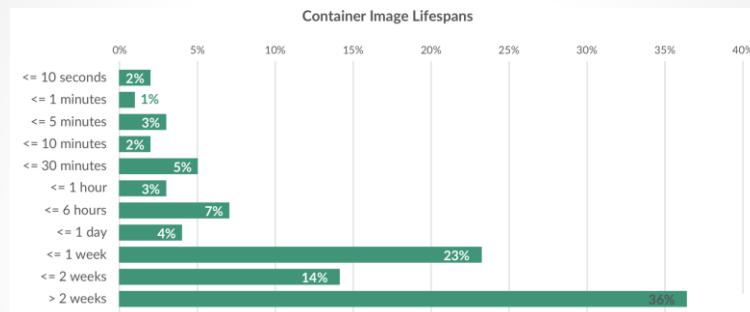


Docker: The Next-Gen of Virtualization



27

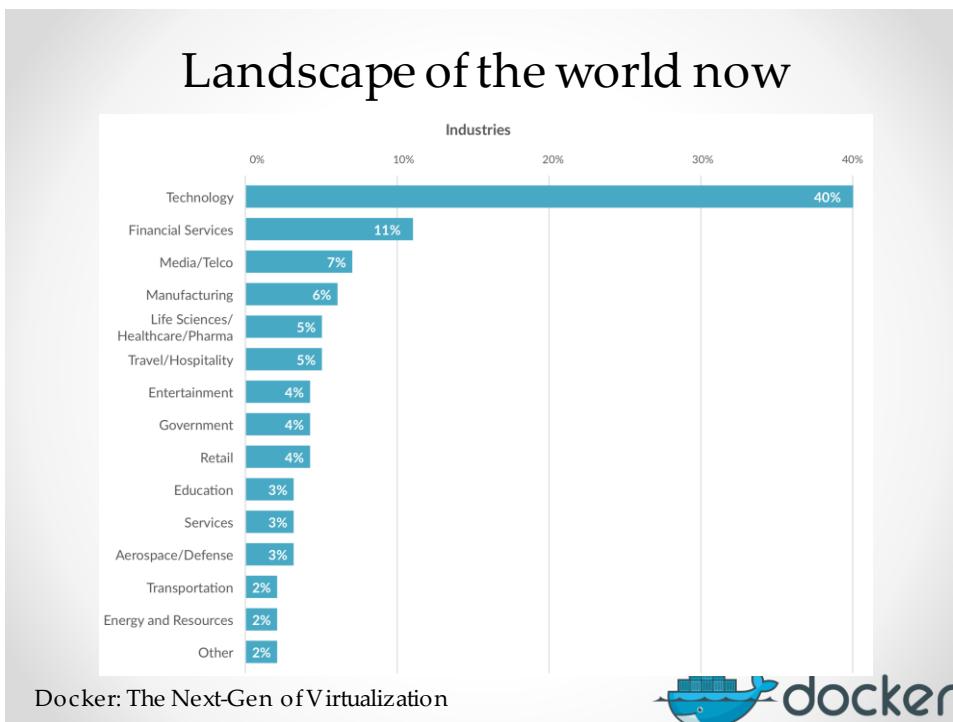
Landscape of the world now



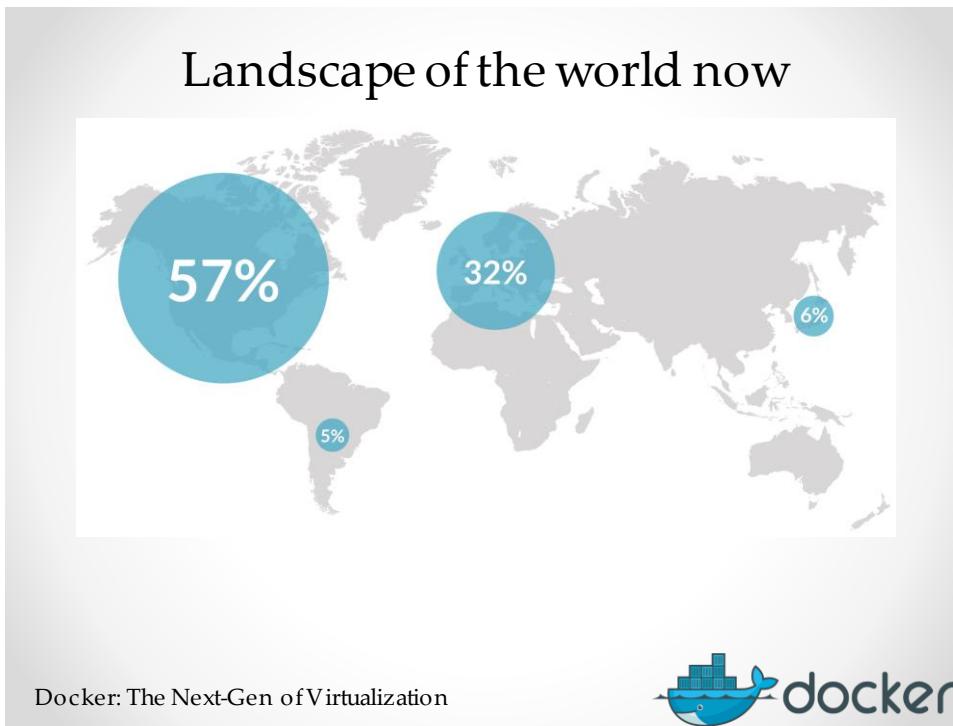
Docker: The Next-Gen of Virtualization



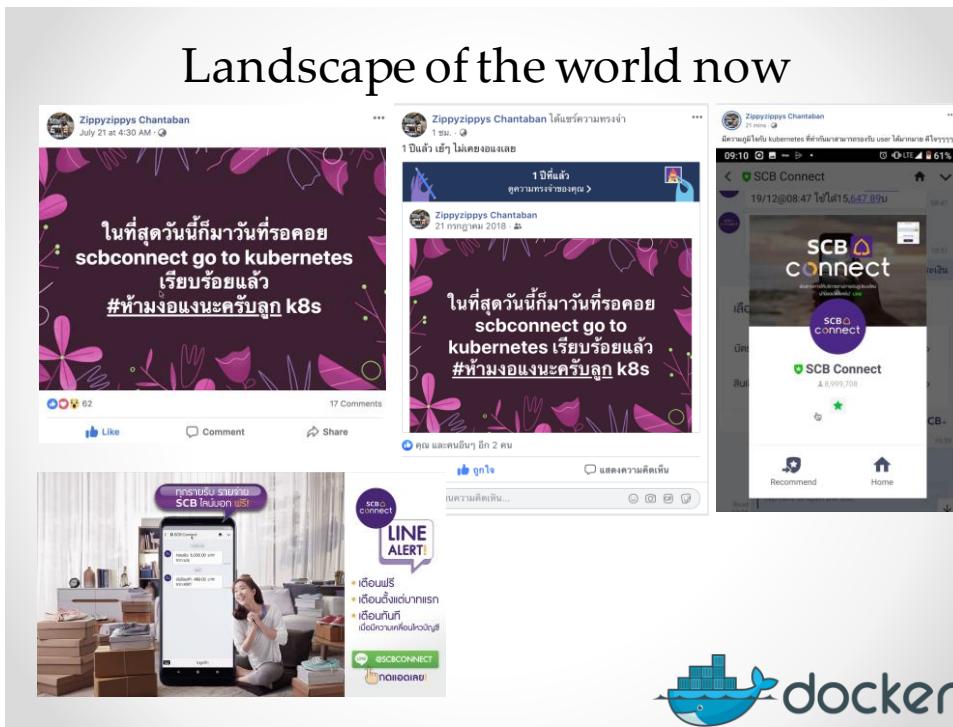
28



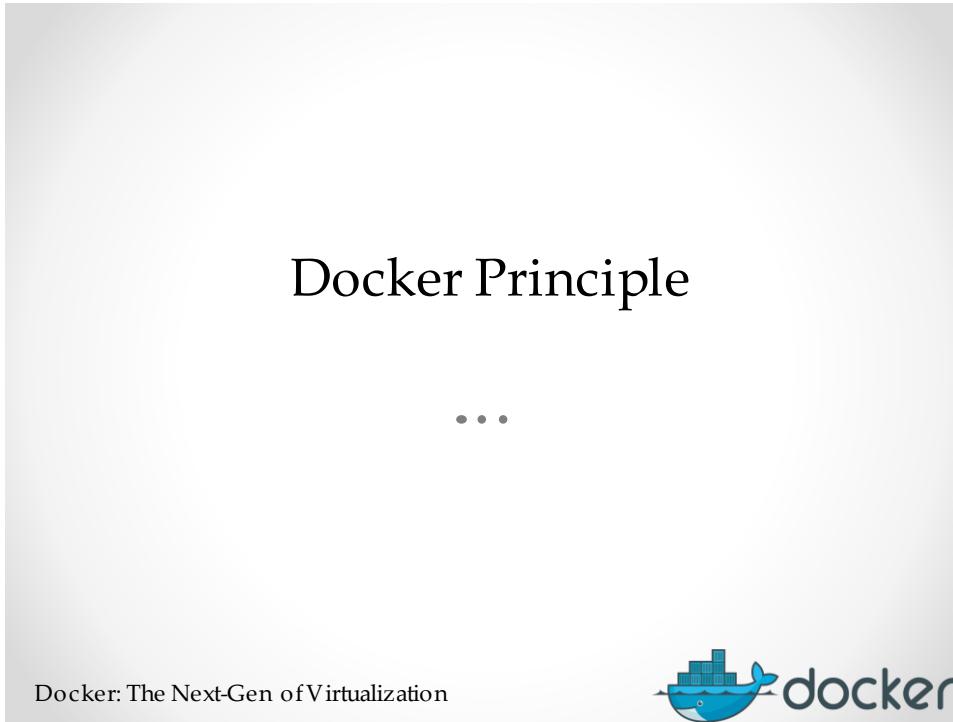
29



30

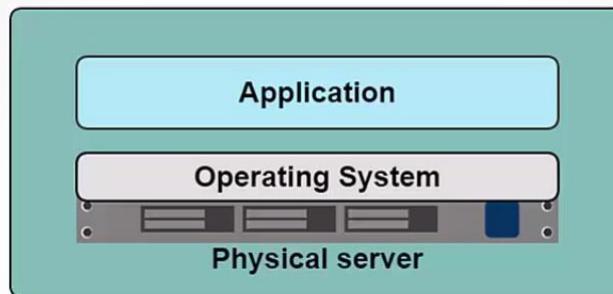


31



32

What is docker ?

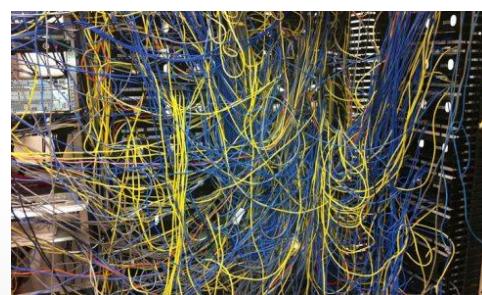


Docker: The Next-Gen of Virtualization



33

Existing Technology



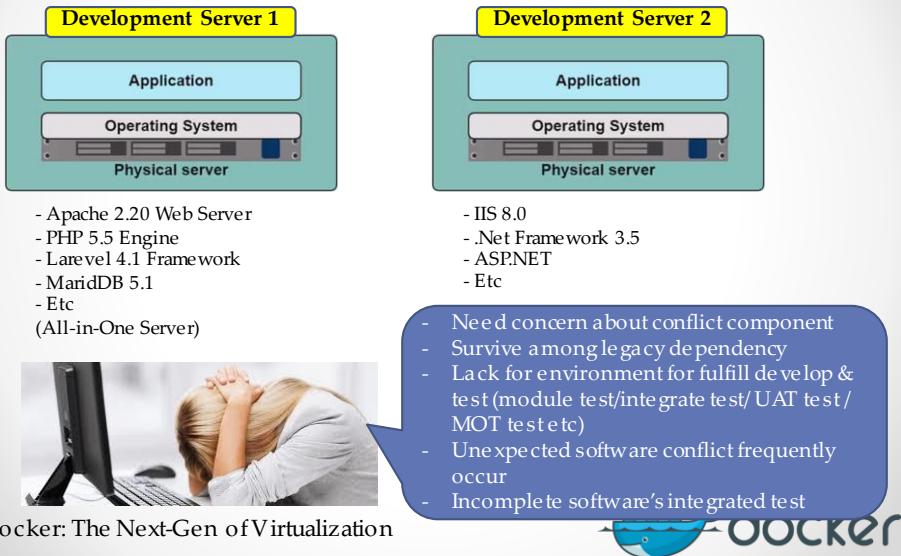
Docker: The Next-Gen of Virtualization



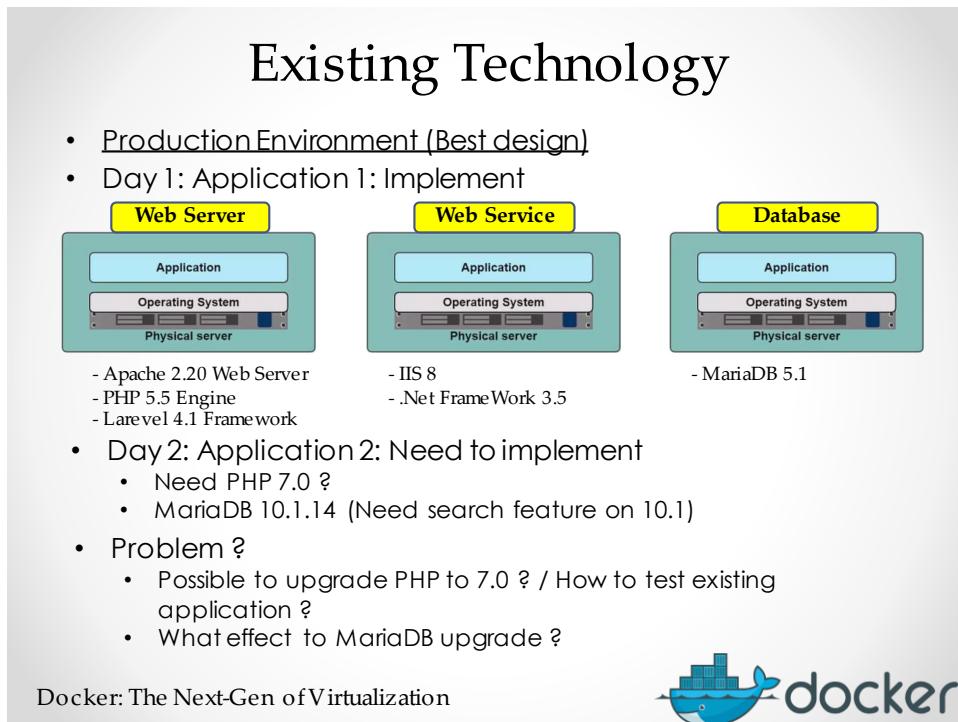
34

Existing Technology

- Development Environment (Mix everything possible)



35



36

Existing Technology

- What operation handle in production environment ?
 - Aware about huge of difference software component between development / production machine
 - Need to know in deep all application dependency (Wow !!!)
 - Take time for discussion and find solution to implement.
 - Possible to confusion and effect to other application that existing on share server.
 - Many unexpected problem & software bug.
 - Hard control software's quality assurance (QA)

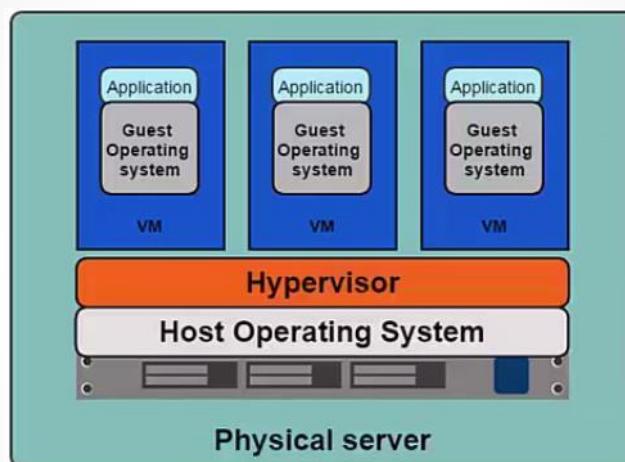


Docker: The Next-Gen of Virtualization



37

What is docker ?



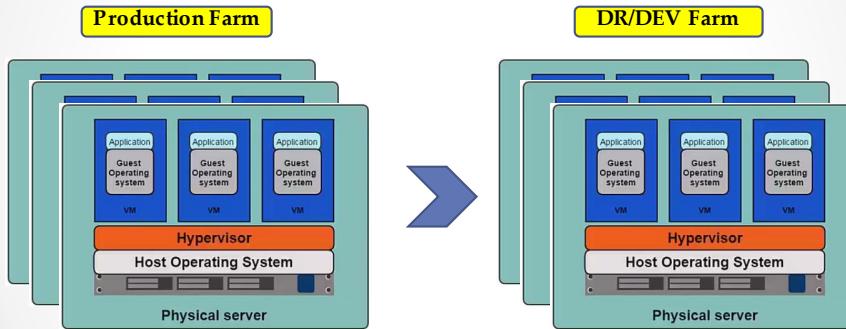
Docker: The Next-Gen of Virtualization



38

Existing Technology

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



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

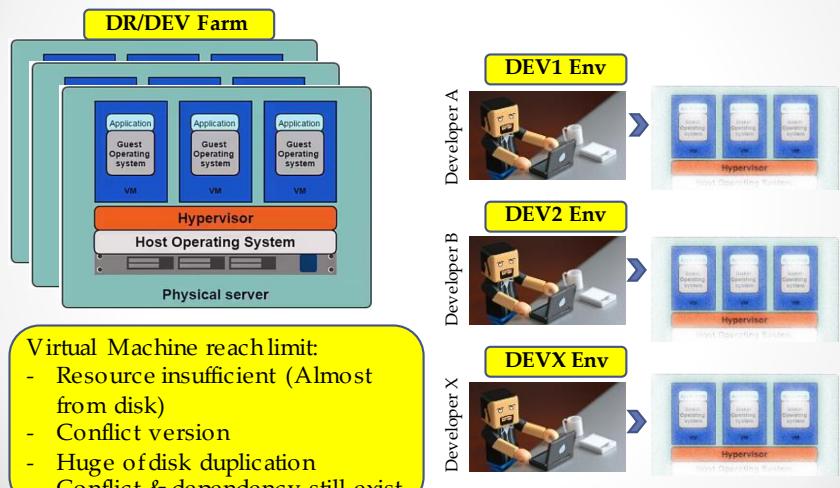
Docker: The Next-Gen of Virtualization



39

Existing Technology

- Development Environment (Clone from Production)



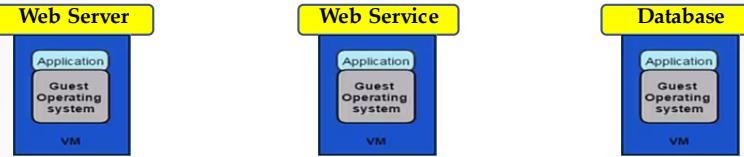
Docker: The Next-Gen of Virtualization



40

Existing Technology

- Production Environment
- Day 1: Application 1: Implement



- Apache 2.20 Web Server
 - IIS 8
 - PHP 5.5 Engine
 - .Net FrameWork 3.5
 - Laravel 4.1 Framework

- MariaDB 5.1

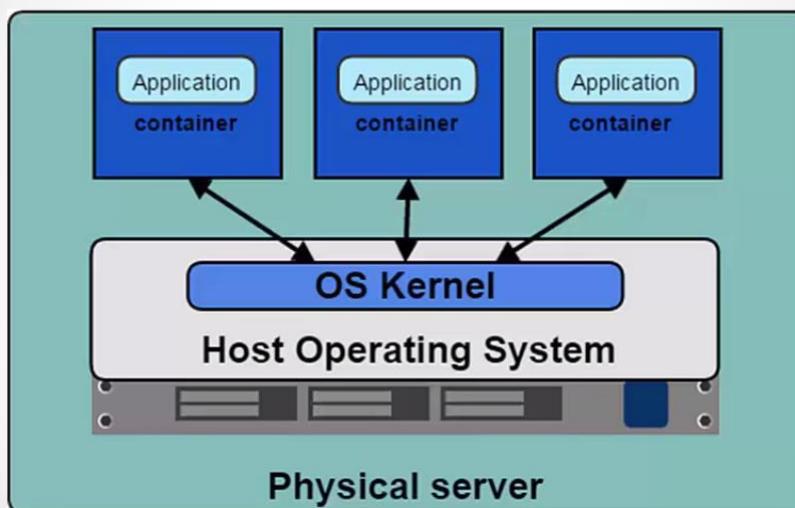
- Day 2: Application 2: Need to implement
 - Need PHP 7.0 ?
 - MariaDB 10.1.14 (Need search feature on 10.1)
- So... The problem still exist.

Docker: The Next-Gen of Virtualization



41

What is docker ?

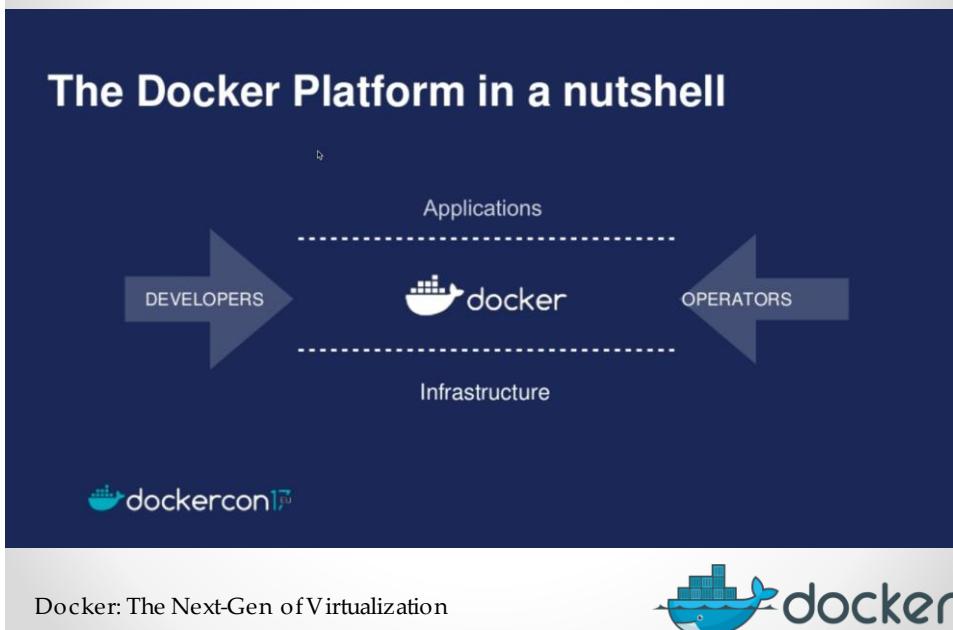


Docker: The Next-Gen of Virtualization

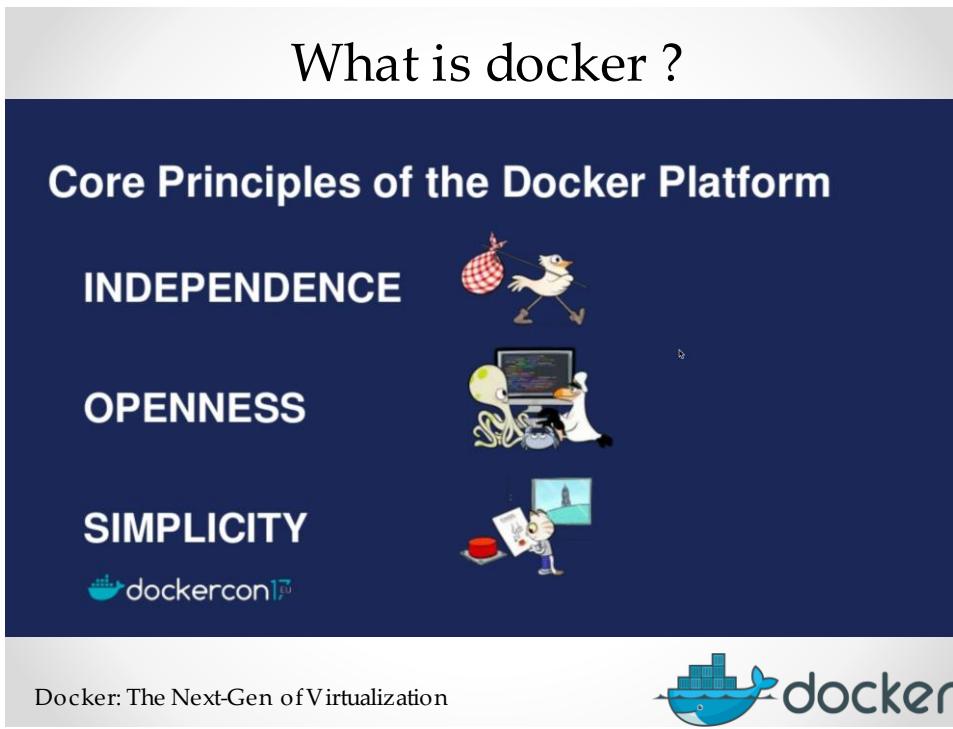


42

What is docker ?



43



44

What is docker ?

- Docker คือ open platform solution ที่ทำงานภายใต้คอนเซ็ปต์ของ container virtualize technology (operating -system –level virtualization)
- ผู้ใช้สามารถสร้างสภาพแวดล้อมเพื่อใช้ในพัฒนาโปรแกรมและส่งมอบเพื่อใช้งานในสภาพเดียวกัน
- Build, Ship, Run
- หมายความว่า Developer, DevOps, Architecture, Engineer
- เขียนภาษา Go
- รองรับการติดตั้งบนลินุกซ์ 64 บิต (kernel 3.1.0) (Official)
 - Ubuntu (X64/ARM/ARM64) (CE/EE Edition)
 - Debian (X64/ARM/ARM64) (CE Edition)
 - Oracle Linux (EE Edition)
 - CentOS (CE/EE Edition)
 - Fedora (CE Edition)
 - Red Hat Enterprise Linux (EE Edition)
 - Microsoft Windows Server 2016 (EE Edition)
 - Microsoft Windows Server 1709 (EE Edition)
 - Microsoft Windows Server 1803 (EE Edition)
 - Suse Linux Enterprise Server (EE Edition)

Docker: The Next-Gen of Virtualization



45

Operating-system-level virtualization

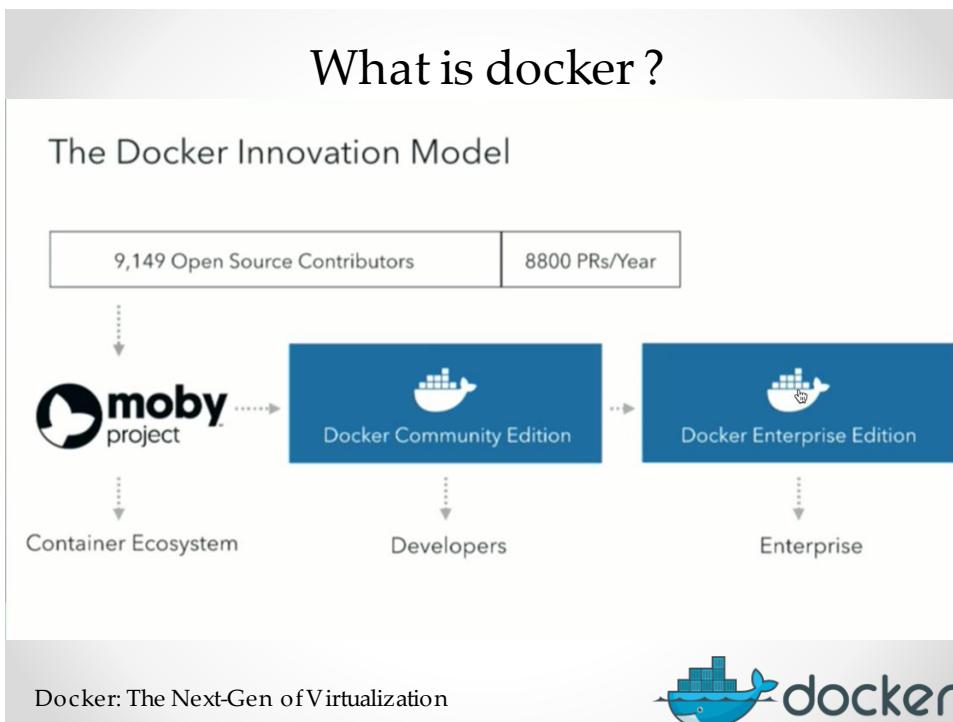
Mechanism	Operating system	License	Available since/between	Features											
				File system isolation	Copy on Write	Disk quotas	I/O rate limiting	Memory limits	CPU quotas	Network isolation	Nested virtualization	Partition checkpointing and live migration	Root privilege isolation		
chroot	most UNIX-like operating systems	varies by operating system	1982	Partial ^[1]	No	No	No	No	No	No	Yes	No	No	No	No
Docker	Linux ^[2] (Security context)	Apache License 2.0 GNU GPLv2	2013 2001	Yes	Yes	Not directly	Not directly	Yes	Yes	Yes	Yes	Yes	No	No	No
Imitay	Linux	Apache License 2.0	2013	Yes	Yes	Yes	Yes ^[3]	Yes	Yes	Yes	Yes	Yes	No	Partial ^[4]	Yes ^[5]
LXC	Linux	Apache License 2.0 GNU GPLv2	2008	Yes	Yes	Partial ^[6]	Partial ^[6]	Yes	Yes	Yes	Yes	Yes	No	Yes ^[6]	Yes ^[6]
LXD	Linux	Apache License 2.0	2015	Yes	Yes	Partial ^[7] (LXC)	Partial ^[7] (LXC)	Yes	Yes	Yes	Yes	Yes	Partial ^[7]	Yes	Yes
OpenVZ	Linux	GNU GPLv2	2005	Yes	No	Yes	Yes ^[8]	Yes	Yes	Yes	Yes	Yes	Partial ^[9]	Yes ^[9]	Yes ^[9]
Virtuozzo	Linux, Windows	Proprietary	2000 ^[10]	Yes	Yes	Yes	Yes ^[11]	Yes	Yes	Yes ^[11]	Yes	Yes	Partial ^[10]	Yes	Yes
Solaris Containers (Zones)	Solaris (OpenSolaris), Solaris	CDDL Proprietary	2004	Yes	Yes (ZFS)	Yes	Partial ^[12]	Yes	Yes	Yes ^[13]	Yes	Yes	Partial ^[12]	Yes ^[12]	Yes ^[12]
FreeBSD (all)	FreeBSD	BSD License	2000 ^[14]	Yes	Yes (ZFS)	Yes ^[15]	No	Yes ^[16]	Yes	Yes ^[16]	Yes	Yes	No	Yes ^[16]	Yes ^[16]
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	No	?	?
WPARs	AIX	Proprietary	2007	Yes	No	Yes	Yes	Yes	Yes	Yes	Yes	Yes	No	Yes ^[17]	?
HP-UX Confinement (SRPf)	HP-UX	Proprietary	2007	Yes	No	Partial ^[18]	Yes	Yes	Yes	Yes	Yes	Yes	?	Yes	?
iCore Virtual Accounts	Windows XP	Proprietary/Freeware	2008	Yes	No	Yes	No	No	No	No	?	No	No	?	?
Sandboxie	Windows	Proprietary/Shareware	2004	Yes	Yes	Partial	No	No	No	Partial	No	No	No	Yes	Yes
Spoon	Windows	Proprietary	2012	Yes	Yes	No	No	No	No	Yes	No	No	No	Yes	Yes
VMware ThinApp	Windows	Proprietary	2008	Yes	Yes	No	No	No	No	Yes	No	No	No	Yes	Yes

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

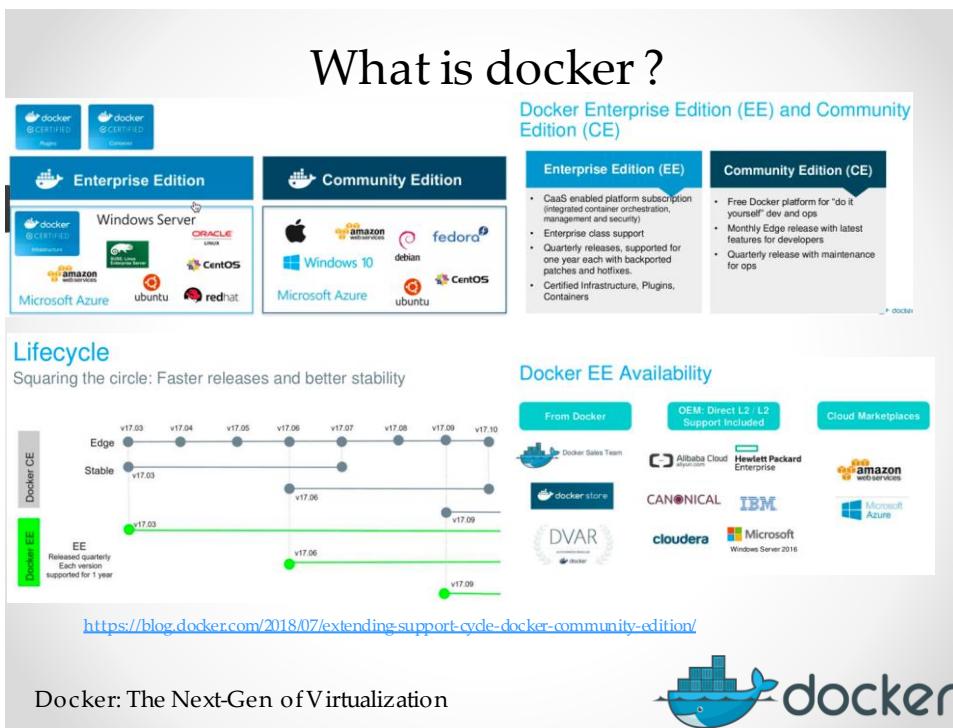
Docker: The Next-Gen of Virtualization



46



47



48

What is docker ?

About Docker CE

Estimated reading time: 7 minutes

Docker Community Edition (CE) is ideal for developers and small teams looking to get started with Docker and experimenting with container-based apps. Docker CE has three types of update channels, **stable**, **test**, and **nightly**:

- **Stable** gives you latest releases for general availability.
- **Test** gives pre-releases that are ready for testing before general availability.
- **Nightly** gives you latest builds of work in progress for the next major release.

For more information about Docker CE, see [Docker Community Edition](#).

Releases

For the Docker CE engine, the open repositories [Docker Engine](#) and [Docker Client](#) apply.

Releases of Docker Engine and Docker Client for general availability are versioned using dotted triples. The components of this triple are `YY.mm.<patch>` where the `YY.mm` component is referred to as the year-month release. The version numbering format is chosen to illustrate cadence and does not guarantee SemVer, but the desired date for general availability. The version number may have additional information, such as beta and release candidate qualifications. Such releases are considered “pre-releases”.

The cadence of the year-month releases is every 6 months starting with the `18.09` release. The patch releases for a year-month release take place as needed to address bug fixes during its support cycle.

Docker CE binaries for a release are available on [download.docker.com](#) as packages for the supported operating systems. Docker EE binaries are available on the [Docker Hub](#) for the supported operating systems. The release channels are available for each of the year-month releases and allow users to “pin” on a year-month release of choice. The release channel also receives patch releases when they become available.

[Ref: http://dockerdocs.gclearning.cn/install/](#)

Docker: The Next-Gen of Virtualization



49

What is docker ?

- o DOCKER CE

DESKTOP

Platform	x86_64
Docker Desktop for Mac (macOS)	✓
Docker Desktop for Windows (Microsoft Windows 10)	✓

SERVER

Platform	x86_64 / amd64	ARM	ARM64 / AARCH64	IBM Power (ppc64le)	IBM Z (s390x)
CentOS	✓		✓		
Debian	✓	✓	✓		
Fedor	✓		✓		
Ubuntu	✓	✓	✓	✓	✓

Docker: The Next-Gen of Virtualization



50

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

Solomon Hykes
CTO and Founder
dockercon17
Docker: The Next-Gen of Virtualization

docker

51

Growth of Docker

By 2020, more than **50%** of global organizations will be running containers in production.

Usage Among Adopters
Docker deployment size has increased 10x IN ONE YEAR

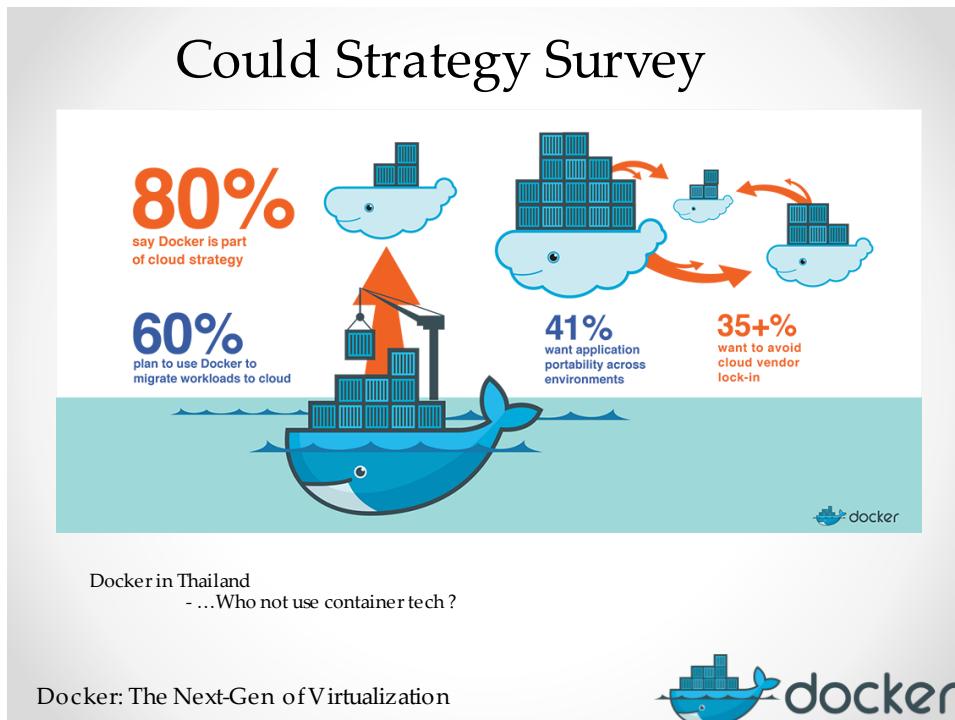
Container adoption and workload increasing

dockercon19

docker

Docker: The Next-Gen of Virtualization

52



53

Who use docker now ?

VISA

Supporting Global Growth in e-Commerce with Microservices

100M e-commerce transactions processed on Black Friday

10x increase in scalability with containers

Docker: The Next-Gen of Virtualization

 docker

54

Who use docker now ?

Docker Innovation Awards



Docker: The Next-Gen of Virtualization



55

Who use docker now ?

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



Docker: The Next-Gen of Virtualization



56

Who use docker now ?



Docker: The Next-Gen of Virtualization



57

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

Docker: The Next-Gen of Virtualization



58

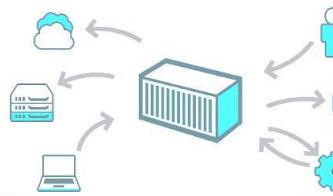
Who use docker now ?



59

Container Life Cycle

- Developer
 - สร้าง template สำหรับโปรแกรม (build)
 - เริ่มรัน template สำหรับแพทชันโปรแกรม (run)
 - พัฒนาเสร็จเรียบร้อย แล้ว save version เพื่อเตรียมนำเข้าใช้งาน (commit)
- Operation
 - เริ่มรัน template ที่ได้รับจาก developer (run) บนเครื่อง production
 - หยุดการให้บริการโปรแกรม (stop, kill)
 - ลบโปรแกรมออกจากเครื่อง production (rm, rmi)

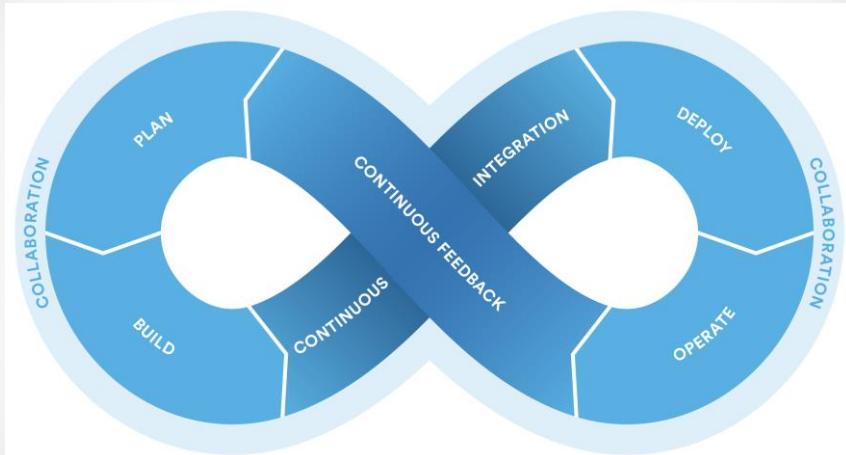


Docker: The Next-Gen of Virtualization



60

Container Life Cycle



Docker: The Next-Gen of Virtualization



61

What Cool of Docker ?

- รวมรวมทุกสิ่งที่จำเป็นต้องใช้ในการรันโปรแกรมไว้ได้ container (component, library etc)
- ขนาดไฟล์ container มีขนาดเล็กมาก (เทียบกับขนาดไฟล์ของ virtual machine หรือ os)
- น้อย overhead ในการรันโปรแกรมทั้งการรันตัว container และทัดสอบโปรแกรม
- ลดระยะเวลาในการติดตั้งและทดสอบโปรแกรม
- ส่งมอบโปรแกรมไปทำงานบนเครื่องแม่ข่าย production ได้โดยไม่มีความจำเป็นต้องปรับแต่งระบบใหม่ (zero configuration)
- สามารถรันโปรแกรมได้บนเครื่องแม่ข่ายทุกรอบแบบปฏิบัติการฯ ที่ติดตั้ง docker ได้
- สามารถ scale-out ได้ง่ายในอนาคต
- World open for docker !!!

Docker: The Next-Gen of Virtualization



62

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

Major components of the container:

- Roof
- Floor
- Walls
- Doors
- Locking Bars
- Header Posts
- Corner Casting
- Side Posts
- Top Rail
- Bottom Rail
- Cross Members

- 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

Docker: The Next-Gen of Virtualization

63

Benefit of docker for DevOps

- Build-Ship-Run

The diagram illustrates the Docker Build-Ship-Run cycle across four stages: Source, Dev, QA, and CI/CD. The cycle involves the following steps:

- Source:** A Source Code Repository containing a Dockerfile.
- Dev:** A Box2Docker Mac/Win Dev Machine.
- QA:** A Linux OS QA Machine.
- CI/CD:** A CI/CD pipeline with TEST, CI/CD, and TEST phases.

The cycle consists of three main phases:

- Build:** Converts source code into Docker images.
- Ship:** Publishes Docker images to the Docker Hub.
- Run:** Deploys Docker images to various environments.

The Docker Hub is shown as a central hub for Docker images, connected to the cycle via the Docker Hub API and Third Party Tools. The Run phase connects to three main deployment environments:

- Physical:** Infrastructure Management layer with Linux OS and Prod Machines.
- Virtual:** Infrastructure Management layer with VMs.
- Cloud:** Infrastructure Management layer with GCE, RAX, IBM, and other cloud providers.

Present by: Praparn L. (eva10409@gmail.com)

64

Docker Version

```
ubuntu@ip-10-0-1-234:~$ docker version
Client: Docker Engine - Community
 Version:           19.03.6
 API version:      1.40
 Go version:       go1.12.16
 Git commit:       369c74a3c
 Built:            Thu Feb 13 01:27:49 2020
 OS/Arch:          linux/amd64
 Experimental:    false

Server: Docker Engine - Community
 Engine:
  Version:          19.03.6
  API version:     1.40 (minimum version 1.12)
  Go version:      go1.12.16
  Git commit:      369c74a3c
  Built:           Thu Feb 13 01:26:21 2020
  OS/Arch:         linux/amd64
  Experimental:   false
 containerd:
  Version:          1.2.10
  GitCommit:        b34a5c8af56e510852c35414db4c1f4fa6172339
 runc:
  Version:          1.0.0-rc8+dev
  GitCommit:        3e425f8a0c931f88ed94a8c831b9d5aa481657
 docker:
  Version:          0.18.0
  GitCommit:        fec3d83
ubuntu@ip-10-0-1-234:~$
```

```
ubuntu@ip-10-0-1-234:~$ docker info
Client:
 Debug Mode: false

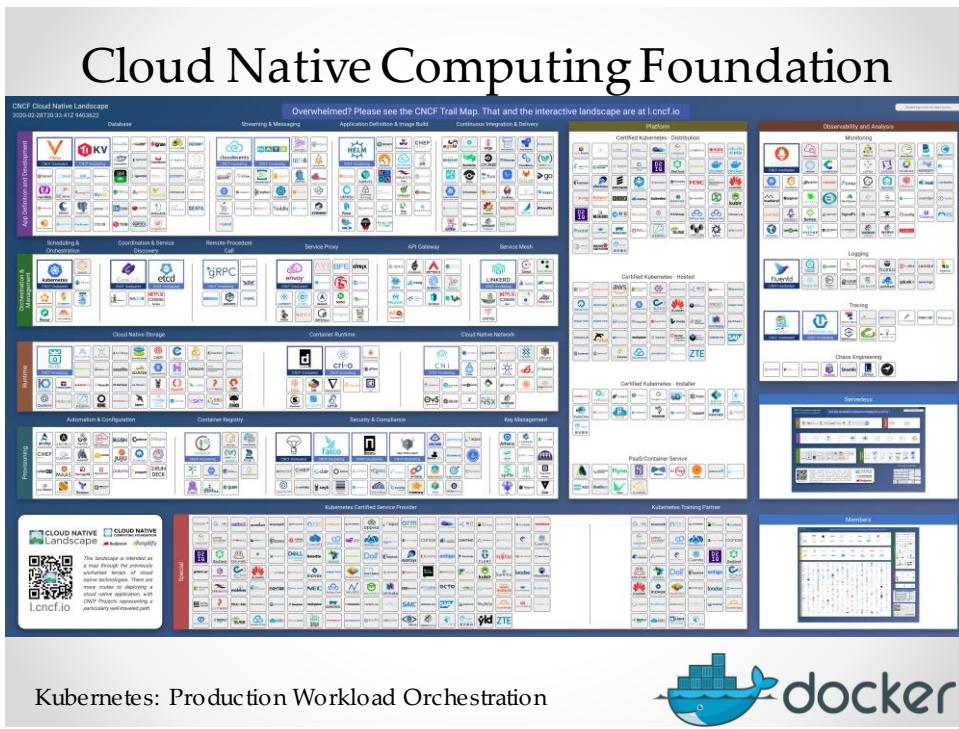
Server:
 Containers: 1
  Running: 0
  Paused: 0
  Stopped: 1
 Environment: 1
 Server Version: 19.03.6
 Storage Driver: overlay2
 Backing Filesystem: extfs
 Supports d_type: true
 Native Overlay Diff: true
 Logging Driver: json-file
 Cgroup Driver: systemd
 Plugins:
  Volume: local
 Network: bridge host ipvlan macvlan null overlay
 Log: awslogs fluentd gcplogs gelf journald json-file local logentries splunk syslog
 Security Options:
  apparmor
  seccomp
   Profile: default
 Kernel Version: 4.15.0-1044-aws
 Operating System: Ubuntu 18.04.2 LTS
 OSType: linux
 Architecture: x86_64
 CPUs: 2
 Total Memory: 1.916GB
 Name: ip-10-0-1-234
 ID: 3WKL:QLSQ:RQ20:ATC:J9MN:SELY:4ITY:UFTU:PAMD:YLBW:6514:CRZR
 Docker Root Dir: /var/lib/docker
 Debug Mode: false
 Registry: https://index.docker.io/v1/
 Labels:
 Experimental: false
 Insecure Registries:
 127.0.0.0/8
 Live Restore Enabled: false
WARNING: No swap limit support
ubuntu@ip-10-0-1-234:~$
```

Docker: The Next-Gen of Virtualization



65

Cloud Native Computing Foundation



Kubernetes: Production Workload Orchestration



66

What's New

The Moby Project website features a dark header with navigation links like 'Secure', 'Medium', 'vagrant', 'Mesos', 'Vue', 'docker', 'NGINX', 'Talos', 'Kubernetes', 'PAIX_Progressive...', 'Other Projects', and social media icons for Twitter, GitHub, and LinkedIn. Below the header is the Moby Project logo (a white whale icon) and the text 'Moby Project' with the subtitle 'An open framework to assemble specialized container systems without reinventing the wheel.' To the right of the logo is a diagram illustrating the assembly of container systems from various components.

Docker: The Next-Gen of Virtualization

The Docker logo consists of a blue whale icon followed by the word 'docker' in a lowercase sans-serif font.

67

What's News

19.03.6 (2020-02-12)

Builder

- builder-next: Allow modern sign hashes for ssh forwarding. [docker/engine#453](#)
- builder-next: Clear onbuild rules after triggering. [docker/engine#453](#)
- builder-next: Fix issue with directory permissions when usernamespace is enabled. [moby/moby#40440](#)
- Bump hcsshim to fix docker build failing on Windows 1903. [docker/engine#429](#)

Networking

- Shorten controller ID in exec-root to not hit UNIX_PATH_MAX. [docker/engine#424](#)
- Fix panic in drivers/overlay/encryption.go. [docker/engine#424](#)
- Fix hwaddr set race between us and udev. [docker/engine#439](#)

Runtime

- Bump Golang 1.12.16. [moby/moby#40433](#)
- Update containerd binary to v1.2.12. [moby/moby#40433](#)
- Update to runc v1.0.0-rc10. [moby/moby#40433](#)
- Fix possible runtime panic in Lgetxattr. [docker/engine#454](#)
- rootless: fix proxying UDP packets. [docker/engine#434](#)

Docker: The Next-Gen of Virtualization

The Docker logo consists of a blue whale icon followed by the word 'docker' in a lowercase sans-serif font.

68

What's News

Windows Server Core
By Microsoft
The official Windows Server Core base image for containers.

100M+
Container | 460.64 | Base Image

Description | Reviews | Resources

Important: You might encounter issues when using Windows Server containers with the February 11, 2020 security update release

You might encounter issues using the Windows Server Container Image for the February 11, 2020 security update when used on a Windows Server container host that is not updated to the same security update. For further information and guidance, visit the Microsoft Support Article for this issue.

Important: Client Firewall Rules Update to Microsoft Container Registry (MCR)

To provide a consistent FQDN, on March 3, 2020 the data endpoint will be changing from *.`cnn.mcr.microsoft.com` to *.`data.mcr.microsoft.com`.

For more info, see MCR Client Firewall Rules.

Featured Tags

- `lts2019` (LTS) `docker pull mcr.microsoft.com/windows/servercore:lts2019`
- `1809` (4K) `docker pull mcr.microsoft.com/windows/servercore:1809`

Featured Repos

- `windows/servercore`: Insider version of this base OS image.

Full Tag Listing

Tags	Architecture	Dockerfile	OsVersion	CreatedTime	LastUpdatedTime
1803	multiarch	No Dockerfile	10.0.18362.608	02/19/2020 02:39:30	02/19/2020 02:39:32
1903-kb4546853	multiarch	No Dockerfile	10.0.18362.608	02/19/2020 02:39:30	02/19/2020 02:39:30
10.0.18362.658	multiarch	No Dockerfile	10.0.18362.608	02/19/2020 02:39:31	02/19/2020 02:39:31
1903-emr64	amd64	No Dockerfile	10.0.18362.608	05/27/2019 17:59:13	02/19/2020 02:34:36
1903-kb4546853-amd64	amd64	No Dockerfile	10.0.18362.608	02/19/2020 02:34:04	02/19/2020 02:34:04
10.0.18362.658-amd64	amd64	No Dockerfile	10.0.18362.608	02/19/2020 02:34:05	02/19/2020 02:34:07
1809	multiarch	No Dockerfile	10.0.18363.608	11/10/2019 18:43:11	02/19/2020 02:39:58
1909-kb4546853	multiarch	No Dockerfile	10.0.18363.608	02/19/2020 02:39:59	02/19/2020 02:39:59
10.0.18363.658	multiarch	No Dockerfile	10.0.18363.608	02/19/2020 02:40:00	02/19/2020 02:40:00
1809-emr64	amd64	No Dockerfile	10.0.18363.608	11/13/2019 18:09:57	02/19/2020 02:34:17
1909-kb4546853-amd64	amd64	No Dockerfile	10.0.18363.608	02/19/2020 02:33:15	02/19/2020 02:33:15
10.0.18363.658-amd64	amd64	No Dockerfile	10.0.18363.608	02/19/2020 02:35:02	02/19/2020 02:35:02
lts2019	multiarch	No Dockerfile	10.0.17763.1040	09/09/2018 20:00:05	02/19/2020 02:35:08
1809	multiarch	No Dockerfile	10.0.17763.1040	09/09/2018 20:02:05	02/19/2020 02:35:02
1809-kb4546852	multiarch	No Dockerfile	10.0.17763.1040	02/19/2020 02:31:00	02/19/2020 02:35:08
10.0.17763.1040	multiarch	No Dockerfile	10.0.17763.1040	02/19/2020 02:31:00	02/19/2020 02:35:01
1909-17763-1040-amd64	amd64	No Dockerfile	10.0.17763.1040	03/12/2019 22:07:45	02/19/2020 02:36:09
1809-emr64	amd64	No Dockerfile	10.0.17763.1040	03/12/2019 22:04:03	02/19/2020 02:34:35
1809-kb4546852-amd64	amd64	No Dockerfile	10.0.17763.1040	03/19/2020 02:33:54	02/19/2020 02:33:54
10.0.17763.1040-amd64	amd64	No Dockerfile	10.0.17763.1040	02/19/2020 02:35:10	02/19/2020 02:35:10
1803	multiarch	No Dockerfile	10.0.17731.341140	10/02/2018 05:11:43	02/19/2020 02:35:44
1803-kb4546851	multiarch	No Dockerfile	10.0.17731.341140	02/19/2020 02:35:47	02/19/2020 02:35:47
10.0.17731.341140	multiarch	No Dockerfile	10.0.17731.341140	02/19/2020 02:35:43	02/19/2020 02:35:43
1803-emr64	amd64	No Dockerfile	10.0.17731.341140	04/09/2019 18:03:45	02/19/2020 02:34:25
1803-kb4546853-amd64	amd64	No Dockerfile	10.0.17731.341140	02/19/2020 02:33:43	02/19/2020 02:33:43
10.0.17731.341140-amd64	amd64	No Dockerfile	10.0.17731.341140	02/19/2020 02:33:04	02/19/2020 02:35:04
lts2016	multiarch	No Dockerfile	10.0.14993.3096	10/05/2018 03:34:11	02/19/2020 02:36:46
1607	multiarch	No Dockerfile	10.0.14993.3096	03/12/2019 22:09:17	02/19/2020 02:37:18
1607-kb4546850	multiarch	No Dockerfile	10.0.14993.3096	02/19/2020 02:06:45	02/19/2020 02:36:45
10.0.14993.3096	multiarch	No Dockerfile	10.0.14993.3096	02/19/2020 02:06:47	02/19/2020 02:36:47
lts2016-amd64	amd64	No Dockerfile	10.0.14993.3096	04/09/2019 21:21:07	02/19/2020 02:40:38
1607-emr64	amd64	No Dockerfile	10.0.14993.3096	04/09/2019 21:20:00	02/19/2020 02:39:53
1607-kb4546850-amd64	amd64	No Dockerfile	10.0.14993.3096	02/19/2020 02:38:09	02/19/2020 02:38:09
10.0.14993.3096-amd64	amd64	No Dockerfile	10.0.14993.3096	02/19/2020 02:41:12	02/19/2020 02:41:12

REF:<https://support.microsoft.com/en-us/help/4542617/you-might-encounter-issues-when-using-windows-server-containers-with-t>

Docker: The Next-Gen of Virtualization



69

What's News

You might encounter issues when using Windows Server containers with the February 11, 2020 security update release

Applies to: Windows Server 2016, Windows Server 2019, all versions, Windows Server version 1803, [More](#)

Last updated February 18, 2020 6:30pm PST

Symptoms

You might encounter issues using Windows Server containers, unless both the Windows container host and Windows Server containers are matched with the February 11, 2020 security update. Compatibility was affected by a security change that must be addressed in both the container host and the container image, requiring them to match.

Symptoms when running or building a container might include:

- When you run the command "docker run" or "docker build" you might not receive output and it might become non-responsive.
- Your Windows Server Container in Kubernetes does not reach the "running" state.
- You receive the error, "docker: Error response from daemon: container <id> encountered an error during Start: failure in a Windows system call: The wait operation timed out. (0x102)."
- Your 32-bit application or processes running inside the container might silently fail.

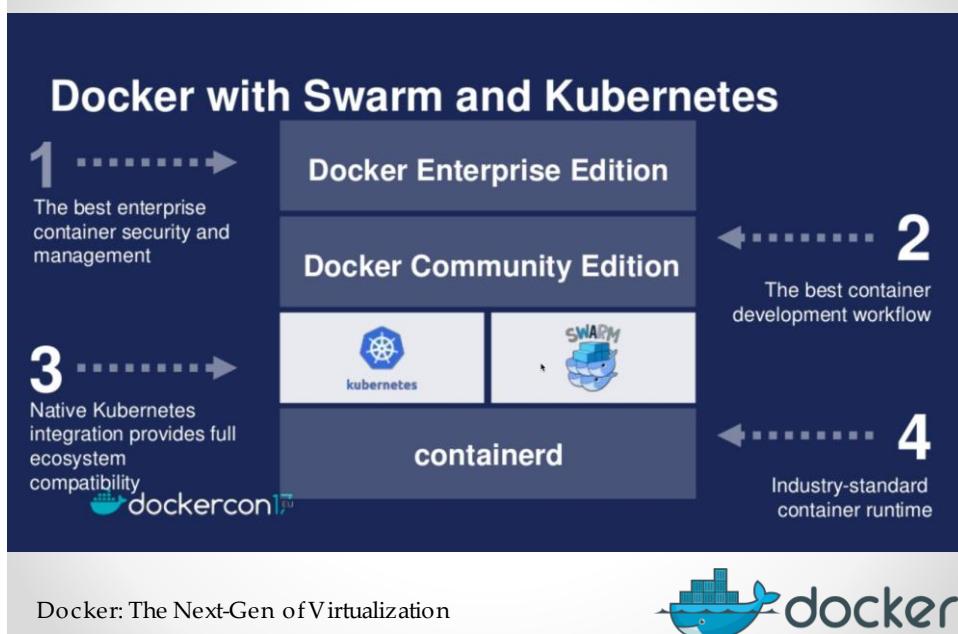
REF:<https://support.microsoft.com/en-us/help/4542617/you-might-encounter-issues-when-using-windows-server-containers-with-t>

Docker: The Next-Gen of Virtualization



70

What's News



71



72

Platform of Docker

- Docker Toolbox (Obsolete) 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)

Docker: The Next-Gen of Virtualization



73

Linux vs Windows vs MAC OS

[boot2docker.io](#)

State of boot2docker

The `boot2docker` CLI tool is *long*-since officially deprecated in favor of [Docker Machine](#).
On the other hand, the `boot2docker` distribution (as in, `boot2docker.iso`) is in "maintenance mode". See <https://github.com/boot2docker/boot2docker> for more details.
It is **highly** recommended that users transition from Boot2Docker over to [Docker for Mac](#) or [Docker for Windows](#) instead.

[boot2docker docs](#)

Get started with Docker Machine and a local VM

Prerequisite Information

Docker Enterprise Edition

Docker Cloud

Docker Compose

Docker for Mac

Docker for Windows

Docker ID accounts

Docker Machine

Machine overview

Install Machine

Docker: The Next-Gen of Virtualization



74

Docker-machine CLI

- ตรวจสอบสถานะ docker-machine

```
docker-machine ls
```

- สั่งเริ่มการทำงานของ docker-machine

```
docker-machine start <machine name>
```

- สั่งหยุดการทำงานของ docker-machine

```
docker-machine stop <machine name>
```

- สร้าง docker-machine เครื่องใหม่

```
docker-machine create --driver virtualbox/hyperv <name>
```

Docker: The Next-Gen of Virtualization



75

Workshop 1-2: Pull Image

ทำการ download image ubuntu ลงมาที่เครื่อง boot2docker โดยใช้คำสั่งดังนี้

```
docker image pull labdocker/alpine:latest
```

```
docker image pull labdocker/alpineweb:latest
```

```
docker image pull labdocker/cadvisor:latest
```

Compare with Ubuntu image

```
docker images/docker image ls
```

```
[ubuntu@ip-10-0-1-104:~$ docker image ls
REPOSITORY          TAG      IMAGE ID      CREATED       SIZE
hello-world         latest   fce289e99eb9  2 months ago  1.84kB
labdocker/alpineweb latest   5770233f642f  4 months ago  50.5MB
labdocker/ubuntu    latest   ea4c82ddcd15a  4 months ago  85.8MB
labdocker/alpine    latest   196d12cf6ab1  5 months ago  4.41MB
ubuntu@ip-10-0-1-104:~$ ]
```

Docker: The Next-Gen of Virtualization



76

Workshop 1-2: Pull Image

ทำการตรวจสอบประวัติของ image ที่ใช้งานผ่านคำสั่ง docker image history

```
docker image history labdocker/alpine:latest
```

```
docker image history labdocker/alpineweb:latest
```

```
[ubuntu@ip-10-0-1-104:~]$ docker image history labdocker/alpine:latest
IMAGE          CREATED     CREATED BY                                      SIZE      COMMENT
19d612cf6ab1  5 months ago  /bin/sh -c #(nop)  CMD ["/bin/sh"]           0B
<missing>      5 months ago  /bin/sh -c #(nop) ADD file:25c10b1d1b41d4ea1...  4.41MB
[ubuntu@ip-10-0-1-104:~]$ docker image history labdocker/alpineweb:latest
IMAGE          CREATED     CREATED BY                                      SIZE      COMMENT
5770233f642f  4 months ago  /bin/sh -c #(nop) EXPOSE 3000              0B
<missing>      4 months ago  /bin/sh -c #(nop) ENTRYPOINT ["node" "hello..."]  0B
<missing>      4 months ago  /bin/sh -c #(nop) WORKDIR /nodejs            0B
<missing>      4 months ago  /bin/sh -c #(nop) COPY file:b974a437806b927a...  263B
<missing>      4 months ago  /bin/sh -c apk update && apk add nodejs ...    46.1MB
<missing>      4 months ago  /bin/sh -c #(nop) ENV NODE_VERSION=v8.11.4 ...  0B
<missing>      4 months ago  /bin/sh -c #(nop) LABEL Description=NodeJS...  0B
<missing>      4 months ago  /bin/sh -c #(nop) MAINTAINER Praparn Lueang...  0B
<missing>      5 months ago  /bin/sh -c #(nop) CMD ["/bin/sh"]           0B
<missing>      5 months ago  /bin/sh -c #(nop) ADD file:25c10b1d1b41d4ea1...  4.41MB
[ubuntu@ip-10-0-1-104:~]$
```

Docker: The Next-Gen of Virtualization



77

General Command in Docker

Command	Description	Command	Description
docker attach	Attach local standard input, output, and error streams to a running container	docker kill	Kill one or more running containers
docker build	Build an image from a Dockerfile	docker load	Load an image from a tar archive or STDIN
docker checkpoint	Manage checkpoints	docker login	Log in to a Docker registry
docker commit	Create a new image from a container's changes	docker logout	Log out from a Docker registry
docker config	Manage Docker configs	docker logs	Fetch the logs of a container
docker container	Manage containers	docker network	Manage networks
docker cp	Copy files/folders between a container and the local filesystem	docker node	Manage Swarm nodes
docker create	Create a new container	docker pause	Pause all processes within one or more containers
docker deploy	Deploy a new stack or update an existing stack	docker plugin	Manage plugins
docker diff	Inspect changes to files or directories on a container's filesystem	docker port	List port mappings or a specific mapping for the container
docker events	Get real time events from the server	docker ps	List containers
docker exec	Run a command in a running container	docker pull	Pull an image or a repository from a registry
docker export	Export a container's filesystem as a tar archive	docker push	Push an image or a repository to a registry
docker history	Show the history of an image	docker rename	Rename a container
docker image	Manage images	docker restart	Restart one or more containers
docker images	List images	docker rm	Remove one or more containers
docker import	Import the contents from a tarball to create a filesystem image	docker rmi	Remove one or more images
docker info	Display system-wide information	docker run	Run a command in a new container
docker inspect	Return low-level information on Docker objects	docker save	Save one or more images to a tar archive (streamed to STDOUT by default)
		docker search	Search the Docker Hub for images
		docker secret	Manage Docker secrets

<https://docs.docker.com/engine/reference/commandline/docker/>

Docker: The Next-Gen of Virtualization



78

General Command in Docker

<code>docker secret</code>	Manage Docker secrets
<code>docker service</code>	Manage services
<code>docker stack</code>	Manage Docker stacks
<code>docker start</code>	Start one or more stopped containers
<code>docker stats</code>	Display a live stream of container(s) resource usage statistics
<code>docker stop</code>	Stop one or more running containers
<code>docker swarm</code>	Manage Swarm
<code>docker system</code>	Manage Docker
<code>docker tag</code>	Create a tag TARGET_IMAGE that refers to SOURCE_IMAGE
<code>docker top</code>	Display the running processes of a container
<code>docker trust</code>	Manage trust on Docker images (experimental)
<code>docker unpause</code>	Unpause all processes within one or more containers
<code>docker update</code>	Update configuration of one or more containers
<code>docker version</code>	Show the Docker version information
<code>docker volume</code>	Manage volumes
<code>docker wait</code>	Block until one or more containers stop, then print their exit codes

<https://docs.docker.com/engine/reference/commandline/docker/>

Docker: The Next-Gen of Virtualization



79

Image, Repository and Container

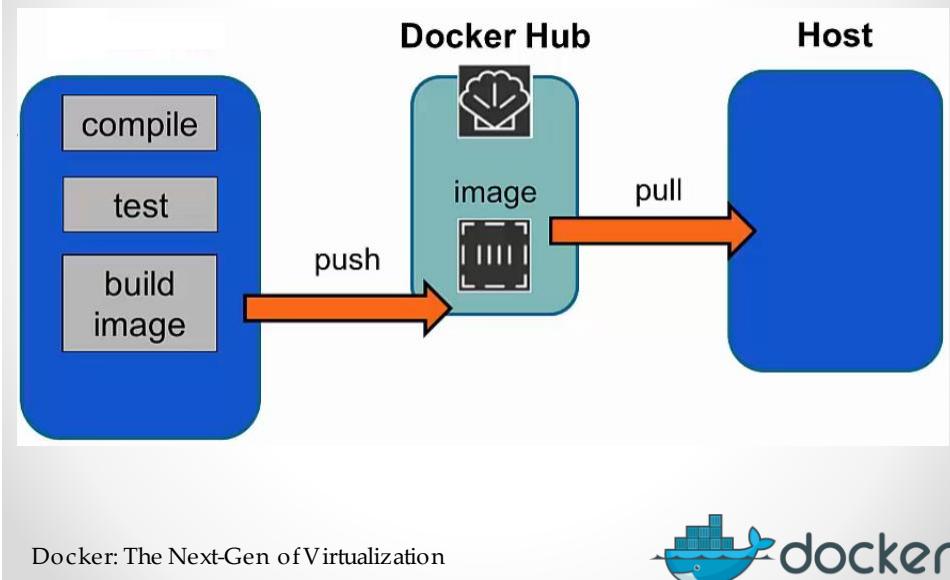
• • •

Docker: The Next-Gen of Virtualization



80

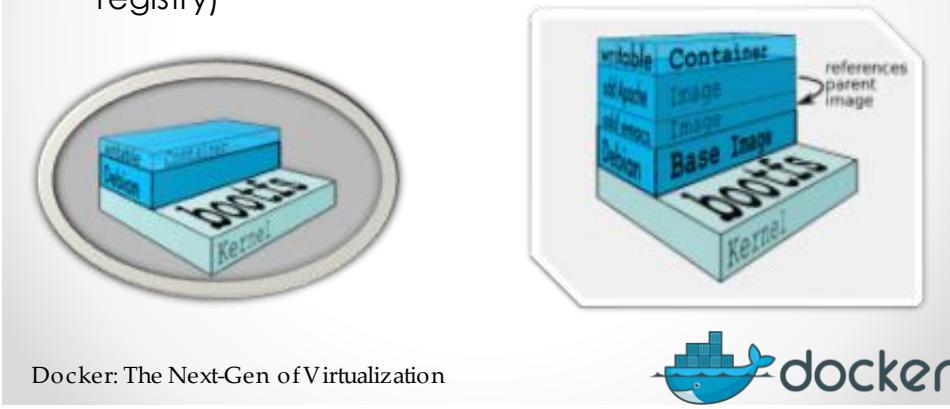
Docker Deployment



81

Docker Image

- Image คือ template ที่ถูกสร้างขึ้นเพื่อเตรียมใช้ในการรัน container
- เป็นไฟล์ที่อ่านได้อย่างเดียว
- ถูกสร้างโดยผู้ใช้งานเอง หรือผู้อื่น
- จัดเก็บไว้ใน repository (hub.docker.com, registry, trust registry)



82

Docker Image

- เรียกดู image ที่อยู่ภายในเครื่อง

```
docker images /docker image ls
```

```
[ubuntu@ip-10-0-1-104:~$ docker image ls
REPOSITORY      TAG      IMAGE ID      CREATED      SIZE
hello-world     latest   fce289e99eb9  2 months ago  1.84kB
labdocker/alpineweb  latest   5770233f642f  4 months ago  50.5MB
labdocker/ubuntu  latest   ea4c82dcd15a  4 months ago  85.8MB
labdocker/alpine    latest   196d12cf6ab1  5 months ago  4.41MB
ubuntu@ip-10-0-1-104:~$ ]
```

- tag image ใหม่เพื่อให้ตรงตามความต้องการในการใช้งาน

```
docker image tag <image id> <acc name/imagename: version>
```

```
Ex: docker image tag 14f89d0e6257 labdocker/alpinelab:1.0
```

Docker: The Next-Gen of Virtualization



83

Docker Image

- ตรวจสอบรายละเอียดของ image

```
docker images history <image id/image name>
```

```
ubuntu@ip-10-0-1-58:~$ docker image history labdocker/alpine:latest
IMAGE      CREATED      CREATED BY      SIZE      COMMENT
196d12cf6ab1  7 months ago   /bin/sh -c #(nop) CMD ["/bin/sh"]
<missing>  7 months ago   /bin/sh -c #(nop) ADD file:25c10b1d1b41d46a1...
ubuntu@ip-10-0-1-58:~$ docker image history labdocker/alpineweb:latest
IMAGE      CREATED      CREATED BY      SIZE      COMMENT
5770233f642f  5 months ago   /bin/sh -c #(nop) EXPOSE 3000
<missing>  5 months ago   /bin/sh -c #(nop) ENTRYPOINT ["node" "hello..."]
<missing>  5 months ago   /bin/sh -c #(nop) WORKDIR /nodejs
<missing>  5 months ago   /bin/sh -c #(nop) COPY file:b974a437806b927a...
<missing>  5 months ago   /bin/sh -c #(nop) /nodejs
<missing>  5 months ago   /bin/sh -c apk update && apk add nodejs ...
<missing>  5 months ago   /bin/sh -c #(nop) ENV NODE_VERSION=v8.11.4 ...
<missing>  5 months ago   /bin/sh -c #(nop) LABEL Description=NodeJS...
<missing>  5 months ago   /bin/sh -c #(nop) MAINTAINER Praparn Lueang...
<missing>  7 months ago   /bin/sh -c #(nop) CMD ["/bin/sh"]
<missing>  7 months ago   /bin/sh -c #(nop) ADD file:25c10b1d1b41d46a1...
ubuntu@ip-10-0-1-58:~$
```

Docker: The Next-Gen of Virtualization



84

Repository (Registry)

- component ในการจัดเก็บ docker (image) ต่างๆรวมไปที่ศูนย์กลาง เพื่อให้ docker engine บนเครื่องต่างๆมาเรียก image ไปใช้งาน
- รองรับการเก็บ version ของ image อย่างเป็นระบบ
- สามารถให้ข้อมูลหรือคู่มือแนะนำการใช้งานกับผู้ download image ได้
- มี official image ที่สร้างจากศูนย์พัฒนาโปรแกรมเอง
- Docker มีบริการ repository บน cloud แก่ผู้ใช้งาน
- Url: <https://hub.docker.com/>
- Free registry without cost
- Private repository (registry, trust registry)

Docker: The Next-Gen of Virtualization



85

Repository (Registry)

- Hub.docker.com

Docker: The Next-Gen of Virtualization



86

Repository (Registry)

google/cadvisor ☆
By people • Updated 12 hours ago
Analyzes resource usage and performance characteristics of running containers.
Container

Docker Pull Command
docker pull google/cadvisor

Owner
Google

cAdvisor

Overview Tags

cAdvisor provides container users an understanding of the resource usage and performance characteristics of their running containers. It is a running daemon that collects, aggregates, processes, and exports information about running containers. Specifically, for each container it keeps resource isolation parameters, historical resource usage, and histograms of complete historical resource usage. This data is exported by container and machine-wide.

<https://github.com/google/cadvisor>

Official *cAdvisor* releases are built on Linux and exported over a scratch image, this guarantees a small image size.

Dockerfile: <https://github.com/google/cadvisor/blob/master/deploy/Dockerfile>

Each *cAdvisor* version is tagged. We also have 2 tags for the beta and stable track:

- **latest**: Latest stable build, this is the latest officially supported releases.
- **canary**: Images built from HEAD periodically. Potentially unstable!

We also have an Automated Build canary release of *cAdvisor* which is continuously built from HEAD. This can be found in the *google/cadvisor-canary* image. It is not recommended for production use due to its size and volatility.

Docker: The Next-Gen of Virtualization

docker

87

Repository (Registry)

- download image ออกจาก registry


```
docker image pull <account name>/image name: tags>
```

Ex: docker image pull labdocker/alpine:latest
- Upload image ขึ้นไปเก็บบน registry
 - >Login


```
docker login <url> -u <username> -p <password> -e <email>
```

Ex: docker login -u labdocker -p xxxxxxxx -e xxxx@xxx.xxx

Ex: docker login 10.38.7.248:8080 -u labdocker

Docker: The Next-Gen of Virtualization

docker

88

Repository (Registry)

- Push image ขึ้นไปเก็บบน registry

```
docker image push<account name/image name: tags>
```

Ex: docker image push labdocker/alpinelab:1.0

- Logoff ออกจาก Repository

```
docker logoff
```

*Docker 1.10 (upper) จะทำการ push แบบ parallel layer ทำให้สามารถ push image ได้เร็วขึ้น สาม เท่าจากเดิม

Docker: The Next-Gen of Virtualization



89

Workshop 1-3: Create & Push Image

- ใน workshop นี้สอนการสร้าง image file สำหรับใช้เป็น web server และจัดเก็บ image file ลงใน repository ส่วนตัวเพื่อเตรียมไว้ใช้งาน
- สมัครใช้งาน <http://hub.docker.com> และแจ้งยืนยันอีเมล์เพื่อเริ่มใช้งาน
- ทำการสร้าง repository ชื่อ alpineweb ตามด้วยอย่างด้านล่าง

The screenshot shows the Docker Hub 'Create Repository' page. The repository name 'alpineweb' is entered in the 'Repository Name' field. The 'Visibility' dropdown is set to 'Public'. Below the form, there are sections for 'Build Settings (optional)' and 'Please re-link a GitHub or Bitbucket account'. At the bottom, there are 'Cancel', 'Create', and 'Create & Build' buttons.

Docker: The Next-Gen of Virtualization



90

Workshop 1-3: Build & Push Image

- ตรวจสอบ image id ด้วยคำสั่ง


```
docker image ls
```

- tag image ใหม่เป็น <accountname>/alpineweb:latest


```
docker image tag labdocker/alpineweb:latest \
<accountname>/alpineweb:latest
```

- Login เข้า hub.docker.com ด้วย username/password ที่ตั้งไว้


```
docker login -u <xxxx>
```

- Push image alpine ที่ tag ให้เข้า repository


```
docker image push <accountname>/alpineweb:latest
```

Docker: The Next-Gen of Virtualization



91

Other service in hub.docker.com

The screenshot shows the Docker Hub homepage with a search bar and filters for Docker Certified images. It lists several popular services:

- Oracle Database Enterprise Edition** by Oracle (updated 1 month ago)
- Oracle Database 12c Enterprise Edition** by Oracle (Docker Certified, Linux, Oracle, Database)
- Oracle Java 8 SE Server JRE** by Oracle (updated 1 month ago)
- Oracle Java 8 SE Server JRE** by Oracle (Docker Certified, Linux, Oracle, Java, Programming Languages)
- MySQL Server Enterprise Edition** by Oracle (Docker Certified, Linux, Oracle, MySQL, Database)
- MySQL Server Enterprise Edition** by Oracle (The world's most popular open source database system, Docker Certified, Linux, Oracle, MySQL, Database)
- Oracle WebLogic Server** by Oracle (updated 1 month ago)
- Oracle WebLogic Server** by Oracle (Docker Certified, Linux, Oracle, Application Frameworks, Application Infrastructure)

At the bottom, there is a sidebar for the **cocktail** image, which is a Couchbase Server.

Docker: The Next-Gen of Virtualization



92

Other service in hub.docker.com

Docker Certified: Trusted & Supported Products

- Certified containers are the top apps available as containers.
- Certified Plugins for memory and volumes in containers.
- Certified Infrastructure refers to optimized and validated Docker platforms for enterprise-OS and Cloud Providers.

[View Certified Images](#)

Oracle Database Enterprise Edition
By Oracle
Oracle Database 12c Enterprise Edition

Developer Tier
Install Oracle Database Server

Description **Reviews** **Resources**

Oracle Database Server 12c R2 is an industry-leading relational database server. The Oracle Database Server Docker image contains Oracle Database Server 12.2.0.1 Enterprise Edition running on Oracle Linux 7. This image contains a default database in a multi-rent configuration with a single pluggable database.

Filters (1) Docker Certified

Categories: Authorization, Logging, Metrics, Storage, Volume

Docker Certified: Docker Certified

Plugins: 1 - 25 of 25 available plugins.

Same Logic Logging Plugin By Same Logic Inc. • Updated 1 week ago

A Docker logging driver plugin to send logs to Same Logic.

Weave Net By Weave Inc. • Updated 1 month ago

Simple, resilient multi-host Docker networking and more.

Hedging Docker Volume Plugin By HEDEVON • Updated 5 months ago

Software-Defined Storage for Containers

vSphere Storage for Docker By vSphere • Updated 1 year ago

vSphere Storage for Docker enables you to run stateful containerized applications on top of VMware vSphere.

Docker: The Next-Gen of Virtualization

93

Other service in hub.docker.com

PUBLISHER PROGRAM

Deliver your business through Docker Hub

Package and publish apps and plugins as containers in Docker Hub for easy download and deployment by millions of Docker users worldwide.

[Apply To Publish](#) [Learn More](#)

Sysdig Monitor (Commercial)
By Sysdig, Inc.

Pre-Cloud \$39 / mo Base Cloud \$25 / mo

Pre-Cloud is per-host, per-month, with up to 30 containers and 500 custom metrics per host. Content is for Sysdig Cloud only.

[Press to Checklist](#)

DESCRIPTION **REVIEWS** **RESOURCES**

What are your containers doing? Sysdig is the most powerful docker monitoring and troubleshooting solution. Built-in Container Health Monitoring, Container Profiling, and Container Performance Monitoring let you monitor and integrate with the entire docker ecosystem, giving you deep visibility into your applications and microservices. Free trial at www.sysdig.com.

GitLab Enterprise Edition
By GitLab, Inc.

Bring Your Own License Details

All EE features can be seen here: <https://about.gitlab.com/products/comparison-options>

SETUP INSTRUCTIONS
<https://docs.gitlab.com/omnibus/docker/README.html>

Copy and paste to pull this image
`docker pull gitlab/gitlab-ce`

Docker: The Next-Gen of Virtualization

94

Status.docker.com

Docker System Status

Current system status information.

All Systems Operational Updated a minute ago

Our system status page is a real-time view of the performance and uptime of Docker products and services.

Docker Package Repositories OK Operational

Docker Support Site OK Operational

Docker Community Forums OK Operational

Docker.com Web OK Operational

<https://status.docker.com/>

Metrics

DOCKER HUB API RESPONSE TIME 500.71MS

DOCKER HUB API UPTIME 100.00%

DOCKER REGISTRY HUB API RESPONSE TIME 504.03MS

DOCKER REGISTRY HUB API UPTIME 100.00%

Docker: The Next-Gen of Virtualization



95

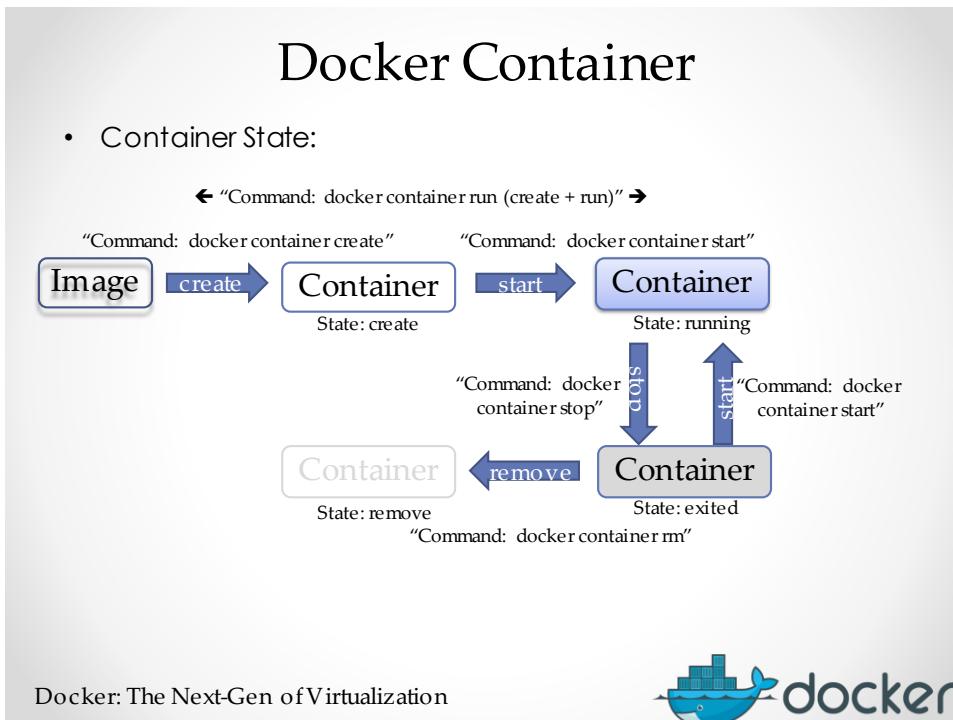
Docker Container

- Container คือชุดของ software layer ที่รันอิสระโดยยังอิงจาก image (run)
- ใช้สำหรับสร้างสภาพแวดล้อมที่จำเป็นต้องใช้ในการรันหรือพัฒนาโปรแกรม
- เมื่อพัฒนาโปรแกรมเสร็จเรียบร้อยแล้ว หรือต้องการ backup container สามารถล็อกเก็บ container ชุดปัจจุบันไปเป็น image เพื่อการเรียกใช้งานต่อไป (commit)

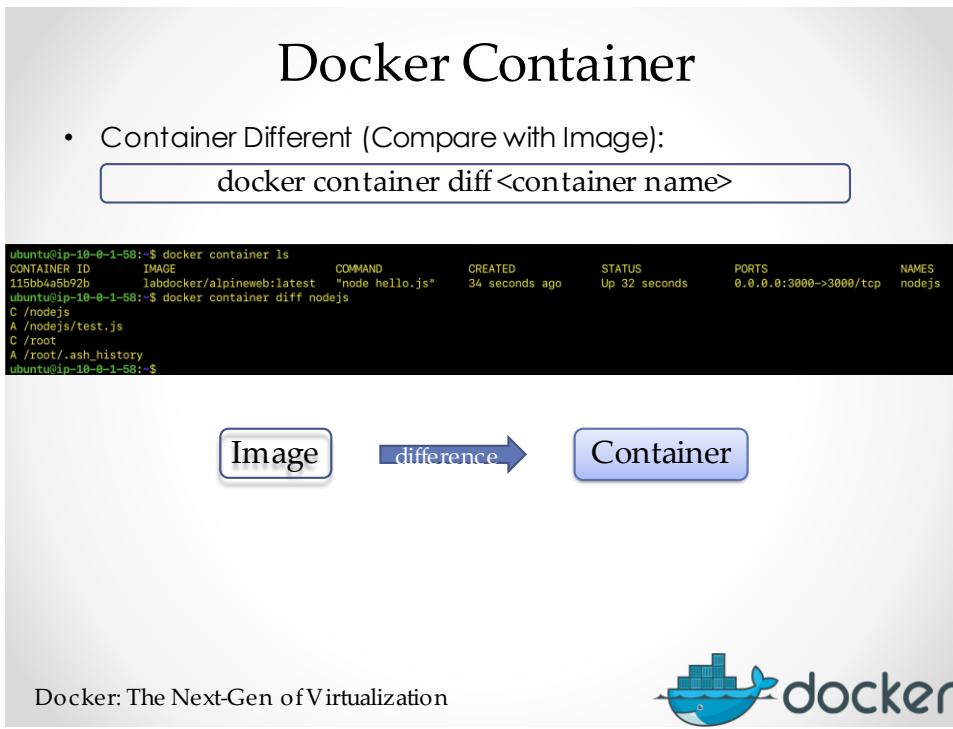
Docker: The Next-Gen of Virtualization



96



97



98

Docker Container

- Container Performance/Port:

```
docker container stats <container name>
```

```
docker container top <container name>
```

```
docker container port <container name>
```

```
CONTAINER ID        NAME          CPU %     MEM USAGE / LIMIT      MEM %     NET I/O             BLOCK I/O          PIDS
115bb4a5b92b      nodejs        0.00%    8.375MiB / 1.91GiB   0.43%    5.9kB / 2.3kB       36.9kB / 0B      6
^C
ubuntu@ip-10-0-1-58:~$ docker container top nodejs
UID           PID   PPID  STIME TTY          TIME          CMD
root         3069   3043  14:15 pts/0        00:00:00  node
hello.js
ubuntu@ip-10-0-1-58:~$ docker container port nodejs
3000/tcp -> 0.0.0.0:3000
ubuntu@ip-10-0-1-58:~$
```

Docker: The Next-Gen of Virtualization



99

Docker Container

- Container change name on-the-fly:

```
docker container rename <old name> <new name>
```

```
ubuntu@ip-10-0-1-58:~$ docker container ls
CONTAINER ID        IMAGE               COMMAND      CREATED     STATUS      PORTS     NAMES
115bb4a5b92b      labdocke/alpineweb:latest "node hello.js"  13 minutes ago   Up 13 minutes   0.0.0.0:3000->3000/tcp   nodejs
ubuntu@ip-10-0-1-58:~$ docker container rename nodejs nodejsnew
ubuntu@ip-10-0-1-58:~$ docker container ls
CONTAINER ID        IMAGE               COMMAND      CREATED     STATUS      PORTS     NAMES
115bb4a5b92b      labdocke/alpineweb:latest "node hello.js"  13 minutes ago   Up 13 minutes   0.0.0.0:3000->3000/tcp   nodejsnew
ubuntu@ip-10-0-1-58:~$
```

Docker: The Next-Gen of Virtualization



100

Docker Container

- สั่ง run docker เพื่อสร้าง container จาก image file

```
docker container run <option><image id/name><command>
```

Ex: docker container run -i -t --rm -p 3000:3000 \ labdocker/alpineweb:latest node hello.js

- สั่ง exec command ไปบน container ที่รันอยู่

```
docker container exec -it <container id/name> <command>
docker container attach <container id/name>
```

- เริ่ม/หยุด container

```
docker container <start/stop> <container id/name>
```

- ลบ container

```
docker container rm <containerid/name>
```

Docker: The Next-Gen of Virtualization



101

Workshop: Run Container

- Interactive NODEJS

- สร้าง container แบบ Interactive โดยตั้งชื่อว่า nodejs และ map network 3000:3000

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

- ทดสอบเปิดหน้าเว็บ URL: http://<ip address>:3000

- ตรวจสอบ container ด้วยคำสั่ง

```
docker container ps -a
```

- Exist with Ctrl+C

Docker: The Next-Gen of Virtualization



102

Workshop: Run Container

- DAEMON LINE BOT

ธนารักษ์แห่งประเทศไทย

ชื่อของ API : REST
รูปแบบข้อมูล : JSON
การอนุญาติ : PUBLIC

ลิงก์ของ API : https://apis.bot.or.th/Stat/Stat-ReferenceRate/DAILY_REF_RATE_V1/

รหัสผ่าน API :
api_key = xLkH6u6u-uUvUfLchchDz2hgh44mgZM6D4h

ข้อมูลการตั้งค่าที่ต้องการใช้งาน API

ชื่อ	ประเภท	รายละเอียด
start_period	Datetime	วันที่เริ่มต้น (YYYY-MM-DD) ตั้งแต่ 2010-06-30
end_period	Datetime	วันที่สิ้นสุด (YYYY-MM-DD) ตั้งแต่ 2017-06-30
<hr/>		
ชื่อรายการที่ต้องการดู	ชื่อ	รายละเอียด
report_name_eng	String	รายงาน ค่าแลกเปลี่ยน
report_name_th	String	รายงาน ค่าเงิน
report_unit_name_eng	String	หน่วย ค่าแลกเปลี่ยน
report_unit_name_th	String	หน่วย ค่าเงิน
report_source_id_date	String	แหล่งมาเดียว
report_name_eng	String	รายงาน ค่าแลกเปลี่ยน
report_name_th	String	รายงาน ค่าเงิน
last_updated	String	วันที่ปรับเปลี่ยน
period	String	วันที่ตรวจสอบ
ref	String	ผลลัพธ์

BOT API Services Process

```

graph LR
    subgraph "BOT API Services Process"
        direction LR
        A[BOT API Services] -- "Result (JSON/XML Format)" --> B[API Services]
        B -- "Call APIs" --> C[Application]
        C -- "Develop" --> D[Developers]
        C -- "Use" --> E[Users]
        D -- "API URL, Reference Rate List, Result List" --> F[Developer Portal]
        E -- "API URL, Reference Rate List, Result List" --> F
    end
    F[Developer Portal  
https://apis.bot.or.th/developer]
    
```

Docker: The Next-Gen of Virtualization

103

Workshop: Run Container

- Ex 2: DAEMON LINE BOT

ธนารักษ์แห่งประเทศไทย

Exchange Rates 2.0.1

APIs

Weighted-average Interbank Exchange Rate - THB / USD

Average Exchange Rate - THB / Foreign Currency

Plans

	Default Plan
Weighted-average Interbank Exchange Rate - ...	200 per hour
Average Exchange Rate - THB / Foreign Curre...	200 per hour
<hr/>	
Free	
<hr/>	
Subscribe	

* = Mouseover for more information

Docker: The Next-Gen of Virtualization

104

Workshop: Run Container

- DAEMON LINE BOT

```
docker@labdocke:~$ docker container run --rm --name linebot -e "TITLE=Line BOT (Bank of Thailand)" -e "TOKEN=EIeqyillzkzp
aCoIRrebZTcGbIyzDZhOjcd0t6CX" 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-aliv
e\": \"timeout=3\", \"x-rateLimit-Limit\": \"1000\", \"x-rateLimit-Remaining\": \"998\", \"x-rateLimit-ImageRate
maining\": \"0\", \"x-rateLimit-Reset\": \"1516421245\"}, \"request\": {\"uri\": \"https://notify-api.line.me/v1/messages\", \"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\", \"method\": \"POST\", \"headers\": {\"Content-Type\": \"application/x-www-form-urlencoded\", \"Authorization\": \"Bearer EIeqyillzkzp
aCoIRrebZTcGbIyzDZhOjcd0t6CX\", \"Content-Length\": 466}}}
{"status":200, "message": "Ok"}
```

Docker: The Next-Gen of Virtualization

105

Workshop: Run Container

- DAEMON LINE BOT
- สั่งรัน container แบบ Daemon โดยตั้งชื่อว่า linebot เพื่อทำการดึงข้อมูลจาก API ของธนาคารแห่งประเทศไทยอອນໄມແລ້ວສັງ LINE Notify ໄປຫາ Token Key ທີ່ກຳທັນ

```
docker container run -it --rm --name linebot \
-e TITLE="Line BOT (Bank of Thailand)" \
-e "TOKEN=<LINE TOKEN>" \
labdocker/linenotify:bot_v2
```

Docker: The Next-Gen of Virtualization

106

Workshop: Run Container

- Detach Mode NODEJS
- สั่งรัน container แบบ detach (daemon) โดยตั้งชื่อว่า nodejs และ map network 3000:3000

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

- ทดสอบเปิดหน้าเว็บบน url: http://<ip address>:3000
- ตรวจสอบ container ด้วยคำสั่ง


```
docker container ps -a
```
- ทดสอบ Shell เข้าสู่ container ด้วยคำสั่ง


```
docker container exec -i -t nodejs sh
```

Docker: The Next-Gen of Virtualization



107

Workshop: Run Container

- Detach Mode NODEJS
- สั่งหยุดเริ่ม container ด้วยคำสั่ง stop


```
docker container stop nodejs
```

```
docker container start nodejs
```
- ทำการตรวจสอบ container offline ด้วยคำสั่ง


```
docker container ls -a
```
- ทดสอบ Shell เข้าสู่ container ด้วยคำสั่ง


```
docker container exec -i -t nodejs sh
```

```
docker container attach nodejs
```

Docker: The Next-Gen of Virtualization



108

Workshop: Run Container

- Detach Mode NODEJS
- ทำการตรวจสอบ process ภายใน container

docker container top nodejs

docker container stats nodejs

- ทำการตรวจสอบความแตกต่างระหว่าง Container กับ image ทันที

docker container diff nodejs

- ตรวจสอบการ map port container

docker container port nodejs

- เปลี่ยนแปลงชื่อ container และ online

docker container rename nodejs nodejsnew

Docker: The Next-Gen of Virtualization



109

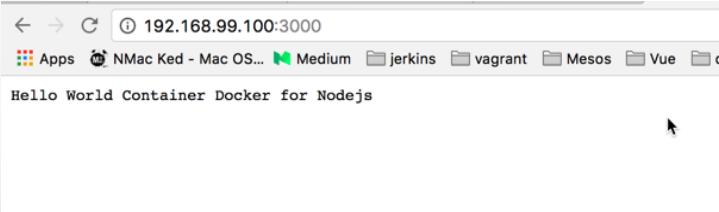
Workshop: Run Container

- Detach Mode NODEJS

```

1  var http = require('http');
2
3  http.createServer(function (req, res) {
4      res.writeHead(200, {'Content-Type': 'text/plain'});
5      res.end('Hello World Container Docker for Nodejs\n');
6  }).listen(3000, '0.0.0.0');
7
8  console.log('Server running at http://0.0.0.0:3000/');

```



Docker: The Next-Gen of Virtualization



110

Workshop: Run Container

- Detach Mode Python
- สร้าง Container และ detach (daemon) โดยตั้งชื่อว่า python และ map network 5000:5000


```
docker create run -t --name python -p 5000:5000 \
lab.docker/cluster/webservicelite
```
- ทำการรัน


```
docker container ls -a
docker container start python
```
- ทดสอบเบิดหน้าเว็บบน URL: http://<ip address>:5000
- ทดสอบ Shell เข้าสู่ container ด้วยคำสั่ง


```
docker container exec -i -t python sh
```

```
docker container attach python
```

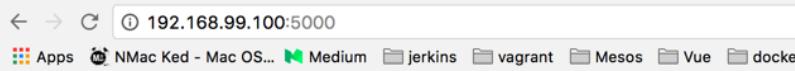
Docker: The Next-Gen of Virtualization 

111

Workshop: Run Container

- Detach Mode PYTHON

```
1 from flask import Flask
2 import os
3 import time
4 app = Flask(__name__)
5
6 @app.route('/')
7 def hello():
8     return '<H1> Welcome Page from Container Python Lab </H1>Checkpoint Date/Time: ' + time.strftime("%c") +'\n'
9
10 if __name__ == "__main__":
11     app.run(host="0.0.0.0", port=5000, debug=True)
```



Welcome Page from Container Python Lab

Checkpoint Date/Time: Thu Aug 3 15:25:25 2017

Docker: The Next-Gen of Virtualization 

112

CPU, Memory & I/O

• • •

Docker: The Next-Gen of Virtualization



113

CPU

- Default container จะมองเห็น CPU resource ทั้งหมดในเครื่องระหว่าง runtime และ share เวลาการทำงานให้ทุก Container เท่ากัน

- config share time ในการใช้งาน (default: 1024)

```
--cpu-shares, -c =“<0 (default): Ratio>”
```

- config share core cpu

```
--cpuset-cpus="1, 3"
```

- กำหนด cpu quota & period (default: 100 ms)

```
--cpu-period=100000 --cpu-quota=500000
```

- NUMA (non-uniform memory access) architecture

```
--cpuset-mems="1,3"
```

Docker: The Next-Gen of Virtualization



114

Workshop: CPU Configure

- Download cAdvisor

```
docker image pull labdocker/cadvisor:latest
```

- สร้าง cAdvisor ตาม command ดังนี้

```
docker container run \
    --mount type=bind,source=/var/run,target=/var/run \
    --mount type=bind,source=/sys,target=/sys,readonly \
    --mount
    type=bind,source=/var/lib/docker,target=/var/lib/docker,readonly \
        --publish=8080:8080 \
        --detach=true \
        --name=cadvisor \
    labdocker/cadvisor:latest
```

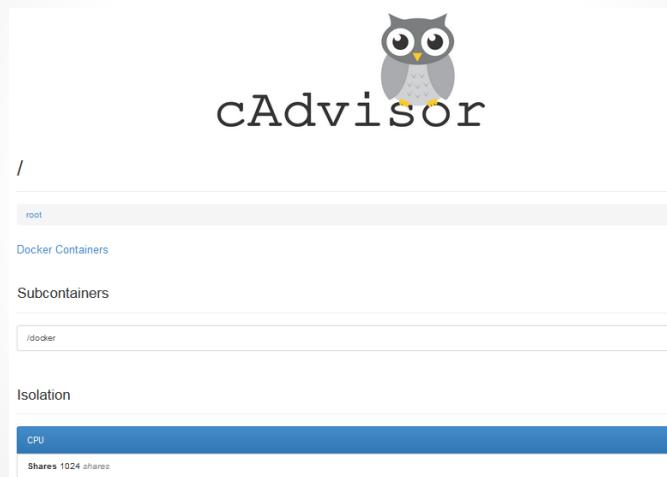
Docker: The Next-Gen of Virtualization



115

Workshop: CPU Configure

- cAdvisor



Docker: The Next-Gen of Virtualization



116

Workshop: CPU Configure

- Download busybox เพื่อใช้ในการทดสอบ


```
docker image pull labdocker/busybox:latest
```

- Scenario 1 (Single CPU: 70/30)

```
docker container run -d \
--name='Share_30' \
--cpuset-cpus=0 \
--cpu-shares=30 \
labdocker/busybox:latest md5sum /dev/urandom
```

```
docker container run -d \
--name='Share_70' \
--cpuset-cpus=0 \
--cpu-shares=70 \
labdocker/busybox:latest md5sum /dev/urandom
```

Docker: The Next-Gen of Virtualization



117

Workshop: CPU Configure

- Result



Docker: The Next-Gen of Virtualization



118

Quiz ?

- Scenario (Multiple CPU: 70/30)

```
docker container run -d \
--name='Share_30' \
--cpuset-cpus=0 \
--cpu-shares=30 \
labdocker/busybox:latest md5sum /dev/urandom
```

```
docker container run -d \
--name='Share_70' \
--cpuset-cpus=1 \
--cpu-shares=70 \
labdocker/busybox:latest md5sum /dev/urandom
```

Docker: The Next-Gen of Virtualization



119

Quiz ?

- Result



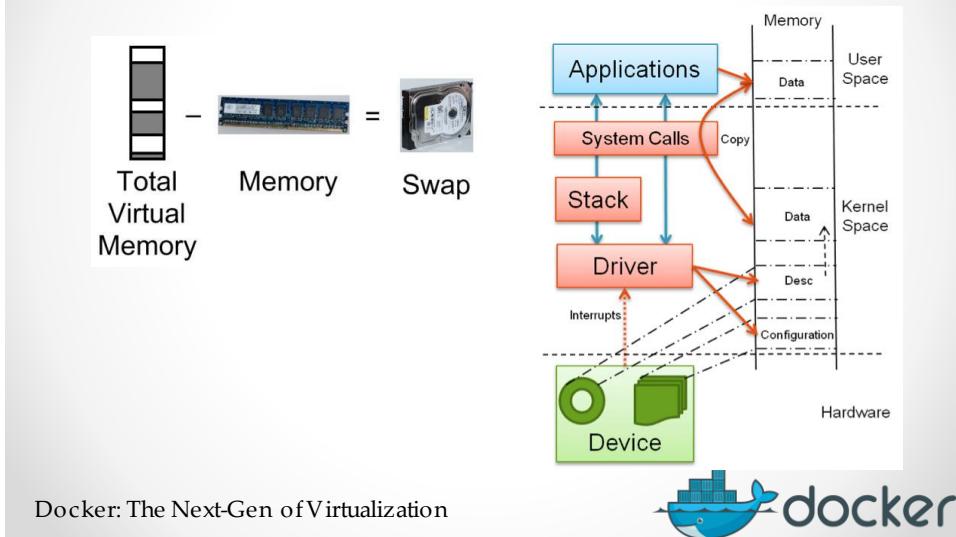
Docker: The Next-Gen of Virtualization



120

Memory

- Default container จะมองเห็น memory resource ทั้งหมดในเครื่องระหว่าง runtime และสามารถใช้งาน memory resource ทั้งหมดเท่าที่ต้องการ



121

Memory

- การคอนฟิก memory บน container สามารถกำหนดได้ดังนี้
- memory limit and reservation (default: unlimit) (B:byte, K:kilobyte, M:megabyte, G:gigabyte)

```
--memory, -m =10G --memory-reservation= 1G
```

- memory swap (default: unlimit)

```
-- memory-swap= -1
```

- kernel memory

```
-- kernel-memory= 500M
```

- memory swappiness (kernel)(0=no swap)

```
-- memory-swappiness =0
```

Docker: The Next-Gen of Virtualization



122

Memory

- memory ใน production system
- กำหนด memory-swappiness = 0 (no swappiness for kernel memory)
- กำหนด memory swap (--memory-swap) เป็น 2 เท่าของ memory (Limit)
- monitor footprint ในการใช้งาน memory ของ application system และกำหนด เป็น “--memory-reservation”
- ทำการ capacity testing เพื่อกำหนด maximum memory ที่ต้องการในการใช้งาน (+30%) และกำหนด “--memory” (Limit)
- สำหรับ critical system ให้กำหนด “--memory-reservation” เท่ากับ “--memory” (Limit)
- Memory Priority ? (Still waiting)

Docker: The Next-Gen of Virtualization



123

I/O

- โดย default ทุก container จะมีโภคสิ่ง I/O เท่าๆ กัน (500 weight)
- กำหนดค่า weight ในการใช้งาน I/O (สำหรับ Direct IO เท่านั้น)
`--blkio-weight 300, --blkio-weight-device "/s01/tdb:500"`
- จำกัดการอ่านข้อมูล เป็น bps (kb: kilobyte, mb: megabyte, gb: gigabyte) / iops
`--device-read-bps /s01/tdb:1mb`
`--device-read-iops /s01/tdb:20000`
- จำกัดการเขียนข้อมูล เป็น bps , iops
`--device-write-bps /s01/tdb:1mb`
`--device-write-iops /s01/tdb:20000`

Docker: The Next-Gen of Virtualization



124

I/O

- I/O configure in production system
- weight I/O ควรกำหนดเฉพาะในกรณีที่ application ต้องทำงานกับ direct i/o (slow than cache)
- Monitor การใช้งาน I/O ของ container (IOSTAT -xtc) และปรับแต่ง

```
$ iostat -xtc
          extended device statistics
      device r/s w/s kr/s kw/s wait actv svc_t %w   tty      cpu
      fd0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0 0 0 0 0 0 0 0 0 0 0 100
      sd0 0.0 0.0 0.4 0.4 0.0 0.0 0.0 49.5 0 0 0 0 0 0 0 0 0 0 0 0
      sd6 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0 0 0 0 0 0 0 0 0 0 0 0
      nfs1 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0 0 0 0 0 0 0 0 0 0 0 0
      nfs49 0.0 0.0 0.0 0.0 0.0 0.0 0.0 15.1 0 0 0 0 0 0 0 0 0 0 0 0
      nfs53 0.0 0.0 0.4 0.0 0.0 0.0 0.0 24.5 0 0 0 0 0 0 0 0 0 0 0 0
      nfs54 0.0 0.0 0.0 0.0 0.0 0.0 6.3 0 0 0 0 0 0 0 0 0 0 0 0 0 0
      nfs55 0.0 0.0 0.0 0.0 0.0 0.0 4.9 0 0 0 0 0 0 0 0 0 0 0 0 0 0
```

- Web server / Application server → IOPS สูง / Transfer ต่ำ
- Database server → IOPS ต่ำ / Transfer สูง
- คำนึงถึง physical storage performance

Docker: The Next-Gen of Virtualization



125

I/O

- Normal disk iops

Disk Speed	IOPS
15,000	175
10,000	125
7200	75
5400	50

RAID	Write Penalty
0	1
1	2
5	4
6	6
DP	2
10	2

- Raw disk performance: disk iops x unit of disk
 - Ex: SAS 128GB (15K) x 6 units => $175 \times 6 = 1060$ iops
- Raid disk performance:
 - $((\text{raw iops} \times \% \text{write}) / \text{penalty}) + (\text{raw} \times \% \text{read})$
 - Normal situation (write 30%, read 70%)
 - Ex: Raid 5 $((1060 \times .3)/4) + (1060 \times .7) \rightarrow 821.6$ iops
 - Ex: Raid 1 $((1060 \times .3)/2) + (1060 \times .7) \rightarrow 901$ iops

Docker: The Next-Gen of Virtualization



126

Network

• • •

Docker: The Next-Gen of Virtualization



127

Network

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

```
[docker@labdocke:~$ docker network ls
NETWORK ID      NAME      DRIVER      SCOPE
0dc1bbf00498    bridge    bridge      local
91d6dd4056a9    host      host       local
fa786a00adda    none     null       local
docker@labdocke:~$ ]
```

- Bridge/User Defined** 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 หรือใช้งานกับการเชื่อมต่อแบบอื่นๆ
- host** คือ network host ที่ container ใช้งาน host network stack ในการทำงาน (ใช้ในกรณี ต้องการ network performance สูงสุด) (security concern)

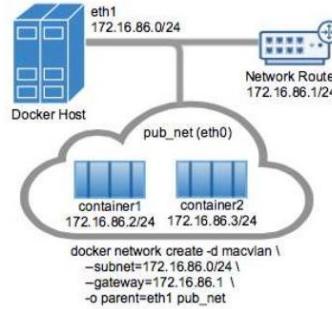
Docker: The Next-Gen of Virtualization



128

Network

- Additional network type
- **MACVLAN**



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

Docker: The Next-Gen of Virtualization



129

Network

- Summary
 - **Bridge/User Defined Network:** best when you need multiple containers to communicate on the same Docker host.
 - **Host networks:** best when the network stack should not be isolated from the Docker host, but you want other aspects of the container to be isolated.
 - **Overlay networks:** best when you need containers running on different Docker hosts to communicate, or when multiple applications work together using swarm services.
 - **Macvlan networks:** best when you are migrating from a VM setup or need your containers to look like physical hosts on your network, each with a unique MAC address
 - **Third-party network:** allow you to integrate Docker with specialized network stacks.

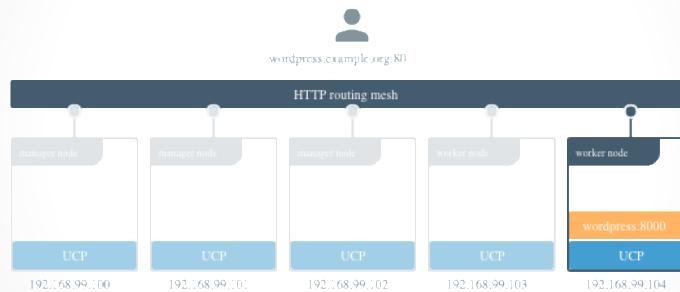
Docker: The Next-Gen of Virtualization



130

Network

- Docker EE Network Feature:
 - **HTTP Route Mesh:** load balance L4 to target container on specific docker host



- **Session Stickiness:** (Cookies Base) for re-route traffic from original to same end-point container (Stateful)

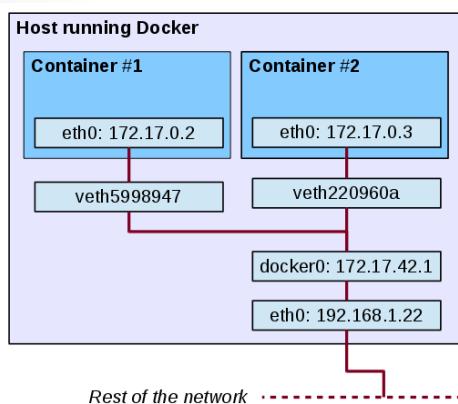
Docker: The Next-Gen of Virtualization



131

Network

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



Docker: The Next-Gen of Virtualization



132

Network

- docker network inspect bridge

Docker: The Next-Gen of Virtualization



133

Network

- Option เกี่ยวกับ network สำหรับสั่ง run container

```
--dns= x.x.x.x (Default --name,--net-alias,--link also  
dns internal docker)  
--net=<brige/none/host/custom>  
--net-alias = "xxxx"  
--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 container run -i -t --rm --name nodejs -p 3000:3000 \  
labdocker/alpineweb:latest node nodejs/hello.js
```

Docker: The Next-Gen of Virtualization



134

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

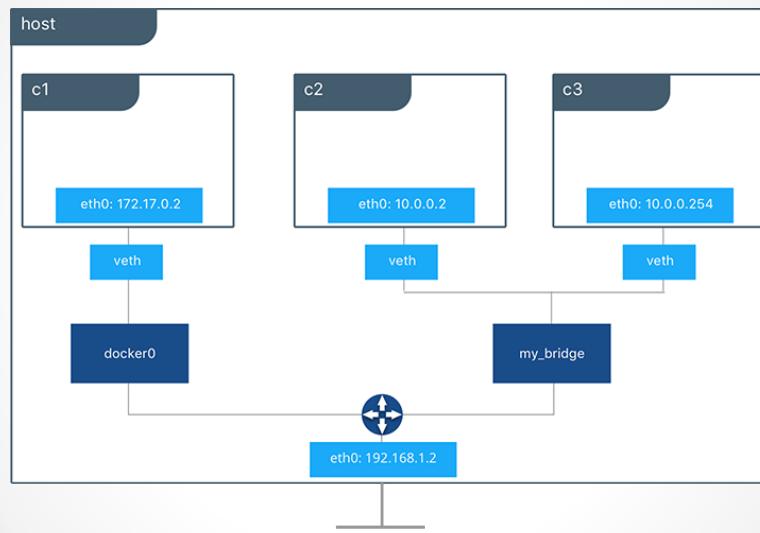
- สามารถเพิ่มเติม option ในการสร้าง virtual switch เพิ่มเติม
 - d macvlan -o parent = กำหนด vlan tagging
 - subnet = xx (ระบุ ip address ของ virtual switch (Ex: 192.168.100.0/24))
 - ip-range = xx (ระบุ ip address ที่จะส่งให้ container ทำงาน) (Ex: 192.168.100.128/25)
 - gateway = xx (ระบุ ip address 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: The Next-Gen of Virtualization



135

Network



Docker: The Next-Gen of Virtualization



136

Network

- ການເພີ່ມ network ໃນ container

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

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

- ການອອກ network ໃນ container

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

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

Docker: The Next-Gen of Virtualization



137

Network

- Link container (add on /etc/hosts) (Legacy)

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

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

Legacy container links

Estimated reading time: 14 minutes

⌚ Warning: The `--link` flag is a legacy feature of Docker. It may eventually be removed. Unless you absolutely need to continue using it, we recommend that you use user-defined networks to facilitate communication between two containers instead of using `--link`. One feature that user-defined networks do not support that you can do with `--link` is sharing environmental variables between containers. However, you can use other mechanisms such as volumes to share environment variables between containers in a more controlled way.

- DNS resolve on docker network (Recommend for Production)

Docker: The Next-Gen of Virtualization



138

Workshop: Network Configure

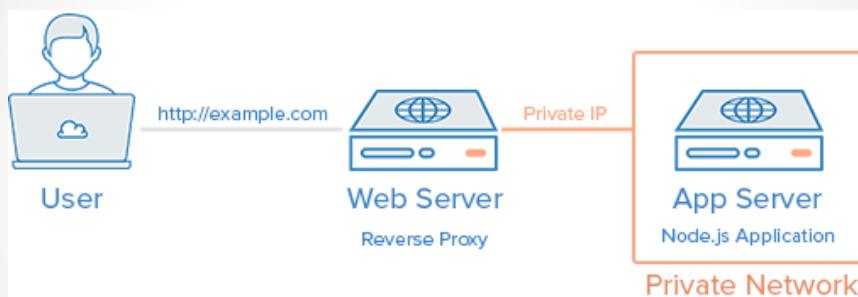
- Part 1: Reverse Proxy Network (DNS)
- Purpose: ทำการ Deploy nodejs webserver ด้วย solution proxy ของ nginx (load balance / reverse proxy)
- **public network** เพื่อให้บุคคลภายนอกเข้าใช้งานผ่านnginx server (reverse proxy)
 - IP Address: 192.168.100.0/24
 - Range Address: 192.168.100.128 – 192.168.100.256
 - MTU: 1500
- **private network** เพื่อไฟ 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: The Next-Gen of Virtualization



139

Workshop: Network Configure

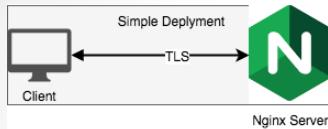


Docker: The Next-Gen of Virtualization

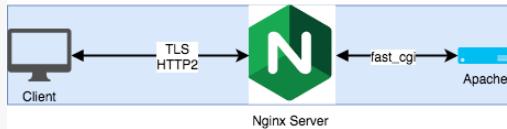


140

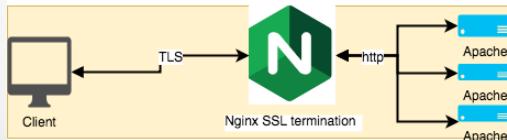
Workshop: Network Configure



Nginx Server A
Plain Deployment



Nginx Reverse Proxy



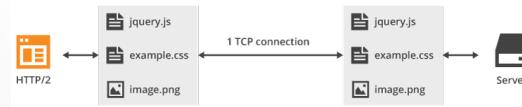
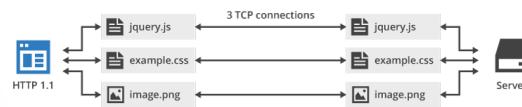
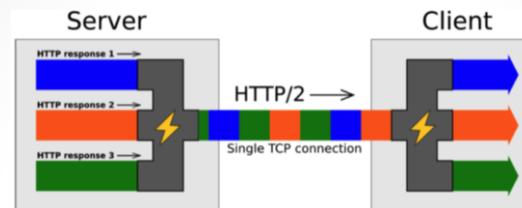
Nginx SSL Offloading

Docker: The Next-Gen of Virtualization



141

Workshop: Network Configure

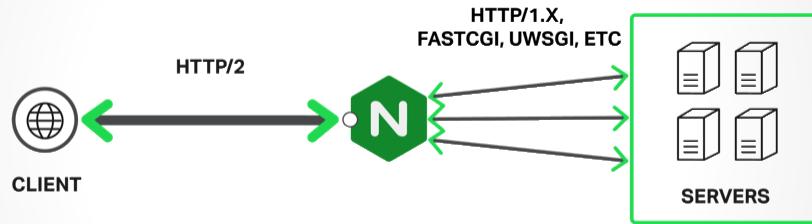


Docker: The Next-Gen of Virtualization



142

Workshop: Network Configure

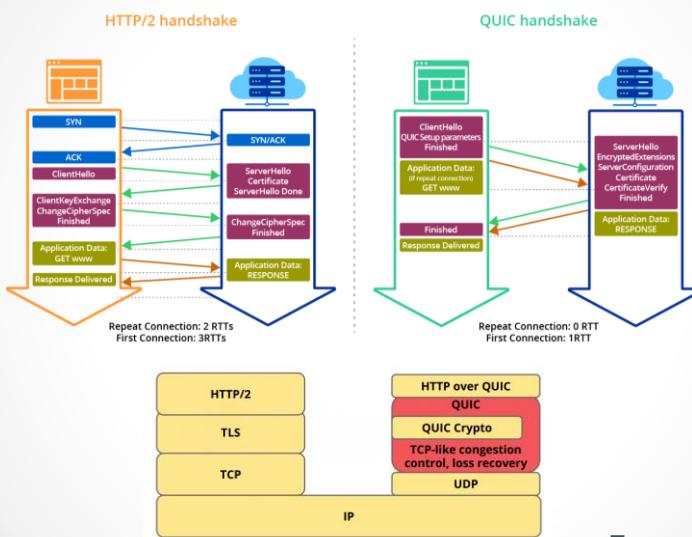


Docker: The Next-Gen of Virtualization



143

Workshop: Network Configure



Docker: The Next-Gen of Virtualization



144

Workshop: Network Configure

[FREE TRIAL](#) [CONTACT US](#)

What's Coming in NGINX 1.17

The first release on the NGINX 1.17 mainline is already here. NGINX 1.17.0 includes support for variables in bandwidth-limiting configurations with the `limit_rate` directive and also allows the `include` directive to be used in all configuration contexts, even inside an `if` block.

Development has also started on support for QUIC and HTTP/3 – the next significant update to the transport protocols that will deliver websites, applications, and APIs. This is a significant undertaking, but likely to arrive during the NGINX 1.17 development cycle. To get the latest insights on the NGINX roadmap and, our other open source projects and products, join us at NGINX Conf, our annual user conference, this September 9–12 in Seattle.



[MICROSERVICES
From Design to Deployment](#)

[DOWNLOAD NOW](#)

<https://www.nginx.com/blog/nginx-1-16-1-17-released/>

Docker: The Next-Gen of Virtualization



145

Workshop: Network Configure

- Normal HTTP/1.1

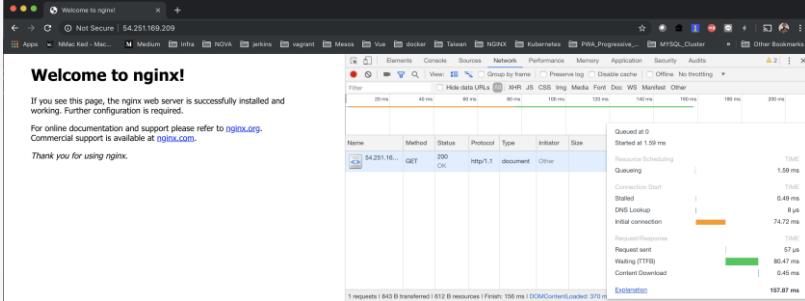
<https://www.nginx.com/>

Welcome to nginx!

If you see this page, the nginx web server is successfully installed and working. Further configuration is required.

For online documentation and support please refer to nginx.org. Commercial support is available at nginx.com.

Thank you for using nginx.



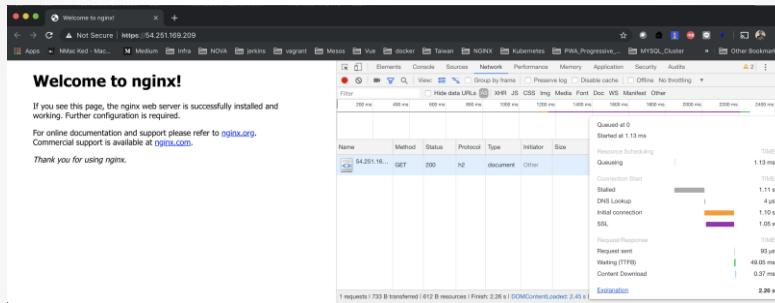
Docker: The Next-Gen of Virtualization



146

Workshop: Network Configure

- HTTP/2 HTTPS (Old word: SPDY)



Docker: The Next-Gen of Virtualization



147

Workshop: Network Configure

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, 443:8443)
Join Network: Webinternal
Join Network: Webpublic

vSwitch: Webpublic
IP Address: 192.168.100.0/24

Map Port
(80:8080
443:8443)

Public
Network

Docker: The Next-Gen of Virtualization



148

Workshop: Network Configure

- ทำกรสร้าง Switch สำหรับเชื่อมต่อ Private / Public Network
 - docker network create --driver bridge \
 --subnet=192.168.100.0/24 --ip-range=192.168.100.128/25 \
 --gateway=192.168.100.5 --opt="com.docker.network.mtu"="1500"
 webpublic
 - docker network create --driver bridge \
 --subnet=192.168.101.0/24 --ip-range=192.168.101.128/25 \
 --gateway=192.168.101.5 --opt="com.docker.network.mtu"="9000"
 webinternalCreate Server

Docker: The Next-Gen of Virtualization



149

Workshop: Network Configure

- สร้างเครื่อง Webserver (nodejs) และ Nginx
 - Web1**
 - docker run -dt --name web1 --net webinternal \
 --net-alias web1 labdocker/alpineweb:web1 node hello.js
 - Web2**
 - docker run -dt --name web2 --net webinternal \
 --net-alias web2 labdocker/alpineweb:web2 node hello.js
 - NGINX**
 - docker run -dt --name nginx --net webinternal \
 -p 80:8080 -p 443:8443 labdocker/nginx:labnetworkhttp2
- docker network connect webpublic nginx

Docker: The Next-Gen of Virtualization



150

Workshop: Network Configure

- url: http://<Public IP>/nodejs

Docker: The Next-Gen of Virtualization



151

Workshop: Network Configure

- curl result:

```
..ntu@ip-10-0-1-55: ~ — -bash ... | ...composemultinodejs — -bash
[ubuntu@ip-10-0-1-55:~/temp]$ docker container exec -it nginx sh
/usr/sbin # curl http://web1:3000
Hello World Container Docker for Nodejs Number 1
/usr/sbin # curl http://web1:3000 -I
HTTP/1.1 200 OK
Content-Type: text/plain
Date: Sat, 22 Jun 2019 02:50:39 GMT
Connection: keep-alive

/usr/sbin # curl http://web2:3000
Hello World Container Docker for Nodejs Number 2
/usr/sbin # curl http://web2:3000 -I
HTTP/1.1 200 OK
Content-Type: text/plain
Date: Sat, 22 Jun 2019 02:50:40 GMT
Connection: keep-alive

/usr/sbin #
```

Docker: The Next-Gen of Virtualization



152

Workshop: Network Configure

- nginx.conf

```
upstream nodejs_web {
    server web1:3000;
    server web2:3000;
}

server {
    listen 8080 default_server;
    location /nodejs{
        proxy_pass http://nodejs_web;
        proxy_http_version 1.1;
        proxy_set_header Upgrade $http_upgrade;
        proxy_set_header Connection 'upgrade';
        proxy_set_header Host $host;
        proxy_cache_bypass $http_upgrade;
    }
}

server {
    listen 8443 ssl http2;
    ssl_certificate      labdocke.com.crt;
    ssl_certificate_key  labdocke.com.key;
    location /nodejs{
        proxy_pass http://nodejs_web;
        proxy_http_version 1.1;
        proxy_set_header Upgrade $http_upgrade;
        proxy_set_header Connection 'upgrade';
        proxy_set_header Host $host;
        proxy_cache_bypass $http_upgrade;
    }
}
```

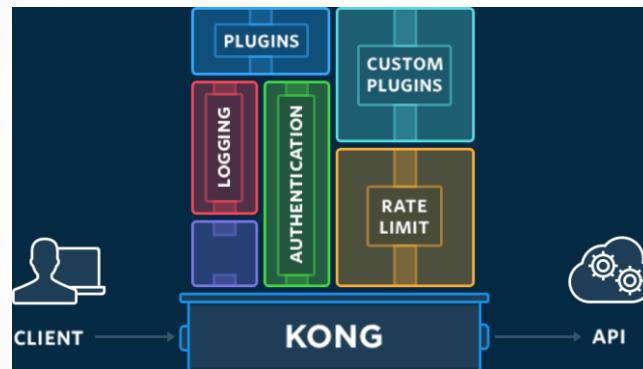
Docker: The Next-Gen of Virtualization



153

Workshop: Network Configure

- Bonus: Kong API Gateway the basic



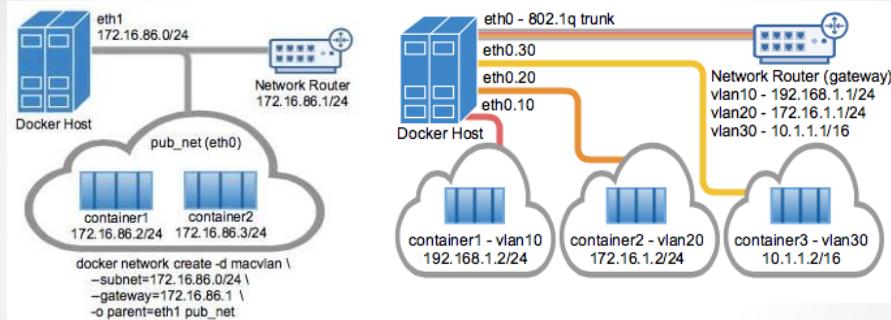
Docker: The Next-Gen of Virtualization



154

Network

- 802.1q Tagging (MACVLAN)



Docker: The Next-Gen of Virtualization



155

Demo: MACVLAN

- Reverse Proxy Network (MACVLAN) (This need docker-machine)
- Purpose: สำหรับ Deploy nodejs webserver ด้วย solution proxy ของ nginx ผ่าน Macvlan
- **Macvlan:**
 - Interface: eth1
 - IP Address: 192.168.99.0/24
 - Gateway: 192.168.99.1 → Your Machine
 - Labdocke: 192.168.99.100 → Default
 - Range Address: 192.168.99.192/28
 - 192.168.99.193 – 192.168.99.206

<https://github.com/docker/libnetwork/blob/master/docs/macvlan.md>

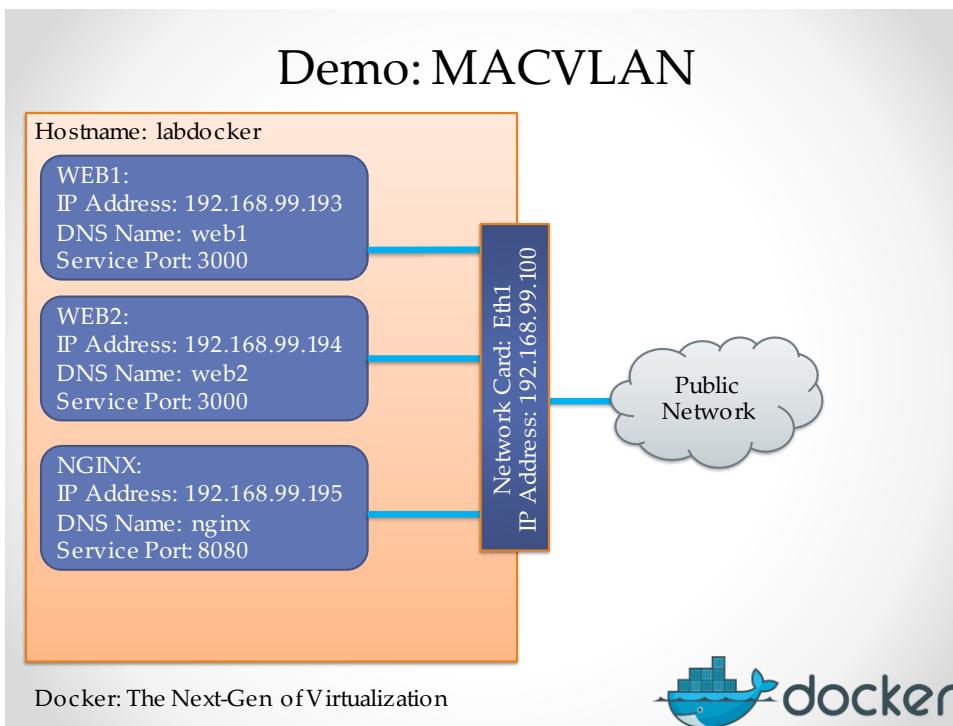
Medium Jenkins vagrant Mesos Vue docker NGINX Taiwan Kubernetes PWA_Progressive_W... MySQL_Cluster GeneralKB Node

• Note: Linux Macvlan interface types are not able to ping or communicate with the default namespace IP address. For example, if you create a container and try to ping the Docker host's eth0, it will not work. That traffic is explicitly filtered by the kernel to offer additional provider isolation and security. This is a common gotcha when a user first uses those Linux interface types since it is natural to ping local addresses when testing.

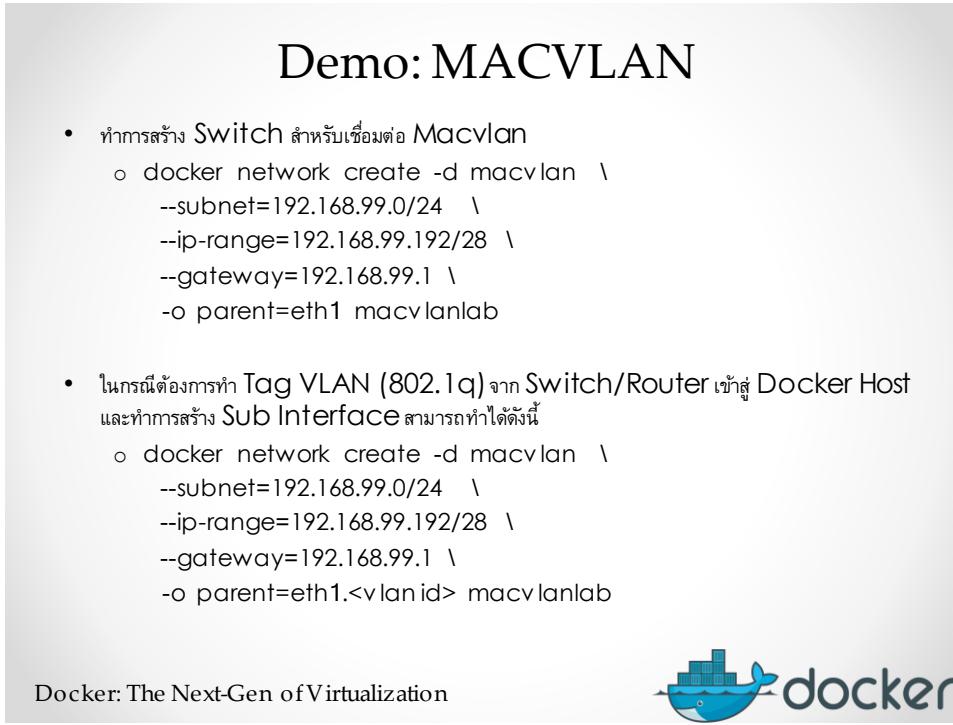
Docker: The Next-Gen of Virtualization



156



157



158

Demo: MACVLAN

- สร้างเครื่อง Webserver (nodejs) และ Nginx
 - Web1**
 - docker run -dt --name web1 --net macvlanlab \
 --net-alias web1 labdocker/alpineweb:web1 node hello.js
- Web2**
- docker run -dt --name web2 --net macvlanlab \
 --net-alias web2 labdocker/alpineweb:web2 node hello.js
- NGINX**
- docker run -dt --name nginx --net macvlanlab \
 labdocker/nginx:labnetworkhttp2

Docker: The Next-Gen of Virtualization



159

Demo: MACVLAN

- Internal test:

```

[docker@labdocke:~$ docker container exec -it nginx curl http://web1:3000
Hello World Container Docker for Nodejs Number 1
[docker@labdocke:~$ docker container exec -it nginx curl http://web1:3000 -I
HTTP/1.1 200 OK
Content-Type: text/plain
Date: Sat, 22 Jun 2019 03:39:07 GMT
Connection: keep-alive

[docker@labdocke:~$ docker container exec -it nginx curl http://web2:3000
Hello World Container Docker for Nodejs Number 2
[docker@labdocke:~$ docker container exec -it nginx curl http://web2:3000 -I
HTTP/1.1 200 OK
Content-Type: text/plain
Date: Sat, 22 Jun 2019 03:39:13 GMT
Connection: keep-alive

[docker@labdocke:~$ docker container exec -it nginx ping web1
PING web1 (192.168.99.193): 56 data bytes
64 bytes from 192.168.99.193: seq=0 ttl=64 time=0.042 ms
^C
--- web1 ping statistics ---
1 packets transmitted, 1 packets received, 0% packet loss
round-trip min/avg/max = 0.042/0.042/0.042 ms
[docker@labdocke:~$ docker container exec -it nginx ping web2
PING web2 (192.168.99.194): 56 data bytes
64 bytes from 192.168.99.194: seq=0 ttl=64 time=0.115 ms
64 bytes from 192.168.99.194: seq=1 ttl=64 time=0.069 ms
^C
--- web2 ping statistics ---
2 packets transmitted, 2 packets received, 0% packet loss
round-trip min/avg/max = 0.069/0.092/0.115 ms
[docker@labdocke:~$ 

```

Docker: The Next-Gen of Virtualization



160

Demo: MACVLAN

- External Test:

```

10:0-1-55: ~ -- bash ... emultinodejs -- bash ... ut.e.amazonaws.com ...
praparns-MacBook-Pro:~ praparns$ ping 192.168.99.193
PING 192.168.99.193 (192.168.99.193): 64 bytes data
64 bytes from 192.168.99.193: icmp_seq=1 ttl=64 time=0.346 ms
64 bytes from 192.168.99.193: icmp_seq=2 ttl=64 time=0.508 ms
...
C
--- 192.168.99.193 ping statistics ---
2 packets transmitted, 2 packets received, 0.0% packet loss
round-trip min/avg/max/stddev = 0.346/0.427/0.508/0.081 ms
praparns-MacBook-Pro:~ praparns$ ping 192.168.99.194
PING 192.168.99.194 (192.168.99.194): 64 bytes data
64 bytes from 192.168.99.194: icmp_seq=1 ttl=64 time=0.365 ms
64 bytes from 192.168.99.194: icmp_seq=2 ttl=64 time=0.470 ms
...
C
--- 192.168.99.194 ping statistics ---
2 packets transmitted, 2 packets received, 0.0% packet loss
round-trip min/avg/max/stddev = 0.365/0.417/0.470/0.052 ms
praparns-MacBook-Pro:~ praparns$ ping 192.168.99.195
PING 192.168.99.195 (192.168.99.195): 64 bytes data
64 bytes from 192.168.99.195: icmp_seq=1 ttl=64 time=0.357 ms
64 bytes from 192.168.99.195: icmp_seq=2 ttl=64 time=0.331 ms
...
C
--- 192.168.99.195 ping statistics ---
2 packets transmitted, 2 packets received, 0.0% packet loss
round-trip min/avg/max/stddev = 0.331/0.344/0.357/0.013 ms
praparns-MacBook-Pro:~ praparns$ curl http://192.168.99.195:8080/nodejs
Hello World Container Docker for Nodejs Number 1
praparns-MacBook-Pro:~ praparns$ curl http://192.168.99.195:8080/nodejs
Hello World Container Docker for Nodejs Number 1
praparns-MacBook-Pro:~ praparns$ curl http://192.168.99.193:3000
Hello World Container Docker for Nodejs Number 1
praparns-MacBook-Pro:~ praparns$ curl http://192.168.99.194:3000
Hello World Container Docker for Nodejs Number 1
praparns-MacBook-Pro:~ praparns$ curl https://192.168.99.195:8443/nodejs -k
Hello World Container Docker for Nodejs Number 1
praparns-MacBook-Pro:~ praparns$ curl https://192.168.99.195:8443/nodejs -k
Hello World Container Docker for Nodejs Number 2
praparns-MacBook-Pro:~ praparns$ curl https://192.168.99.194:8443/nodejs -k
Hello World Container Docker for Nodejs Number 2
praparns-MacBook-Pro:~ praparns$ curl https://192.168.99.193:8443/nodejs -k
Hello World Container Docker for Nodejs Number 2

```

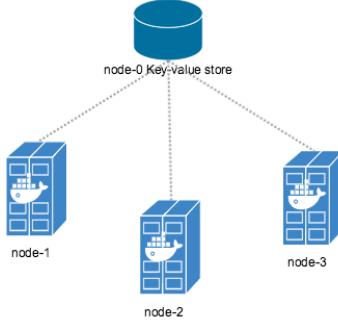
Docker: The Next-Gen of Virtualization



161

Network

- Network across host (Overlay Network)



Docker: The Next-Gen of Virtualization



162

Network

- 3rd Party Network Plugin

Plugin	Description
Contiv Networking	An open source network plugin to provide infrastructure and security policies for a multi-tenant micro services deployment, while providing an integration to physical network for non-container workload. Contiv Networking implements the remote driver and IPAM APIs available in Docker 1.9 onwards.
Kuryr Network Plugin	A network plugin is developed as part of the OpenStack Kuryr project and implements the Docker networking (libnetwork) remote driver API by utilizing Neutron, the OpenStack networking service. It includes an IPAM driver as well.
Weave Network Plugin	A network plugin that creates a virtual network that connects your Docker containers - across multiple hosts or clouds and enables automatic discovery of applications. Weave networks are resilient, partition tolerant, secure and work in partially connected networks, and other adverse environments - all configured with delightful simplicity.

Docker: The Next-Gen of Virtualization



163

Volume

• • •

Docker: The Next-Gen of Virtualization



164

Volume

- ตามปกติแล้ว docker จะเก็บข้อมูลทุกอย่างเอาไว้ภายใน container
 - /var/log
 - /sys/data
 - /etc/nagios/
 - /etc/mysql/
 - Etc.
- สำหรับการใช้งาน container ใน production environment docker ได้สร้าง tool สำหรับอำนวยความสะดวกในการใช้งานข้อมูลร่วมกันในหลาย ๆ กรณี อาทิ เช่น
 - การ share ข้อมูลร่วมกันระหว่าง Container
 - การ share ข้อมูลร่วมกันระหว่าง Container / host
 - จัดเก็บ configuration / log / data ของทุกๆ container ไว้ที่ศูนย์กลาง
 - data migration
 - backup / restore

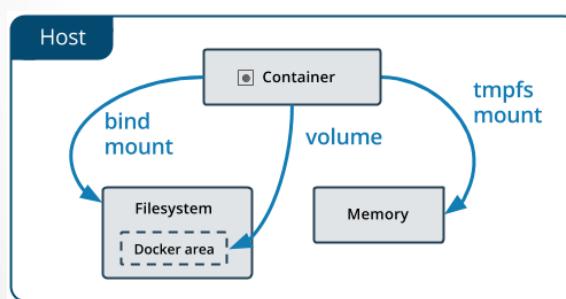
Docker: The Next-Gen of Virtualization



165

Volume

- รูปแบบการใช้งาน Volume บน Docker
 - Bind Mount (Map path with Host Directly)
 - Volumes Mount (Local, Special Driver)
 - Tmpfs Mount (Memory Only)
- พารามิเตอร์ในการใช้งาน (`--mount`, `-v (Obsolete)`)



Docker: The Next-Gen of Virtualization



166

Volume

- 1. Bind Mount: map volume ระหว่าง host directory และ container directory

```
-v /volume-host:/volume-container:(ro/rw)
```

```
--mount type=bind,source=/volume-host  
,target=/volume-container,(readonly)
```

Ex: docker container run -dt -v /etc/nginx.conf:/etc/nginx.conf:ro

Ex: docker container run -dt --mount type=bind, \
source=/etc/nginx.conf,target=/etc/nginx.conf,readonly

- ตรวจสอบการคอนฟิกบัน Container (inspect)

```
"HostConfig": {  
    "Binds": [  
        "/var/log:/var/log:ro"],
```

Docker: The Next-Gen of Virtualization



167

Volume

- 2. Volumes สร้าง volume กลางเพื่อใช้ในการจัดเก็บข้อมูล หรือ share volume ระหว่าง host/container/container (local: /var/lib/docker/volumes, driver: 3rd Party)
 - สร้าง Volume สำหรับใช้เก็บข้อมูล

```
docker volume create --driver <xxx> --opt <xxx> <name>
```

Ex: docker volume create --driver local datavol

Ex: docker volume create --driver local \
--opt type=nfs --opt o=addr=192.168.99.100,rw \
--opt device=/nfs_share datavol

Docker: The Next-Gen of Virtualization



168

Volume

- ☞ Volumes (-v option is not allow for service level (Swarm/Compose))

Ex: docker volume create --driver local datavol

Ex: docker container run -dt --name web1 \
-v datavol:/data labdocker/alpine sh

Ex: docker container run -dt --name web2 \
-v datavol:/data labdocker/alpine sh

Ex: docker container run -dt --name web1 \
--mount source=datavol,target=/data labdocker/alpine sh

Ex: docker container run -dt --name web2 \
--mount source=datavol,target=/data labdocker/alpine sh

Docker: The Next-Gen of Virtualization



169

Volume

- ☞ ลบ volume ที่จาก container ให้ใช้คำสั่ง docker volume rm เพื่อลบ volume

docker volume rm <volume name>

- Third Party Volume Plugin (Since 1.8)

docker volume create --driver=<third party> <volume name>

Ex: docker volume create --driver=gce --name gce-disk \
-o SizeGb=90

Ex: docker run -dt -v gce-disk:/store/database alpine sh

Docker: The Next-Gen of Virtualization



170

Volume

Name	Description
Azure File Storage plugin	Lets you mount Microsoft Azure File Storage shares to Docker containers as volumes using the SMB 3.0 protocol. Learn more.
BeefsFS Volume Plugin	An open source volume plugin to create persistent volumes in a BeefsFS parallel file system.
Blockbridge plugin	A volume plugin that provides access to an extended set of container-based persistent storage options. It supports single and multihost Docker environments with features that include tenant isolation, automated provisioning, encryption, secure deletion, snapshots and QoS.
Ceph Volume Plugin	An open source volume plugin that provides multi-tenant, persistent, distributed storage with intent based consumptions. It has support for Ceph and NFS.
Cgroup plugin	A volume plugin for a variety of storage backends, including device mapper and NFS. It's a simple standalone executable written in Go and provides the framework to support vendor specific extensions such as snapshots, backups and restore.
DigitalOcean Block Storage plugin	Integrates DigitalOcean's block storage solution into the Docker ecosystem by automatically attaching a given block storage volume to a DigitalOcean droplet and making the contents of the volume available to Docker containers running on that droplet.
DRBD plugin	A volume plugin that provides highly available storage replicated by DRBD . Data written to the docker volume is replicated in a cluster of DRBD nodes.
EbsVolume plugin	A volume plugin that provides multihost portable volumes for Docker, enabling you to run databases and other stateful containers and move them around across a cluster of machines.
Easi Volume Plugin	A volume plugin that is developed as part of the OpenStack Kuryr project and implements the Docker volume plugin API by utilizing Cinder, the OpenStack block storage service.
GlusterFS plugin	A volume plugin able to attach, format and mount Google Compute Persistent Disk .
Horcrux Volume Plugin	A volume plugin that provides multihost volumes management for Docker using GlusterFS.
HPE 3Par Volume Plugin	A volume plugin that allows on-demand, version controlled access to your data. Horcrux is an open-source plugin, written in Go, and supports SCP, Minio and Amazon S3.
iSCSI Volume Plugin	An open source volume plugin that allows using an iSCSI filesystem as a volume.
Kvstore plugin	A plugin that provides credentials and secret management using Keyvibiz as a central repository.
Local Persistent Plugin	A volume plugin that extends the default local driver's functionality by allowing you specify <code>mountpoint</code> anywhere on the host, which enables the files to <i>always persist</i> , even if the volume is removed via docker volume rm.
NetApp Plugin(DVP)	A volume plugin that provides direct integration with the Docker ecosystem for the NetApp storage portfolio. The nDVP package supports the provisioning and management of storage resources from the storage platform to Docker hosts, with a robust framework for adding additional platforms in the future.
Nimble plugin	A volume plugin that provides volume management for NFS 3/4, AWS EBS and CIFS file systems.
Nimble Storage Volume Plugin	A volume plugin that integrates with Nimble Storage Unified Flash Fabric arrays. The plugin abstracts array volume capabilities to the Docker administrator to allow self-provisioning of secure multi-tenant volumes and clones.
OpenStorage Plugin	A cluster-aware volume plugin that provides volume management for file and block storage solutions. It implements a vendor neutral specification for implementing extensions such as CoS, encryption, and snapshots. It has example drivers based on FUSE, NFS, NBD and HBS to name a few.
Paragon Volume Plugin	A volume plugin that turns any server into a scale-out converged compute/storage node, providing container granular storage and highly available volumes across any node, using a sharding/roofing storage backend that works with any docker scheduler.
QEMU Volume Plugin	A volume plugin that connects Docker to QEMU 's data center file system, a general-purpose, scalable and fault-tolerant storage platform.
RDXCopy plugin	A volume plugin which is written in Go and provides advanced storage functionality for many platforms including VirtualBox, EC2, Google Compute Engine, OpenStack, and BMC.
Virtazzzo Storage and Block plugin	A volume plugin with support for Virtazzzo Storage distributed cloud file system as well as ploop devices.
VMware Sphere Storage Plugin	Docker Volume Driver for Sphere enables customers to address persistent storage requirements for Docker containers in Sphere environments.

Docker: The Next-Gen of Virtualization



171

Volume

- ms backup volume જીનું container

Ex:

```
docker container run -it --rm --mount source=datavol,target=/data \
--mount type=bind,source=$(pwd),target=/backup \
tar cvf /backup/vol001.tar /data
```

- ms restore volume જીનું container

Ex:

```
docker container run -it --rm --mount source=datavol,target=/data \
--mount type=bind,source=$(pwd),target=/backup \
bash -c "cd /data && tar xvf /backup/vol001.tar --strip 1"
```

Docker: The Next-Gen of Virtualization



172

Volume

- 3. tmpfs mount ใช้เพื่อจัดเก็บข้อมูลใน memory ไม่บัน non-persistent space โดย tmpfs ไม่สามารถ Share ระหว่าง container และไม่สามารถใช้งานบน Windows ได้

```
docker volume create --driver <xxx> --opt <xxx> <name>
```

Ex: docker volume create --driver local --opt type=tmpfs \ --opt device=tmpfs --opt o=size=100m,uid=1000 tmptest

Ex: docker container run -dt --name tmptest --mount type=tmpfs,destination=/app nginx:latest

Ex: docker container run -dt --name tmptest --tmpfs /app \ nginx:latest

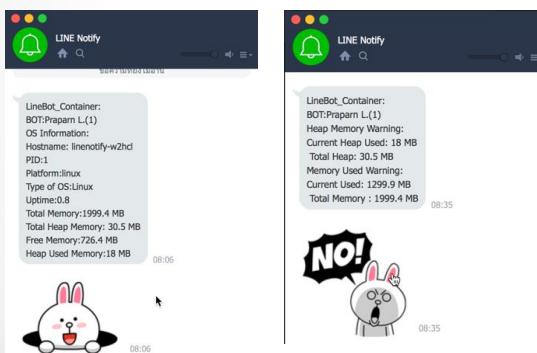
Docker: The Next-Gen of Virtualization



173

Workshop: Share Volume

- Part 1: Host Path debug program
- Purpose: ทำการ Map volume เพื่อนำ source code เข้าไปทำงาน debug/run บน container
- Example on WorkShop: "Monitor and LINE notify container"



LINE

Docker: The Next-Gen of Virtualization



174

Workshop: Share Volume

- Generate LINE Token
 - <https://notify-bot.line.me>

Docker: The Next-Gen of Virtualization



175

Workshop: Share Volume

- Generate LINE Token
 - <https://notify-bot.line.me>

Docker: The Next-Gen of Virtualization



176

Workshop: Share Volume

```

1 const Monitor = require('monitor');
2 const request = require('request');
3 const TITLE=process.env.TITLE;
4 const INTERVAL=process.env.INTERVAL;
5 const HEAP_HIGH =process.env.HEAP_HIGH;
6 const MEM_HIGH =process.env.MEM_HIGH;
7 const SH_OS =process.env.SH_OS;
8 const TOKEN = process.env.TOKEN;
9 const critical_StickerPkg = 2;
10 const critical_StickerId = 39;
11 const information_StickerPkg = 2;
12 const information_StickerId = 34;
13 var options = {
14   probeClass: 'Process',
15   initParams: {
16     | pollInterval: INTERVAL
17   }
18 }
19 var processMonitor = new Monitor(options);
20 var stickerPkg=0; //stickerPackageId
21 var stickerId=0; //stickerId
22 processMonitor.on('change', () => {
23   var sendNotify="N" //final decision to send notify
24   var heapWarning="N" //flag for heap memory warning
25   var memoryWarning="N" //flag for memory warning
26   var processMonitor = new Monitor(options);
27   var stickerPkg=0; //stickerPackageId
28   var stickerId=0; //stickerId
29   processMonitor.on('change', () => {
30     var sendNotify="N" //final decision to send notify
31     var heapWarning="N" //flag for heap memory warning
32     var memoryWarning="N" //flag for memory warning
33     var processMonitor = new Monitor(options);
34     var stickerPkg=0; //stickerPackageId
35     var stickerId=0; //stickerId
36     processMonitor.on('change', () => {
37       var sendNotify="N" //final decision to send notify
38       var heapWarning="N" //flag for heap memory warning
39       var memoryWarning="N" //flag for memory warning
40       var processMonitor = new Monitor(options);
41       var stickerPkg=0; //stickerPackageId
42       var stickerId=0; //stickerId
43       processMonitor.on('change', () => {
44         var sendNotify="N" //final decision to send notify
45         var heapWarning="N" //flag for heap memory warning
46         var memoryWarning="N" //flag for memory warning
47         var processMonitor = new Monitor(options);
48         var stickerPkg=0; //stickerPackageId
49         var stickerId=0; //stickerId
50         processMonitor.on('change', () => {
51           var sendNotify="N" //final decision to send notify
52           var heapWarning="N" //flag for heap memory warning
53           var memoryWarning="N" //flag for memory warning
54           var processMonitor = new Monitor(options);
55           var stickerPkg=0; //stickerPackageId
56           var stickerId=0; //stickerId
57           processMonitor.on('change', () => {
58             var sendNotify="N" //final decision to send notify
59             var heapWarning="N" //flag for heap memory warning
60             var memoryWarning="N" //flag for memory warning
61             var processMonitor = new Monitor(options);
62             var stickerPkg=0; //stickerPackageId
63             var stickerId=0; //stickerId
64             processMonitor.on('change', () => {
65               var sendNotify="N" //final decision to send notify
66               var heapWarning="N" //flag for heap memory warning
67               var memoryWarning="N" //flag for memory warning
68               var processMonitor = new Monitor(options);
69               var stickerPkg=0; //stickerPackageId
70               var stickerId=0; //stickerId
71               processMonitor.on('change', () => {
72                 var sendNotify="N" //final decision to send notify
73                 var heapWarning="N" //flag for heap memory warning
74                 var memoryWarning="N" //flag for memory warning
75                 var processMonitor = new Monitor(options);
76                 var stickerPkg=0; //stickerPackageId
77                 var stickerId=0; //stickerId
78                 processMonitor.on('change', () => {
79                   var sendNotify="N" //final decision to send notify
80                   var heapWarning="N" //flag for heap memory warning
81                   var memoryWarning="N" //flag for memory warning
82                   var processMonitor = new Monitor(options);
83                   var stickerPkg=0; //stickerPackageId
84                   var stickerId=0; //stickerId
85                   processMonitor.on('change', () => {
86                     var sendNotify="N" //final decision to send notify
87                     var heapWarning="N" //flag for heap memory warning
88                     var memoryWarning="N" //flag for memory warning
89                     var processMonitor = new Monitor(options);
90                     var stickerPkg=0; //stickerPackageId
91                     var stickerId=0; //stickerId
92                     processMonitor.on('change', () => {
93                       var sendNotify="N" //final decision to send notify
94                       var heapWarning="N" //flag for heap memory warning
95                       var memoryWarning="N" //flag for memory warning
96                       var processMonitor = new Monitor(options);
97                       var stickerPkg=0; //stickerPackageId
98                       var stickerId=0; //stickerId
99                     })
100                   })
101                 })
102               })
103             })
104           })
105         })
106       })
107     })
108   })
109 
```

Docker: The Next-Gen of Virtualization



177

Workshop: Share Volume

The screenshot shows a Mac desktop with two windows open. On the left is a Terminal window titled 'Terminal MAC Pro — ssh + docker-machine ssh labdocke — 102>25'. It contains several lines of command-line text related to Docker and memory usage. On the right is a 'LINE Notify' window titled 'Line Notify' with a message from 'LineBot_Container'. The message includes a 'NO!' icon, some text, and a timestamp of '11:30'.

```

Terminal MAC Pro — ssh + docker-machine ssh labdocke — 102>25
mit": "50", "x-rateLimit-remaining": "997", "x-rateLimit-imageryRemaining": "50", "x-rateLimit-reset": "1501996
936", "request": {"url": "https://notify-api.line.me/", "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 EleyqillzKzpaCoIRreb2TcGbIyzDZ
HpoJicdd6t6CX", "content-length": "343"}}, {"\\"status\": 200, \\"message\": "\\"ok\\""}]
'C
/nodejs # export HEAP_HIGH=20
/nodejs # export MEM_HIGH=20
/nodejs # export SH_OS=N
/nodejs # node index.js
null
{"statusCode":200,"body": "[{"status":200,"message":"\\\"ok\\\""}, {"headers":{"server": "nginx", "date": "Sun, 06 Aug 2017 04:30:41 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": "997", "x-rateLimit-ImageRemaining": "50", "x-rateLimit-reset": "1501996
936"}, "request": {"url": "https://notify-api.line.me/", "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 EleyqillzKzpaCoIRreb2TcGbIyzDZ
HpoJicdd6t6CX", "content-length": "290"}}, {"\\"status\": 200, \\"message\": "\\"ok\\\""}]
'C
/nodejs #

```

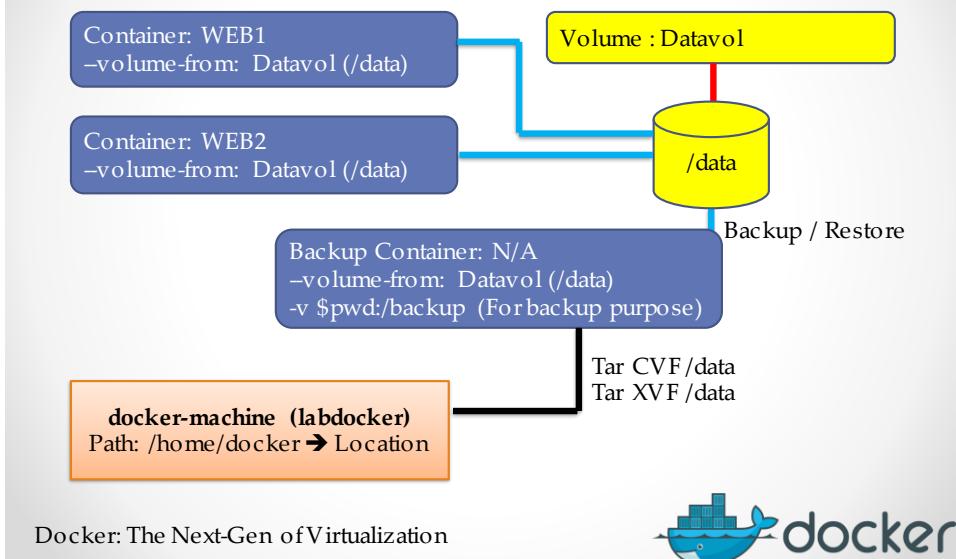
Docker: The Next-Gen of Virtualization



178

Workshop: Share Volume

- Part 2: Container Volume



179

Log and Inspect

• • •

Docker: The Next-Gen of Virtualization



180

Log

- เพื่อตรวจสอบ log การทำงานบน container อย่างต่อเนื่อง docker ได้เตรียม tool ในการตรวจสอบ log การทำงาน

```
docker container logs <options> <container name>
```

- options ในการดูล็อกไฟล์
 - f, --follow เพื่อตรวจสอบล็อกไฟล์ใหม่ตลอดเวลา
 - t, --timestamps กำหนดให้สวัสดีและเวลาในล็อกไฟล์
 - since= <unix timestamps> กำหนดเวลาเริ่มตรวจสอบล็อกไฟล์ หรือกำหนด yyyy-mm-dd
 - tail = "all" / number of line
 - until แสดง log ก่อนเวลาที่กำหนด (Ex: 20180619220001) หรือ Relative (Ex: 42m = 42 minute before)
 - details แสดงรายละเอียดเพิ่มเติมของล็อกเช่น Env variable, labels โดยจะเป็นรายละเอียดเพิ่มเติมจากการใส่ option --log-opt ตอนสร้าง container

Docker: The Next-Gen of Virtualization



181

Log

- Server side. Big problem will come with logging !!!
- All container log will keep on "/var/lib/docker/containers/<container id>/<container-id>-json.log" and it take mega size on this path until full
- We have 2 option for manage this
- Global Configuration
 - Add daemon.json on path /etc/docker/daemon.json for global apply (need to restart docker daemon for take effect)

```
Workshop-1-8-Inspect-Logs > {} daemon.json > ...
1  [
2   "log-driver": "json-file",
3   "log-opts": {
4     "max-size": "10m",
5     "max-file": "10"
6   }
7 ]
```

Docker: The Next-Gen of Virtualization



182

Log

Driver	Description
none	No logs are available for the container and <code>docker logs</code> does not return any output.
local	Logs are stored in a custom format designed for minimal overhead.
json-file	The logs are formatted as JSON. The default logging driver for Docker.
syslog	Writes logging messages to the <code>syslog</code> facility. The <code>syslog</code> daemon must be running on the host machine.
journald	Writes log messages to <code>journald</code> . The <code>journald</code> daemon must be running on the host machine.
gelf	Writes log messages to a Graylog Extended Log Format (GELF) endpoint such as Graylog or Logstash.
fluentd	Writes log messages to <code>fluentd</code> (forward input). The <code>fluentd</code> daemon must be running on the host machine.
awslogs	Writes log messages to Amazon CloudWatch Logs.
splunk	Writes log messages to <code>splunk</code> using the HTTP Event Collector.
etwlogs	Writes log messages as Event Tracing for Windows (ETW) events. Only available on Windows platforms.
gclogs	Writes log messages to Google Cloud Platform (GCP) Logging.
logentries	Writes log messages to Rapid7 Logentries.

Ref <https://docs.docker.com/config/containers/logging/configure/>

Docker: The Next-Gen of Virtualization



183

Log

- Container Configuration
 - Specific on run command of docker container
 - `--log-driver json-file` (default)
 - `--log-opt max-size=10m`
 - `--log-opt max-file=10`

```
ubuntu@ip-10-0-1-55:~$ docker container run -dt --name nginx -p 80:80 \
> --log-driver json-file \
> --log-opt max-size=10m \
> --log-opt max-file=10 \
> labdocker/nginx:badversion
07af88d6368bdf8a74608882963a50052e5c887fc32da812369d78068643e51
ubuntu@ip-10-0-1-55:~$
```

Docker: The Next-Gen of Virtualization



184

Inspect

- การตรวจสอบค่าคอนฟิกของ container / network / volume etc

```
docker inspect <container>
```

```
docker network inspect
```

```
docker volume inspect
```

Docker: The Next-Gen of Virtualization



185

Workshop: Log & Inspect

- ทดสอบสร้าง container nginx และ map port 80 ดังนี้

```
docker container run -dt --name nginx \
-p 80:8080 labdocker/nginx: badversion
```

- ตรวจสอบค่าคอนฟิกภายใน container ด้วยคำสั่ง inspect

```
docker container inspect nginx
```

- ตรวจสอบล็อกการทำงานภายใน container ด้วยคำสั่ง logs

```
docker container logs nginx
```

Docker: The Next-Gen of Virtualization



186

Commit

• • •

Docker: The Next-Gen of Virtualization



187

Commit

- เมื่อต้องการเก็บ container ที่ทำการรันเรียบข้อเป็น snap image เมื่ogeneration สามารถทำได้ผ่านคำสั่ง “commit”

`docker container commit <options> <container><image name:tag>`

- options
 - a, --author=" " ระบุผู้จัดทำ image
 - c, --change = [] เพิ่มเติม command ที่จะจัดเก็บภายใน image
 - m, --message = " " ระบุ comment ใน image ที่จัดเก็บ
 - p, --pause=true ก้าหนดให้หยุดการใช้งาน container ชั่วคราวในระหว่างทำการ commit

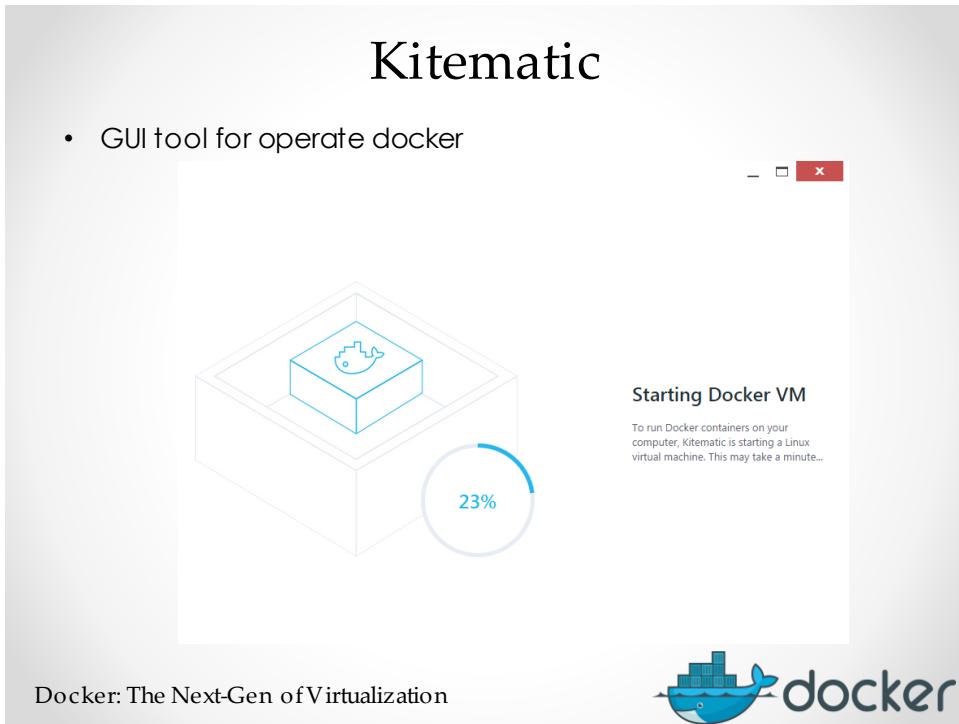
Docker: The Next-Gen of Virtualization



188



189



190

• GUI tool for operate docker

Docker: The Next-Gen of Virtualization



191

Recapture for Day 1

- Docker principle
- Docker machine
- Image, Repository & Tag, container
- CPU, Memory and I/O
- Network
- Volume
- Inspect and Log
- Commit
- Kitematic

Docker: The Next-Gen of Virtualization



192

Q&A for Day 1

• • •

Docker: The Next-Gen of Virtualization



193

A photograph of a container yard with numerous shipping containers stacked in rows under a clear blue sky with white clouds. Overlaid on the image are several promotional elements:

- A large blue rectangular box in the top left corner contains the Docker logo.
- A smaller white circular badge in the bottom left corner contains the Docker Swarm logo, which features a blue whale icon and the word "SWARM".
- A large blue text overlay in the center-right reads "Docker:Zero to Hero (Day 2)".
- A blue text box at the bottom left provides author information: "By Praparn Luengphoonlap" and "Email: eva10409@gmail.com".
- A small white rectangular box in the top right corner contains the KhaDev logo, which features a green and yellow circular icon with the word "KhaDev".

Docker: The Next-Gen of Virtualization



194

Outline Day 2

- Dockerfile and Build
- Compose
- Docker Security
- Registry
- Swarm Mode
 - Conceptual of Swarm
 - Swarm Mode Architecture
 - Swarm init/join cluster system
 - Swarm service
 - Orchestrator Assignment
 - Config and Secret
 - Network Overlay and Ingress
 - HA Manager Role
 - Docker Stack Deploy (Compose Swarm Mode)
- Portainer for Docker
- Q&A

Docker: The Next-Gen of Virtualization



195

Dockerfile and Build

• • •

Docker: The Next-Gen of Virtualization



196

Dockerfile

- docker file คือการคำสั่งที่ใช้สร้าง image ขึ้นเพื่อใช้งานโดยเฉพาะ และสามารถกำหนดเป็นมาตรฐานสำหรับการทำงานในองค์กรโดยสร้าง text file ตาม syntax ที่ทาง docker กำหนดไว้
 - FROM
 - M AINTAINER
 - RUN (shell),["exec","parameter1","parameter2","etc"]
 - C M D (shell),["exec","parameter1","parameter2","etc"]
 - EXPOSE
 - ARGS
 - ENV
 - COPY /ADD (source) (Destination), [(source) (destination)]
 - ENTRYPOINT
 - WORKDIR
 - Etc(<https://docs.docker.com/engine/reference/builder>)
- เมื่อสร้าง dockerfile เสร็จแล้วสามารถดึง image ผ่านคำสั่ง build

```
docker build <option> <path of dockerfile>
```

Docker: The Next-Gen of Virtualization



197

Dockerfile

- Best practice** ในการสร้าง dockerfile
 - simple is the best
 - 1 container / 1 service
 - เลือก source image จาก official owner
 - สร้าง dockerfile โดยพยายามให้มี footprint ต่ำที่สุด
 - สร้าง dockerfile บน path ที่สร้างขึ้นเฉพาะงาน
 - วางเฉพาะไฟล์ที่จำเป็นต้องใช้งานบน path
 - install component เท่าที่จำเป็นต้องใช้งานบน container
 - คำสั่ง run เพื่อติดตั้ง software รวมยอดใน 1 ชุดคำสั่ง
 - ควรมีการลบ temp ที่เกิดขึ้นจากการติดตั้ง software เพื่อลดขนาดของ image
 - จัดเรียง dockerfile ให้อ่านง่าย
 - apt-update && apt-install -y \
 - curl \
 - automake \
 - ควรระบุ EXPOSE port ที่ออกแนวให้ใช้งานໄว์ในทุกๆครั้ง

Docker: The Next-Gen of Virtualization



198

Dockerfile

```
RUN apt-get update && apt-get install -y \
    aufs-tools \
    automake \
    build-essential \
    curl \
    dpkg-sig \
    libcap-dev \
    libsqlite3-dev \
    mercurial \
    reprepro \
    ruby1.9.1 \
    ruby1.9.1-dev \
    s3cmd=1.1.* \
&& rm -rf /var/lib/apt/lists/*
```

Docker: The Next-Gen of Virtualization



199

Dockerfile

```
FROM php:5.6-apache
MAINTAINER Volker Wiegand <volker.wiegand@cvw.de>

RUN a2enmod rewrite

RUN apt-get update && apt-get install -y \
    libpng12-dev \
    libjpeg-dev \
    libpq-dev \
    libxml2-dev \
    vim-tiny \
    && docker-php-ext-configure gd --with-png-dir=/usr --with-jpeg \
    && docker-php-ext-install gd mbstring mysql mysqli soap \
    && rm -rf /var/lib/apt/lists/* /var/www/html/index.html

ENV MANTIS_VER 1.3.2
ENV MANTIS_MD5 f30acb6d41ba757b7c09f922a0f68b06
ENV MANTIS_URL https://sourceforge.net/projects/mantisbt/files/mantis-
ENV MANTIS_FILE mantisbt.tar.gz

RUN mkdir -p /var/lib/mantisbt && cd /var/lib/mantisbt \
    && curl -fSL ${MANTIS_URL} -o ${MANTIS_FILE} \
    && echo "${MANTIS_MD5} ${MANTIS_FILE}" | md5sum -c \
    && tar -xz --strip-components=1 -f ${MANTIS_FILE} \
    && rm ${MANTIS_FILE} \
    && chown -R www-data:www-data .
```

Docker: The Next-Gen of Virtualization



200

Dockerfile

- ควรระบุ WORKDIR เพื่อกำหนด path เริ่มต้นในการทำงานทุกครั้ง (before ENTRYPOINT, RUN, CMD, COPY etc)
- ADD/COPY operation
 - COPY เป็น basic transfer simple file
 - ADD ใช้ในกรณี remote file URL, self extract (.tar) (Not recommend)
- ENV PATH ควรระบุ path ของ executable ที่จะเป็นตัวตั้งตัวนอน
- พิจารณาการใช้ cache ของ image layer โดยสามารถ ignore cache ด้วย option “–no-cache=true”
 - ตามปกติ docker จะเบร์ยนเทิร์บ instruction ที่ก่อหนี้ไว้กับ image layer ที่
 - Consider: instruction, base image

Docker: The Next-Gen of Virtualization



201

How can check dockerfile ?

FROM:latest

```

1 FROM lddocker/alpine:latest
2 MAINTAINER Proppen Luongphouneap <eva1040@gmail.com>
3 LABEL Description="NodeJS/NGINX Build Container" Version="1.0"
4 ENV NODE_VERSION=v4.3.0 NPM_VERSION=2.14.12
5 RUN apk update && \
6     apk add nginx
7 COPY nginx.conf /etc/nginx/nginx.conf
8 WORKDIR /usr/sbin
9 ENTRYPOINT ["/nginx","-c","/etc/nginx/nginx.conf"]
10 EXPOSE 8080

```

Line 5: Consider `--no-cache` or `--update` with `rm -rf /var/cache/apk/*` (Optimization)

Line 1: Base Image Latest Tag (Clarity)

2 issues found Made by REPLICATED

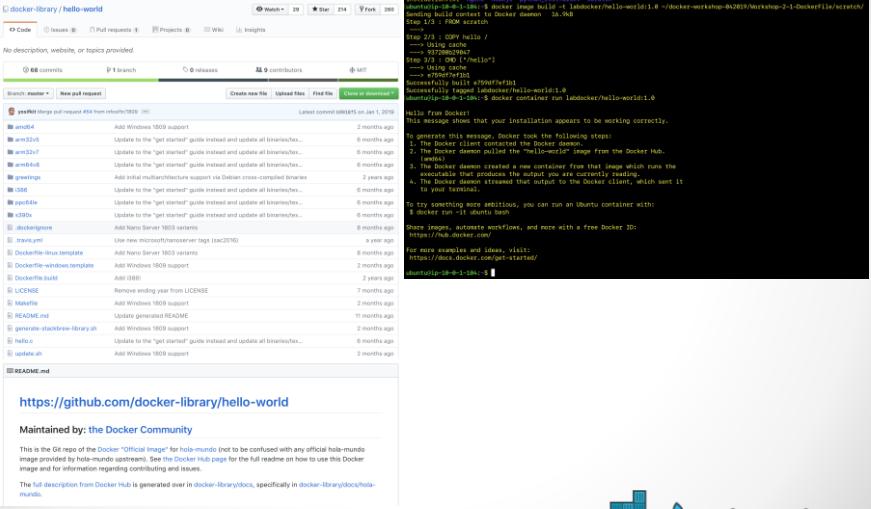
Docker: The Next-Gen of Virtualization



202

Workshop: DockerFile

- Part1: SCRATCH



Docker: The Next-Gen of Virtualization



203

Workshop: DockerFile

- Part2: NGINX/NODEJS

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

Container Name: NGINX
 IP Address: X.X.X.X (Port: 8080:80)

AWS's Machine
Path: ~/dockerworkshop/Workshop-2-1-DockerFile

/nodejs



/nginx



Docker: The Next-Gen of Virtualization



204

Workshop: DockerFile

- ตรวจสอบ dockerfile ตามตัวอย่างด้านล่าง

```
FROM labdocker/alpine:latest
MAINTAINER Praparn Lueangphoonlap (eva10409@gmail.com)
LABEL Description="NodeJS/NGINX Build Container" Version="1.0"
ENV NODE_VERSION=v4.3.0 NPM_VERSION=2.14.12
RUN apk update && \
    apk add nodejs
RUN mkdir /nodejs
COPY hello.js /nodejs/
WORKDIR /nodejs
ENTRYPOINT ["node","hello.js"]
EXPOSE 3000
```

```
docker image build -t labdocker/node:1.0 ~/docker-workshop-072019/Workshop-2-1-DockerFile/nodejs
```

Docker: The Next-Gen of Virtualization



205

Workshop: DockerFile

- Output

```
Sending Build context to Docker daemon 3.072 kB
Step 1 : FROM labdocker/alpine:latest
--> 14f89d0e6257
Step 2 : MAINTAINER Praparn Lueangphoonlap (eva10409@gmail.com)
--> Using cache
--> 10f80fb4997d
Step 3 : LABEL Description "NodeJS/NGINX Build Container" Version "1.0"
--> Using cache
--> 59bc5a927e19
Step 4 : ENV NODE_VERSION v4.3.0 NPM_VERSION 2.14.12
--> Using cache
--> ce23d23959b7
Step 5 : RUN apk update && apk add nodejs
--> Running in b12c752ef6a0
fetch http://dl-4.alpinelinux.org/alpine/v3.3/main/x86_64/APKINDEX.tar.gz
fetch http://dl-4.alpinelinux.org/alpine/v3.3/community/x86_64/APKINDEX.tar.gz
v3.3.1-99-gcd7905f [http://dl-4.alpinelinux.org/alpine/v3.3/main]
v3.3.1-59-g48b0368 [http://dl-4.alpinelinux.org/alpine/v3.3/community]
OK: 5857 distinct packages available
(1/4) Installing libgcc (5.3.0-r0)
(2/4) Installing libstdc++ (5.3.0-r0)
(3/4) Installing libuv (1.7.5-r0)
(4/4) Installing nodejs (4.3.0-r0)
Executing busybox-1.24.1-r7.trigger
OK: 29 kB in 15 packages
--> 8f2b2f208ab8
Removing intermediate container b12c752ef6a0
Step 6 : RUN mkdir /nodejs
--> Running in 2ee51d1af642
--> d3edf2c2f511
Removing intermediate container 2ee51d1af642
Step 7 : COPY hello.js /nodejs/
--> f703ccbc7d4
Removing intermediate container 952935d69cce
Step 8 : CMD nginx -c /etc/nginx/nginx.conf
--> Running in 67940385f5ac
--> ed1fbaf609e20a
```

Docker: The Next-Gen of Virtualization

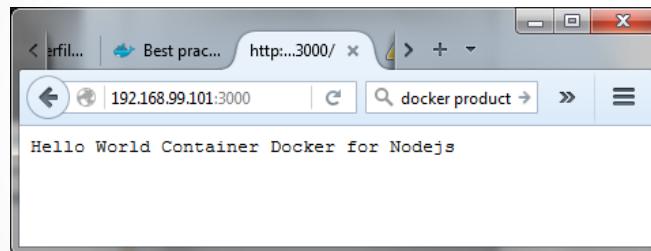


206

Workshop: DockerFile

- Test run

```
docker container run -dt --name NODEJS \
-p 3000:3000 labdocker/node:1.0
```



Docker: The Next-Gen of Virtualization



207

Workshop: DockerFile

- ตรวจสอบ dockerfile ตามตัวอย่างด้านล่าง

```
FROM labdocker/alpine:latest
MAINTAINER Praparn Lueangpho onlap (eva10409@gmail.com)
LABEL Description="NodeJS/NGINX Build Container" Version="1.0"
ENV NODE_VERSION=v4.3.0 NPM_VERSION=2.14.12
RUN mkdir -p /run/nginx
RUN apk update && \
    apk add nginx && \
    rm /etc/nginx/conf.d/default.conf
COPY nginx.conf /etc/nginx/nginx.conf
WORKDIR /usr/sbin
ENTRYPOINT ["nginx", "-c", "/etc/nginx/nginx.conf"]
EXPOSE 8080
```

```
docker build -t labdocker/web:1.0 ~/docker-workshop-
072019/Workshop-2-1-DockerFile/nginx
```

Docker: The Next-Gen of Virtualization



208

Workshop: DockerFile

- Test run

```
docker container run -dt --name NGINX -p 8080:8080 -p 8443:8443 labdocker/web:1.0
```

Docker: The Next-Gen of Virtualization 

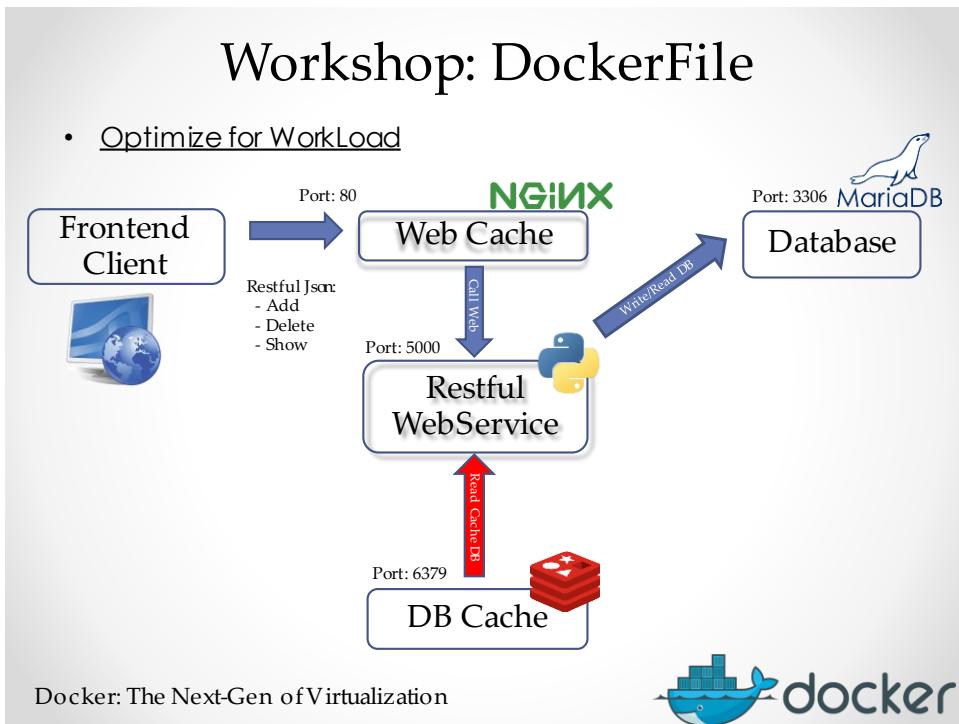
209

Workshop: DockerFile

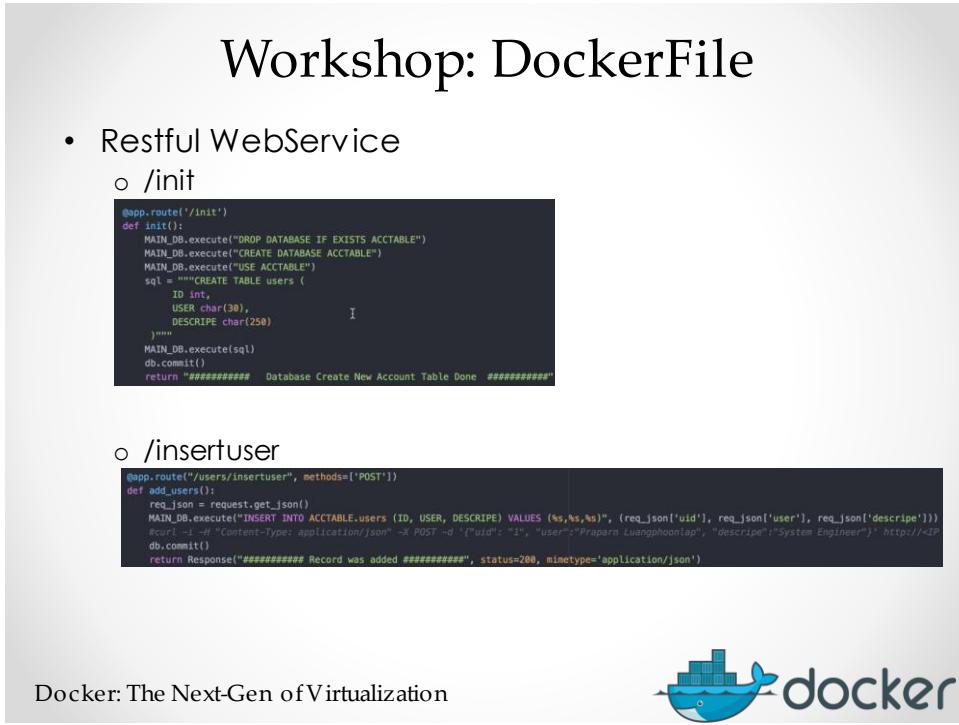
- Part3: Python RESTFUL
- Basic Component

Docker: The Next-Gen of Virtualization 

210



211



212

Workshop: DockerFile

- Restful WebService
 - /removeuser/<uid>

```
@app.route('/users/removeuser/<uid>')
def remove_user(uid):
    hash = hashlib.sha224(str(uid)).hexdigest()
    key = "sql_cache:" + hash
    MAIN_DB.execute("DELETE FROM ACCTABLE.users WHERE ID =" + str(uid))
    db.commit()
    #curl http://<IP Host>:<Port>/users/removeuser/<uid>
    if (CACHE_DB.get(key)):
        CACHE_DB.delete(key)
        return Response("##### Record was deleted (Both Database Cache) #####", status=200, mimetype='application/json')
    else:
        return Response("##### Record was deleted #####", status=200, mimetype='application/json')
```

Docker: The Next-Gen of Virtualization



213

Workshop: DockerFile

- Restful WebService
 - /users/<uid>

```
@app.route('/users/<uid>')
def get_users(uid):
    hash = hashlib.sha224(str(uid)).hexdigest()
    key = "sql_cache:" + hash
    #curl http://<IP Host>:<Port>/users/<uid>
    if (CACHE_DB.get(key)):
        return CACHE_DB.get(key) + "(Database Cache)"
    else:
        MAIN_DB.execute("select USER from ACCTABLE.users where ID=" + str(uid))
        data = MAIN_DB.fetchone()
        if data:
            CACHE_DB.set(key,data[0])
            CACHE_DB.expire(key, 36);
            return CACHE_DB.get(key)
        else:
            return "##### Record not found #####"
```

Docker: The Next-Gen of Virtualization



214

Workshop: DockerFile

- Web Cache (NGINX)

```
http {
    client_max_body_size 500M;
    client_body_timeout 3000s;
    include /etc/nginx/mime.types;
    default_type application/octet-stream;
    log_format main '$remote_addr - $remote_user [$time_local] "$request" '
    '$status $body_bytes_sent "'.$http_referer" '
    '"$http_user_agent" "$http_x_forwarded_for"';

    access_log /var/log/nginx/access.log main;
    tcp_nopush on;

    keepalive_timeout 65;
}

gzip on;

include /etc/nginx/conf.d/*.conf;

server {
    listen 80;
    client_body_buffer_size 50M;
    index index.html      index.htm;
    location / {
        proxy_pass http://webservice:5000;
        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;
    }
}
```

Docker: The Next-Gen of Virtualization



215

Workshop: DockerFile

- Database / Database Cache

```
maindb:
  image: labdocker/mysql:latest
  container_name: maindb
  environment:
    MYSQL_ROOT_PASSWORD: password

cachedb:
  image: labdocker/redis:latest
  container_name: cachedb

webservice:
  build: .
  dockerfile: dockerfile_python
  container_name: webservice
  ports:
    - "5000:5000"
  links:
    - cachedb:cachedb
    - maindb:maindb

webcache:
  build: .
  dockerfile: dockerfile_nginx
  container_name: webcache
  ports:
    - "80:80"
  links:
    - webservice:webservice
```

Docker: The Next-Gen of Virtualization



216

Workshop: DockerFile

- Dockerfile: WebService

```
FROM labdocker/alpinepython:2.7-onbuild
MAINTAINER Praparn Lueangphoonlap (eva10409@gmail.com)
LABEL Description="Python Special Build for Alpine (On Build)" Version="1.0"
EXPOSE 5000
CMD [ "python", "main.py" ]
```

```
docker build -t labdocker/webservice:1.0 \
-f dockerfile_python Path: ~/dockerworkshop/Workshop-2-
1-DockerFile/python_restfulset/
```

Docker: The Next-Gen of Virtualization



217

Workshop: DockerFile

- Dockerfile: Webcache

```
FROM labdocker/alpine:latest
MAINTAINER Praparn Lueangphoonlap (eva10409@gmail.com)
LABEL Description="NGINX Build Container" Version="1.0"
RUN apk update && \
apk add nginx
COPY nginx.conf /etc/nginx/nginx.conf
WORKDIR /usr/sbin
ENTRYPOINT ["nginx", "-c", "/etc/nginx/nginx.conf"]
EXPOSE 8080
```

```
docker build -t labdocker/webcache:1.0 \
-f dockerfile_nginx ~/dockerworkshop/Workshop-2-1-
DockerFile/python_restfulset/
```

Docker: The Next-Gen of Virtualization



218

Workshop: DockerFile

- Test run container

```

praparn-MacBook-Pro:python_restfulset praparn$ docker container ps
CONTAINER ID        IMAGE               COMMAND             CREATED            STATUS              PORTS               NAMES
dc8e81f64982        labdoker/webcache:1.0   "nginx -c /etc/nginx..."   5 seconds ago      Up 3 seconds       0.0.0.0:80->8080/tcp   webcache
702b554db429        labdoker/webservice:1.0  "python main.py"       10 seconds ago    Up 9 seconds       0.0.0.0:5000->5000/tcp   webservice
f29ycf6fe8a         labdoker/redis:latest   "docker-entrypoint..."  22 seconds ago     Up 20 seconds      6379/tcp           cachedb
b6dab1856d6        labdoker/mariadb:latest  "/docker-entrypoint..."  48 seconds ago     Up 47 seconds      3306/tcp          maindb
praparn-MacBook-Pro:python_restfulset praparn$ 
```

...
praparn-MacBook-Pro:python_restfulset praparn\$ export Server_IP=127.0.0.1
praparn-MacBook-Pro:python_restfulset praparn\$ export Server_Port=80
praparn-MacBook-Pro:python_restfulset praparn\$ curl http://\$Server_IP:\$Server_Port/
<H1> Welcome Page from Container Python Lab </H1>
Checkpoint Date/Time: Sun Oct 21 09:36:04 2018
praparn-MacBook-Pro:python_restfulset praparn\$]

Welcome Page from Container Python Lab
Checkpoint Date/Time: Sun Oct 21 09:36:18 2018

Docker: The Next-Gen of Virtualization



219

Workshop: DockerFile

- Init / Insert Data

```

praparn-MacBook-Pro:curl1 praparn$ curl http://$Server_IP:$Server_Port/init
#####
Database Create New Account Table Done #####
praparn-MacBook-Pro:curl1 -i -H "Content-Type: application/json" -X POST -d '{"uid": "1", "user": "Praparn Luangphoohlap", "describe": "Slave"}' http://$Server_IP:$Server_Port/users/insertuser
curl -i -H "Content-Type: application/json" -X POST -d '{"uid": "2", "user": "Somchai Sunsukan", "describe": "Security Guard"}' http://$Server_IP:$Server_Port/users/insertuser
curl -i -H "Content-Type: application/json" -X POST -d '{"uid": "3", "user": "Somachan Panruat", "describe": "House Keeper"}' http://$Server_IP:$Server_Port/users/insertuser
curl -i -H "Content-Type: application/json" -X POST -d '{"uid": "4", "user": "Somchai Panruat", "describe": "House Keeper"}' http://$Server_IP:$Server_Port/users/insertuser
curl -i -H "Content-Type: application/json" -X POST -d '{"uid": "5", "user": "Chatchai Muengang", "describe": "Programmer"}' http://$Server_IP:$Server_Port/users/insertuser
curl -i -H "Content-Type: application/json" -X POST -d '{"uid": "6", "user": "Anusit Kamnaphat", "describe": "Programmer"}' http://$Server_IP:$Server_Port/users/insertuser
curl -i -H "Content-Type: application/json" -X POST -d '{"uid": "7", "user": "Meelarp Meisanuk", "describe": "DevOps Manager"}' http://$Server_IP:$Server_Port/users/insertuser
curl -i -H "Content-Type: application/json" -X POST -d '{"uid": "8", "user": "Penisa Bunbung", "describe": "Security Guard"}' http://$Server_IP:$Server_Port/users/insertuser
curl -i -H "Content-Type: application/json" -X POST -d '{"uid": "9", "user": "Wiphanee Wongsaissawan", "describe": "Administrator"}' http://$Server_IP:$Server_Port/users/insertuser

HTTP/1.1 200 OK
Server: nginx/1.8.1
Date: Sun, 06 Aug 2017 16:06:09 GMT
Content-Type: application/json
Content-Length: 41
Connection: keep-alive
#####
Record was added #####
HTTP/1.1 200 OK
Server: nginx/1.8.1
Date: Sun, 06 Aug 2017 16:06:09 GMT
Content-Type: application/json
Content-Length: 41
Connection: keep-alive
#####
Record was added #####
HTTP/1.1 200 OK
Server: nginx/1.8.1
Date: Sun, 06 Aug 2017 16:06:09 GMT
Content-Type: application/json
Content-Length: 41
Connection: keep-alive
#####
Record was added #####
HTTP/1.1 200 OK
Server: nginx/1.8.1
Date: Sun, 06 Aug 2017 16:06:09 GMT
Content-Type: application/json
Content-Length: 41
Connection: keep-alive
#####
```

Docker: The Next-Gen of Virtualization



220

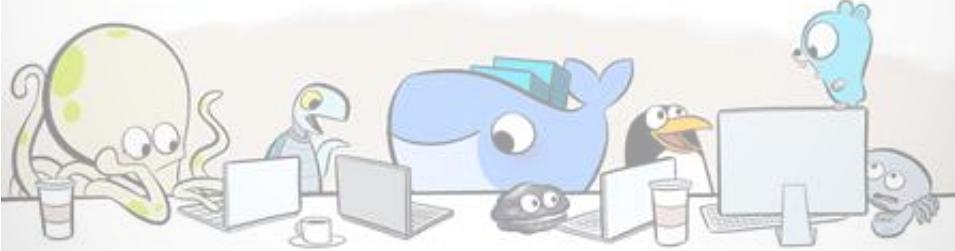
Workshop: DockerFile

- Get Data (Direct/Cache) and Delete Data

```

praparn-MacBook-Pro:~$ curl http://$Server_IP:$Server_Port/users/1
Praparn-Yanicharoen@MacBook-Pro:~$ curl http://$Server_IP:$Server_Port/users/1
Praparn-Yanicharoen@MacBook-Pro:~$ curl http://$Server_IP:$Server_Port/users/1
Praparn-Yanicharoen@MacBook-Pro:~$ curl http://$Server_IP:$Server_Port/users/4
Sakkan-Yanicharoen@Database Direct
praparn-MacBook-Pro:~$ curl http://$Server_IP:$Server_Port/users/4
Sakkan-Yanicharoen@Database Cache
praparn-MacBook-Pro:~$ curl http://$Server_IP:$Server_Port/users/removeuser/1
#####
Record was deleted (Both Database Cache) #####
praparn-MacBook-Pro:~$ curl http://$Server_IP:$Server_Port/users/removeuser/2
#####
Record was deleted #####
praparn-MacBook-Pro:~$ curl http://$Server_IP:$Server_Port/users/removeuser/3
#####
Record was deleted #####
praparn-MacBook-Pro:~$ curl http://$Server_IP:$Server_Port/users/removeuser/4
#####
Record was deleted (Both Database Cache) #####
praparn-MacBook-Pro:~$ 

```



Docker: The Next-Gen of Virtualization 

221

Workshop: DockerFile

- Case for POSTMAN

The first screenshot shows a GET request to `http://192.168.99.103` with a response body containing:

```

<h1> Welcome Page from Container Python Lab </h1>
<h2> Checkpoint Date/Time: Sun Aug 13 14:02:51 2017

```

The second screenshot shows a GET request to `http://192.168.99.103/init` with a response body containing:

```

#####
Database Create New Account Table Done #####

```



Docker: The Next-Gen of Virtualization 

222

Workshop: DockerFile

- Case for POSTMAN

Docker: The Next-Gen of Virtualization 

223

Workshop: DockerFile

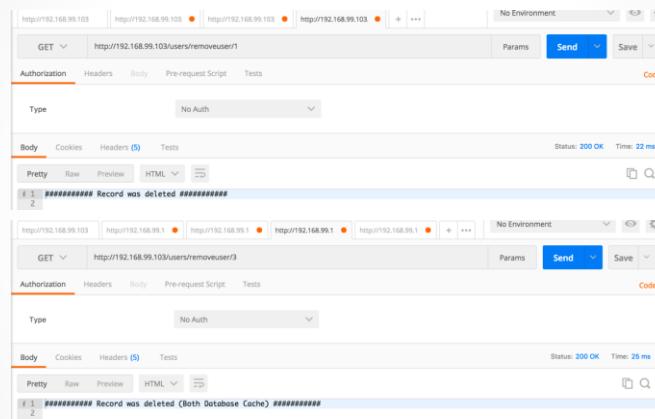
- Case for POSTMAN

Docker: The Next-Gen of Virtualization 

224

Workshop: DockerFile

- Case for POSTMAN



Docker: The Next-Gen of Virtualization



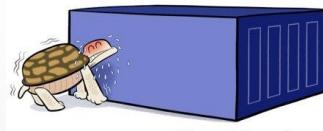
225

Dockerfile

- Multi-State was introduced on April 2017 (Since 17.06)
- Write multiple "FROM" on same dockerfile
- Reduce duplicate workload by copy "intermediate state" during build to new image

Build smaller images with Multi-stage builds

**First stage:
complete build
environment**



One Dockerfile, one build

**Second stage:
minimal runtime
environment** ❤



Docker: The Next-Gen of Virtualization



226

Dockerfile

Multi-state dockerfile:

```

1 ##########
2 # STEP 1 build executable binary
3 #########
4 FROM golang:alpine AS builder
5 # Install git.
6 # Git is required for fetching the dependencies.
7 RUN apk update && apk add --no-cache git
8 WORKDIR $GOPATH/src/mypackage/myapp/
9 COPY . .
10 # Fetch dependencies.
11 # Using go get.
12 RUN go get -d -
13 # Build the binary.
14 RUN go build -o /go/bin/hello
15 #########
16 # STEP 2 build a small image
17 #########
18 FROM scratch
19 # Copy our static executable.
20 COPY --from=builder /go/bin/hello /go/bin/hello
21 # Run the hello binary.
22 ENTRYPOINT ["/go/bin/hello"]

```

Output: Final Image 1 Unit

Docker: The Next-Gen of Virtualization



227

Compose

• • •

Docker: The Next-Gen of Virtualization



228

Compose

- Compose เป็นเครื่องมือที่ใช้ในการสร้าง system service ที่จะใช้ในการรันระบบงานทั้งระบบ ซึ่งตามปกติจะประกอบไปด้วยหลาย Component อาทิเช่น
 - o Web component service
 - o Application component service
 - o Database component service
 - o Load balance component service
 - o etc
- compose สามารถควบคุมการ start / stop / monitor การทำงานของ service ทั้งระบบเป็น single point
- แนะนำสำหรับการสร้าง development environment / test environment / automatic deploy to production (docker-machine / swarm) (build one ➔ ship to everywhere)
- *No Support -ip now

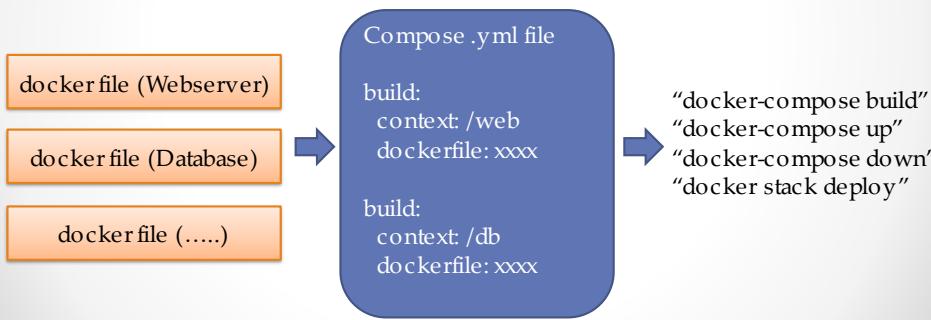
Docker: The Next-Gen of Virtualization



229

Compose

- ขั้นตอนในการสร้าง compose
 - o สร้าง docker file สำหรับแต่ละ component
 - o กำหนดค่า running parameter ใน .yml/.yaml file ของ compose ซึ่งจะอ้างอิงถึง docker file แต่ละตัว
 - o docker-compose up



Docker: The Next-Gen of Virtualization



230

Compose

Compose file format	Docker Engine release
3.7	18.06.0+
3.6	18.02.0+
3.5	17.12.0+
3.4	17.09.0+
3.3	17.06.0+
3.2	17.04.0+
3.1	1.13.1+
3.0	1.13.0+
2.4	17.12.0+
2.3	17.06.0+
2.2	1.13.0+
2.1	1.12.0+
2.0	1.10.0+
1.0	1.9.1+

Docker Compose release notes
Estimated reading time: 75 minutes

1.24.0
 (2019-03-28)

Features

- Added support for connecting to the Docker Engine using the `ssh` protocol.
- Added an `--all` flag to `docker-compose ps` to include stopped one-off containers in the command's output.
- Added bash completion for `ps --all|-a`.
- Added support for `credential_spec`.
- Added `--parallel` to `docker build`'s options in `bash` and `zsh` completion.

Bug Fixes

- Fixed a bug where some valid credential helpers weren't properly handled by Compose when attempting to pull images from private registries.
- Fixed an issue where the output of `docker-compose start` before containers were created was misleading.
- Compose will no longer accept whitespace in variable names sourced from environment files. This matches the Docker CLI behavior.
- Compose will now report a configuration error if a service attempts to declare duplicate mount points in the volumes section.
- Fixed an issue with the containerized version of Compose that prevented users from writing to `stdin` during interactive sessions started by `run` or `exec`.
- One-off containers started by `run` no longer adopt the restart policy of the service, and are instead set to never restart.
- Fixed an issue that caused some container events to not appear in the output of the `docker-compose events` command.
- Missing images will no longer stop the execution of `docker-compose down` commands. A warning is now displayed instead.
- Force `virtualess` version for macOS CI.
- Fixed merging of Compose files when network has `None` config.
- Fixed `CTRL+C` issues by enabling `bootloader_ignore_signals` in `pyinstaller`.
- Bumped `docker-py` version to 3.7.2 to fix SSH and proxy configuration issues.
- Fixed release script and some typos on release documentation.

Docker: The Next-Gen of Virtualization 

231

Compose

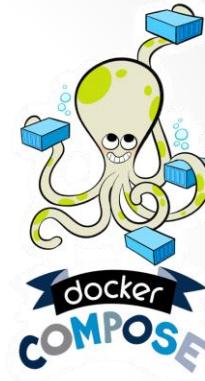
- Compose syntax: (`docker compose up/down/run`)
 - version: x (Current Version 3)
 - services:
 - XXX (service name):
 - image: <image name>
 - build:
 - target: /xxxx (New on Version 3.4)
 - cache_from: (New on Version 3.2)
 - <image name>
 - labels: (New on Version 3.3)
 - <label>; <value>
 - context: ./<path>
 - dockerfile: <file>
 - container_name: <name>
 - args:
 - <xxx>;<value>
 - dns: x.x.x.x
 - dns_search: xxx.com
 - entrypoint: ["nginx","-c","/etc/nginx/nginx.conf"]
 - env_file: xxxx.xxx (VAR=VAL)
 - environment:
 - TOKEN: XXXX
 - ports:
 - "9999:9999"
 - link:
 - xx:<alias>
 - healthcheck:
 - test: ["CMD-SHELL", "pg_isready"]
 - interval: 30s
 - timeout: 30s
 - retries: 3
 - start_period: 100s (New on Version 3.4)

Docker: The Next-Gen of Virtualization 

232

Compose

- Compose syntax:
 - XXX (service name):
 - depends_on:
 - XXXX
 - XXXX
 - networks:
 - XXXX1
 - XXXX2



Ref:<https://docs.docker.com/compose/compose-file/#command>

Docker: The Next-Gen of Virtualization



233

Compose

Control startup and shutdown order in Compose

Estimated reading time: 2 minutes

You can control the order of service startup and shutdown with the `depends_on` option. Compose always starts and stops containers in dependency order, where dependencies are determined by `depends_on`, `links`, `volumes_from`, and `network_mode "service":..."`.

However, for startup Compose does not wait until a container is "ready" (whatever that means for your particular application) - only until it's running. There's a good reason for this.

The problem of waiting for a database (for example) to be ready is really just a subset of a much larger problem of distributed systems. In production, your database could become unavailable or move hosts at any time. Your application needs to be resilient to these types of failures.

To handle this, design your application to attempt to re-establish a connection to the database after a failure. If the application retries the connection, it can eventually connect to the database.

The best solution is to perform this check in your application code, both at startup and whenever a connection is lost for any reason. However, if you don't need this level of resilience, you can work around the problem with a wrapper script:

- Use a tool such as `wait-for-it`, `dockerize`, or sh-compatible `wait-for`. These are small wrapper scripts which you can include in your application's image to poll a given host and port until it's accepting TCP connections.

For example, to use `wait-for-it.sh` or `wait-for` to wrap your service's command:

```
version: "2"
services:
  web:
    build: .
    ports:
      - "80:8000"
    depends_on:
      - "db"
    command: ["./wait-for-it.sh", "db:5432", "--", "python", "app.py"]
  db:
    image: postgres
```

Ref:<https://docs.docker.com/compose/startup-order/>

Docker: The Next-Gen of Virtualization



234

Compose

[eficode / wait-for](https://github.com/Eficode/wait-for)
forked from makako-wait-for

[Code](#) [Pull requests 12](#) [Projects 0](#) [Wiki](#) [Security](#) [Insights](#)

./wait-for is a script to wait for another service to become available.

Branch: master [New pull request](#) [Create new file](#) [Upload files](#) [Find File](#) [Clone or download](#)

This branch is 8 commits ahead of makako-wait-for.

suoonto Merge pull request #7 from szTheory/patch-1 [#16](#) [Compare](#)
Latest commit 8283864 on Apr 9, 2018

- [.gitignore](#) Add testing using bats 2 years ago
- [.travis.yml](#) Move travis build and status under Eficode 2 years ago
- [Dockerfile](#) Add testing using bats 2 years ago
- [LICENSE](#) initial commit 2 years ago
- [README.md](#) Fix also the other url that had capital E last year
- [package.json](#) add name and version so it can be installed with npm 2 years ago
- [wait-for](#) use 'exec "\$@"' 2 years ago
- [wait-for.bats](#) Add testing using bats 2 years ago

[README.md](#)

Wait for another service to become available

./wait-for is a script designed to synchronize services like docker containers. It is sh and alpine compatible. It was inspired by vishnububy/wait-for-it, but the core has been rewritten at Eficode by dsuni and mrako.

When using this tool, you only need to pick the `wait-for`.file as part of your project.

[Build](#) [Issues](#)

Ref: <https://github.com/Eficode/wait-for>

Docker: The Next-Gen of Virtualization 

235

Compose

[vishnubob / wait-for-it](https://github.com/vishnubob/wait-for-it)

[Code](#) [Issues 16](#) [Pull requests 18](#) [Projects 0](#) [Wiki](#) [Security](#) [Insights](#)

Pure bash script to test and wait on the availability of a TCP host and port

Branch: master [New pull request](#) [Create new file](#) [Upload files](#) [Find File](#) [Clone or download](#)

douglas-gibbons Merge branch 'woeffel-master' [#16](#) [Compare](#)
Latest commit 54d1fb on Nov 4, 2018

- [test](#) Fixes to test script for flake8 2 years ago
- [.gitignore](#) Start of test framework 2 years ago
- [.travis.yml](#) Fixes to test script for flake8 2 years ago
- [LICENSE](#) license added 3 years ago
- [README.md](#) README: community section + mention Debian package 10 months ago
- [wait-for-it.sh](#) Merge branch 'master' of https://github.com/woeffel/wait-for-it into master 8 months ago

[README.md](#)

wait-for-it

`wait-for-it.sh` is a pure bash script that will wait on the availability of a host and TCP port. It is useful for synchronizing the spin-up of interdependent services, such as linked docker containers. Since it is a pure bash script, it does not have any external dependencies.

Usage

```
wait-for-it.sh host:port [-s] [-t timeout] [-- command args]
-h HOST | --host=HOST      Host or IP under test
-p PORT | --port=PORT      TCP port under test
--strict                   Alternatively, you specify the host and port as host:port
-s | --strict               Only execute subcommand if the test succeeds
-q | --quiet                Don't output any status messages
-t TIMEOUT | --timeout=TIMEOUT
                           Timeout in seconds, zero for no timeout
-- COMMAND ARGS             Execute command with args after the test finishes
```

Ref: <https://github.com/vishnubob/wait-for-it>

Docker: The Next-Gen of Virtualization 

236

Compose

```

1  from flask import Flask
2  from flask import Response
3  from flask import request
4  from redis import Redis
5  from datetime import datetime
6  import MySQLdb
7  import sys
8  import redis
9  import time
10 import shlib
11 import os
12 import json
13 flagloop= 0
14 app = Flask(__name__)
15 startTime = datetime.now()
16 while flagloop == 0:
17     try:
18         db = MySQLdb.connect("mainsdb","root","password")
19     except:
20         time.sleep(2)
21         continue
22     else:
23         flagloop= 1
24
25 CACHE_DB = Redis(host=os.environ.get('REDIS_HOST'), port=6379)
26 MAIN_DB = db.cursor()
27 @app.route('/')
28 def hello():
29     return '<H1> Welcome Page From Container Python Lab </H1>Checkpoint Date/Time: ' + time.strftime("%c") +'\n'
30 @app.route('/init')
31 def init():
32     MAIN_DB.execute("DROP DATABASE IF EXISTS ACCTABLE")
33     MAIN_DB.execute("CREATE DATABASE ACCTABLE")
34     MAIN_DB.execute("USE ACCTABLE")
35     sql="""
36     CREATE TABLE users (
37         ID INT,
38         USER char(30),
39         DESCRIBE char(250)
40     )"""
41     MAIN_DB.execute(sql)
42     db.commit()
43     return "##### Database Create New Account Table Done #####\n".

```

Docker: The Next-Gen of Virtualization



237

Compose

- Compose syntax: (docker stack deploy)
 - services:
 - XXX (service name):
 - image: <image name>
 - deploy: <SWARM>
 - mode:
 - global
 - replicated
 - resource:
 - limits:
 - cpus: '0.5'
 - memory: 100M
 - reservations:
 - cpus: '0.1'
 - memory: 20M
 - restart_policy:
 - condition: on-failure
 - delay: 5s
 - max_attempts: 3
 - update_config:
 - parallelism: 2
 - delay: 10s
 - failure_action: 10ms
 - max_failure_ratio: X
 - order: {stop-first,start-first} <New on Version 3.4>



Docker: The Next-Gen of Virtualization



238

Compose

- DOCKER STACK DEPLOY
 - Not Supported
 - build
 - cgroup_parent
 - container_name
 - devices
 - dns
 - dns_search
 - external_links
 - links
 - network_mode
 - security_opt
 - stop_signal
 - sysctls
 - userns_mode



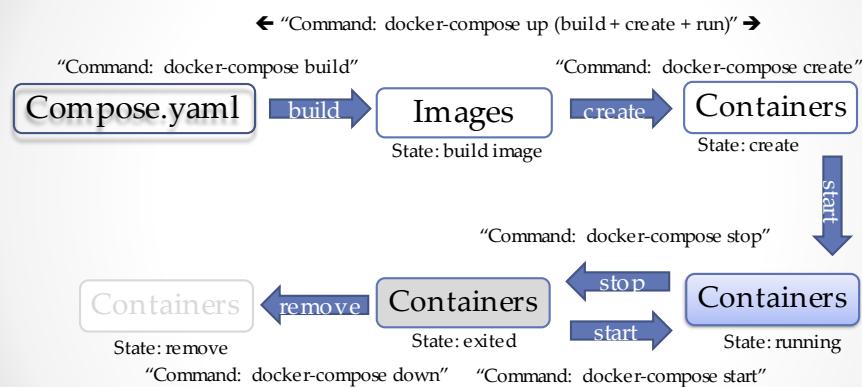
Docker: The Next-Gen of Virtualization



239

Compose

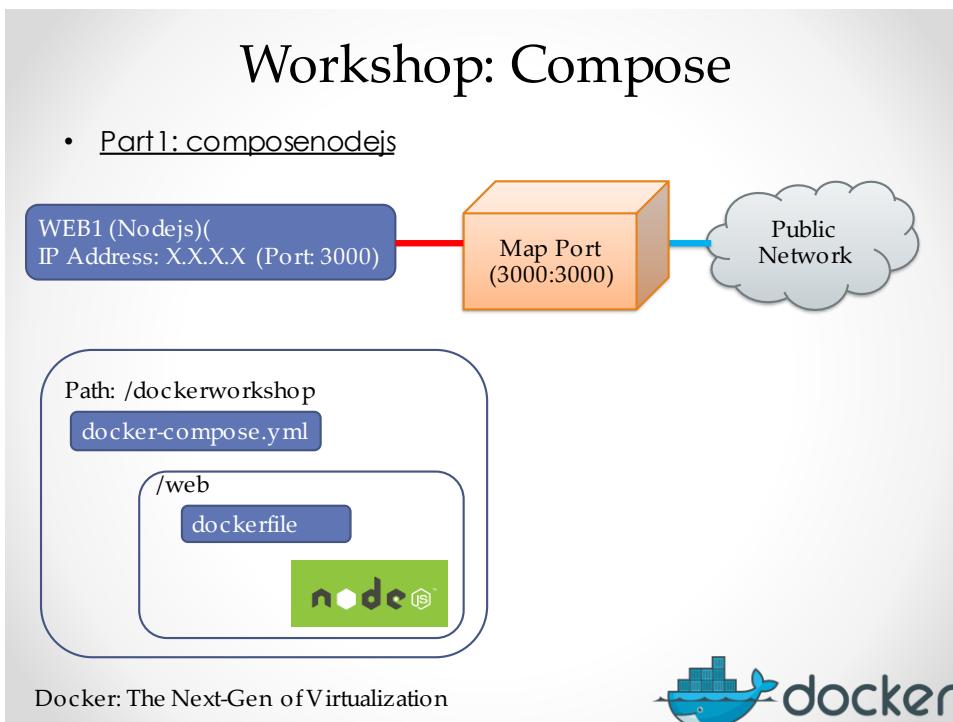
- Compose State:



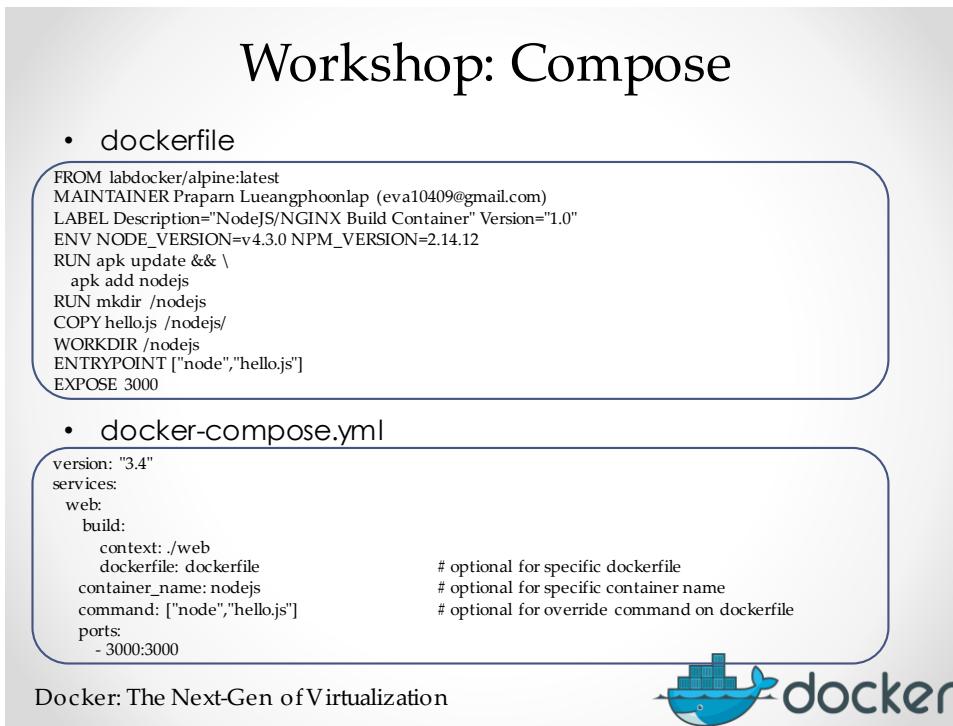
Docker: The Next-Gen of Virtualization



240



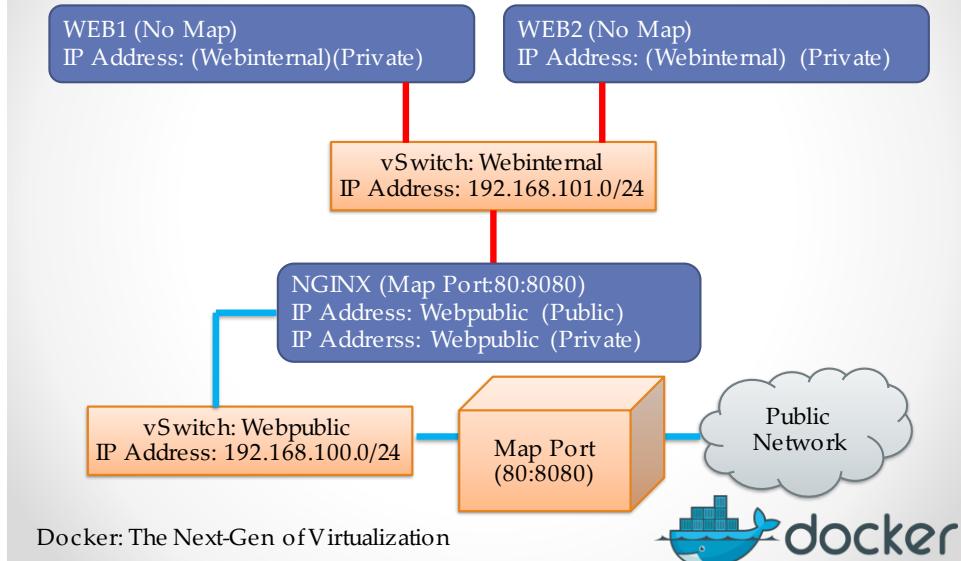
241



242

Workshop: Compose

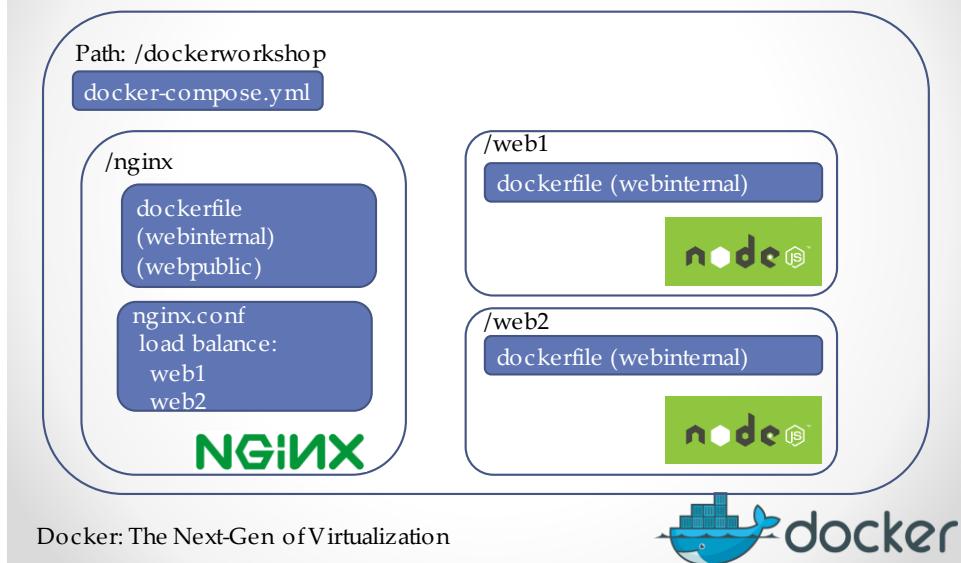
- Part 2: composemultinodejs



243

Workshop: Compose

- Use case: multinodejs with nginx



244

Workshop: Compose

- nginx: dockerfile

```
FROM labdocker/alpine:latest
MAINTAINER Praparn Lueangphoonlap (eva10409@gmail.com)
LABEL Description="NodeJS/NGINX Build Container" Version="1.0"
ENV NODE_VERSION=v10.14.2-r0 NPM_VERSION=v10.14.2-r0
RUN mkdir -p /run/nginx
RUN apk update && \
apk add nginx curl && \
rm /etc/nginx/conf.d/default.conf
COPY nginx.conf /etc/nginx/nginx.conf
COPY labdocker.com.crt /etc/nginx/labdocker.com.crt
COPY labdocker.com.key /etc/nginx/labdocker.com.key
WORKDIR /usr/sbin
ENTRYPOINT ["nginx", "-c", "/etc/nginx/nginx.conf"]
EXPOSE 8080 8443
```

Docker: The Next-Gen of Virtualization



245

Workshop: Compose

- Web1 & Web2: dockerfile

```
FROM labdocker/alpine:latest
MAINTAINER Praparn Lueangphoonlap (eva10409@gmail.com)
LABEL Description="NodeJS/NGINX Build Container" Version="1.0"
ENV NODE_VERSION=v4.3.0 NPM_VERSION=2.14.12
RUN apk update && \
apk add nodejs
RUN mkdir /nodejs
COPY hello.js /nodejs/
WORKDIR /nodejs
ENTRYPOINT ["node","hello.js"]
EXPOSE 3000
```

Docker: The Next-Gen of Virtualization



246

Workshop: Compose

- Web1: Hello.js

```
var http = require('http');
http.createServer(function (req, res)
{res.writeHead(200, {'Content-Type': 'text/plain'});
res.end('Hello World Container Docker for Nodejs Node 1 \n');}).listen(3000,
'0.0.0.0');
console.log('Server running at http://0.0.0.0:3000/');
```

- Web2: Hello.js

```
var http = require('http');
http.createServer(function (req, res)
{res.writeHead(200, {'Content-Type': 'text/plain'});
res.end('Hello World Container Docker for Nodejs Node 2 \n');}).listen(3000,
'0.0.0.0');
console.log('Server running at http://0.0.0.0:3000/');
```

Docker: The Next-Gen of Virtualization



247

Workshop: Compose

- docker-compose.yml

```
1 | version: '3.4'
2 | services:
3 |   nginx:
4 |     build:
5 |       context: ./nginx
6 |       dockerfile: dockerfile      # optional for specific dockerfile
7 |       cache_from:
8 |         - alpine:latest
9 |       container_name: nginx
10 |       depends_on:
11 |         - web1
12 |         - web2
13 |       healthcheck:
14 |         test: ["CMD", "curl", "-f", "http://localhost:80"] # option for health check and send curl for check http://localhost:3000
15 |         interval: 30s          # interval health check
16 |         timeout: 10s          # timeout for check
17 |         retries: 3            # maximum retries
18 |         start_period: 30s    # start period <news on 3.4>
19 |       networks:
20 |         webpublic:
21 |           aliases:
22 |             - nginx
23 |         webinternal:
24 |           aliases:
25 |             - nginx
26 |         ports:
27 |           - "80:8080"
```

Docker: The Next-Gen of Virtualization



248

Workshop: Compose

- docker-compose.yml

```

29   web1:
30     build:
31       context: ./web1
32       dockerfile: dockerfile      # optional for specific dockerfile
33       cache_from:
34         - labdocker/alpineweb:latest
35       container_name: web1
36       healthcheck:
37         test: ["CMD", "curl", "-f", "http://localhost:3000"] # option for health check and send curl for check http://localhost:3000
38         interval: 30s           # interval health check
39         timeout: 10s           # timeout for check
40         retries: 3             # maximum retries
41         start_period: 30s     # start period <news on 3.4>
42     networks:
43       webinternal:
44         aliases:
45           - web1
46
47   web2:
48     build:
49       context: ./web2
50       dockerfile: dockerfile      # optional for specific dockerfile
51       cache_from:
52         - labdocker/alpineweb:latest
53       container_name: web2
54       healthcheck:
55         test: ["CMD", "curl", "-f", "http://localhost:3000"] # option for health check and send curl for check http://localhost:3000
56         interval: 30s           # interval health check
57         timeout: 10s           # timeout for check
58         retries: 3             # maximum retries
59         start_period: 30s     # start period <news on 3.4>
60     networks:
61       webinternal:
62         aliases:
63           - web2

```

Docker: The Next-Gen of Virtualization



249

Workshop: Compose

- docker-compose.yml

```

65   networks:
66     webpublic:
67       driver: bridge
68       ipam:
69         driver: default
70         config:
71           - subnet: 192.168.100.0/24
72     webinternal:
73       driver: bridge
74       ipam:
75         driver: default
76         config:
77           - subnet: 192.168.101.0/24
78

```

Docker: The Next-Gen of Virtualization



250

Docker Security

• • •

Docker: The Next-Gen of Virtualization



251

Docker Security

- เพื่อให้การรันงานบน docker container มีความปลอดภัยสูง docker มีการพิจารณาเรื่อง security ในการทำงานแม่ของก้าเต็งนี้
- Control Group (CG Group) / Name Space
 - Docker container run in separate namespace (Process independence, Not release / touch with other container)
 - Separate network stack
 - CG group will create for separate resource (CPU, Memory, I/O) and limit for protect single container run process and make host fail (denied-of-service)
- Docker daemon operation (root privilege)
 - Be aware for allow user who operate with docker daemon
 - Some operation quite risk for security (docker pull, docker run –v with share host path etc)
 - Docker API should be aware for RESTFUL connection with secure method

Docker: The Next-Gen of Virtualization



252

Docker Security

- Secure enhancement by limit capability of container inside
 - Normally user will be “root” inside container
 - Almost container job (web server, restful server, database server) don’t need high privilege as “root” for operate
 - Control user inside container with limit capability
 - Some activity should be prohibited on container
 - Mount disk operation
 - Access system path (/bin, /usr/bin, /proc etc)
 - Create network socket
 - Execute shell
 - etc

Docker: The Next-Gen of Virtualization



253

Docker Security

- Docker Image Scanning
 - How can you make sure that your image is not vulnerability

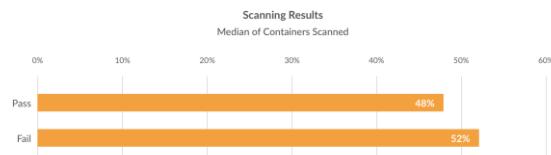
2019 Container Usage Report

Image scanning

Regardless of the source of the container images, it is critical to perform image scanning and identify known vulnerabilities prior to deploying into production. To quantify the scope of the risk of vulnerabilities, we sampled pass and fail rates for images scanned over a five-day period. Over half of the images failed, meaning they were found to have known vulnerabilities with a severity of high or greater.

"We need to check configurations and validate that our images are free of vulnerabilities before pushing to production."

—
Global Travel Company



Ref: <https://docs.anchore.com/current/>

Docker: The Next-Gen of Virtualization



254

Docker Security

- Docker Bench for Security
 - Docker container / script for check best practice to use docker in production (Ref: CIS Test)

[Docker Bench for Security](#)

```

# Docker Bench for Security v1.3.5
# Docker, Inc. (c) 2015-
# Checks for dozen of common best-practices around deploying Docker containers in production.
# Inspired by the CIS Docker Benchmarks v1.2.0.
# ...

Initializing Tue Nov  5 18:27:42 UTC 2019
[INFO] 1 - Host Configuration
[INFO] 1.1 - General Configuration
[INFO] 1.1.1 - Ensure the container host has been Hardened
[INFO] 1.1.2 - Ensure Docker is up to date
[INFO] 1.1.3 - Using 19.4.3+, verify it's up to date as deemed necessary
[INFO] 1.1.4 - Your operating system vendor may provide support and security maintenance for Docker
[INFO] 1.1.5 - Linux Hosts Specific Configuration
[WARN] 1.1.5.1 - Ensure only trusted users are allowed to control Docker daemon
[INFO] 1.1.5.2 - Ensure a separate partition for containers has been created
[INFO] 1.1.5.3 - Ensure auditing is configured for the Docker daemon
[INFO] 1.1.5.4 - Ensure Docker socket file permissions are correct
[INFO] 1.1.5.5 - Ensure auditing is configured for Docker files and directories - /var/lib/docker
[INFO] 1.1.5.6 - Ensure Docker socket file permissions are correct
[INFO] 1.1.5.7 - Ensure auditing is configured for Docker files and directories - /etc/docker
[INFO] 1.1.5.8 - Ensure Docker socket file permissions are correct
[INFO] 1.1.5.9 - Ensure auditing is configured for Docker files and directories - /etc/sysconfig/docker
[INFO] 1.1.5.10 - File not found
[INFO] 1.1.5.11 - Ensure Docker socket file permissions are correct
[INFO] 1.1.5.12 - Ensure auditing is configured for Docker files and directories - /usr/bin/runc
[INFO] 1.1.5.13 - File not found

[INFO] 1.2 - Docker daemon configuration
[INFO] 1.2.1 - Docker daemon configuration is restricted between containers on the default bridge
[INFO] 1.2.2 - Ensure the logging level is set to 'Info'
[INFO] 1.2.3 - Ensure Docker is allowed to make changes to its labels

The Docker Bench for Security is a script that checks for dozens of common best-practices around deploying Docker containers in production. The tests are all automated, and are inspired by the CIS Docker Benchmarks v1.2.0.

```

Initializing Tue Mar 3 12:24:54 UTC 2020

```

[INFO] 1 - Host Configuration
[INFO] 1.1 - General Configuration
[INFO] 1.1.1 - Ensure the container host has been Hardened
[INFO] 1.1.2 - Ensure Docker is up to date
[INFO] 1.1.3 - Using 19.4.3+, verify it's up to date as deemed necessary
[INFO] 1.1.4 - Your operating system vendor may provide support and security maintenance for Docker
[INFO] 1.1.5 - Linux Hosts Specific Configuration
[WARN] 1.1.5.1 - Ensure a separate partition for containers has been created
[INFO] 1.1.5.2 - Ensure only trusted users are allowed to control Docker daemon
[INFO] 1.1.5.3 - Docker socket file permissions are correct
[INFO] 1.1.5.4 - Ensure auditing is configured for the Docker daemon
[INFO] 1.1.5.5 - Ensure Docker socket file permissions are correct
[INFO] 1.1.5.6 - Ensure auditing is configured for Docker files and directories - /var/lib/docker
[INFO] 1.1.5.7 - Ensure Docker socket file permissions are correct
[INFO] 1.1.5.8 - Ensure auditing is configured for Docker files and directories - /etc/docker
[INFO] 1.1.5.9 - Ensure Docker socket file permissions are correct
[INFO] 1.1.5.10 - Ensure auditing is configured for Docker files and directories - /etc/sysconfig/docker
[INFO] 1.1.5.11 - File not found
[INFO] 1.1.5.12 - Ensure Docker socket file permissions are correct
[INFO] 1.1.5.13 - Ensure auditing is configured for Docker files and directories - /etc/default/docker
[INFO] 1.1.5.14 - File not found
[INFO] 1.1.5.15 - Ensure Docker socket file permissions are correct
[INFO] 1.1.5.16 - Ensure auditing is configured for Docker files and directories - /etc/systemconfig/docker
[INFO] 1.1.5.17 - File not found
[INFO] 1.1.5.18 - Ensure auditing is configured for Docker files and directories - /etc/docker/demon.json
[INFO] 1.1.5.19 - Ensure Docker socket file permissions are correct
[INFO] 1.1.5.20 - Ensure auditing is configured for Docker files and directories - /usr/bin/runc
[INFO] 1.1.5.21 - Ensure Docker socket file permissions are correct
[INFO] 1.1.5.22 - Ensure auditing is configured for Docker files and directories - /usr/sbin/runc
[INFO] 1.1.5.23 - File not found

The Docker Bench for Security is a script that checks for dozens of common best-practices around deploying Docker containers in production. The tests are all automated, and are inspired by the CIS Docker Benchmarks v1.2.0.

```

[Ref: https://github.com/docker/docker-bench-security](https://github.com/docker/docker-bench-security)

Docker: The Next-Gen of Virtualization



255

Docker Security

- Anchore for docker image scanning
 - Scan in each layer of docker image and compare with know vulnerability

Anchore Engine PASSED

anchore

For the most up-to-date information on Anchore Engine, Anchore CLI, and other Anchore software, please refer to the [Anchore Documentation](#)

The Anchore Engine is an open source project that provides a centralized service for inspection, analysis and certification of container images. The Anchore engine is provided as a Docker container image that can be run standalone, or within an orchestration platform such as Kubernetes, Docker Swarm, Rancher, Amazon ECS, and other container orchestration platforms.

The Anchore engine can be accessed directly through a RESTful API or via the Anchore CLI.

With a deployment of Anchore Engine running in your environment, container images are downloaded and analyzed from Docker V2 compatible container registries, and then evaluated against user customizable policies to perform security, compliance and best practices enforcement checks. Anchore Engine is appropriate to use stand alone/interactively, as a service integrated with your CI/CD to bring security/compliance/best-practice enforcement to your build pipeline, or as a component integrated into existing container monitoring and control frameworks via integration with its RESTful API.

Anchore Engine is also the OSS foundation for [Anchore Enterprise](#), which adds a graphical UI (providing policy management, user management, a summary dashboard, security and policy evaluation reports, and many other graphical client controls), and other back-end features and modules.

Docker: The Next-Gen of Virtualization



256

Docker Security

- Default apparmor in docker (1.13.0 and above) apply on all docker container

```

root@ubuntu-16-0-1-104:~$ docker container exec -it sectestdefault sh
/ # touch /tmp/testcreatefile
/ # ls -hl /tmp
total 0
drwxr-xr-x 1 root root 0 Mar 3 07:33 testcreatefile
/ # touch /proc/testprohibitfile
touch: /proc/testprohibitfile: No such file or directory
/ # ls -hl /proc
total 0
drwxr-xr-x 9 root root 0 Mar 3 07:33 1
drwxr-xr-x 9 root root 0 Mar 3 07:34 14
drwxr-xr-x 9 root root 0 Mar 3 07:33 6
drwxrwxrwt 2 root root 0 Mar 3 07:33 audit
drwxr-xr-x 1 root root 0 Mar 3 07:33 auditinfo
drwxr-xr-x 4 root root 0 Mar 3 07:33 bus
-rw-r--r-- 1 root root 0 Mar 3 07:34 cgroups
-rw-r--r-- 1 root root 0 Mar 3 07:34 configfs
-rw-r--r-- 1 root root 0 Mar 3 07:34 controllers
-rw-r--r-- 1 root root 0 Mar 3 07:34 cpufreq
-rw-r--r-- 1 root root 0 Mar 3 07:34 cpunfo
-rw-r--r-- 1 root root 0 Mar 3 07:34 crypt
-rw-r--r-- 1 root root 0 Mar 3 07:34 devices
-rw-r--r-- 1 root root 0 Mar 3 07:34 devicemapper
-rw-r--r-- 1 root root 0 Mar 3 07:34 dm
drwxr-xr-x 2 root root 0 Mar 3 07:34 driver
-rw-r--r-- 1 root root 0 Mar 3 07:34 edac
-rw-r--r-- 1 root root 0 Mar 3 07:34 ecryptfs
drwxr-xr-x 6 root root 0 Mar 3 07:33 fs
-rw-r--r-- 1 root root 0 Mar 3 07:34 interrupts
-rw-r--r-- 1 root root 0 Mar 3 07:34 kernel
-rw-r--r-- 1 root root 0 Mar 3 07:34 klocks
drwxr-xr-x 23 root root 0 Mar 3 07:33 irq
-rw-r--r-- 1 root root 0 Mar 3 07:34 key-users
-rw-r--r-- 1 root root 1, 0 Mar 3 07:34 keys
crw-rw-rw- 1 root root 1, 3 Mar 3 07:33 keys
/ # ls -hl /proc/testcreatefile
ls: /proc/testcreatefile: No such file or directory
/ # ls -hl /proc/testprohibitfile
ls: /proc/testprohibitfile: No such file or directory
/ # exit
root@ubuntu-16-0-1-104:~$ docker container stop sectestdefault
sectestdefault

```

Ref: <https://github.com/moby/moby/blob/master/profiles/apparmor/template.go>

Docker: The Next-Gen of Virtualization

257



Docker Security

- Image Content Trust

Key Type	Description
Offline Key	A offline key is used to create tagging keys. Offline keys belong to a person or an organization. Resides client-side. You should store these in a safe place and back them up.
Tagging key	A tagging key is associated with an image repository. Creator with this key can push or pull any tag in the repository. This resides on client-side.
Timestamp Key	A timestamp key is associated with an image repository. This is created by Docker and resides on the server.
Signed tag	A signed tag is a tag that has been signed using Docker's 'tag' or 'push' command.

Docker: The Next-Gen of Virtualization

258

Docker Security

- Recommendation for enhance docker security
 - ระวังการจัดตั้ง User ที่มีหน้าที่ทำงานกับ docker daemon และ privilege คำสั่งบน docker command
 - ใช้ non-root user ใน การจัดตั้ง docker daemon operation (docker run etc)
 - จัดการ Image Content Trust (Sign Image) ใน การสั่ง pull/push image
 - เปลี่ยนการจัดตั้ง docker daemon socket เป็น HTTP socket (with TLS verify and authentication)
 - จำกัดศักยภาพและขอบเขตการทำงานของ user บน container (Secure compute mode (seccomp))

Docker: The Next-Gen of Virtualization



259

Workshop: Docker Security

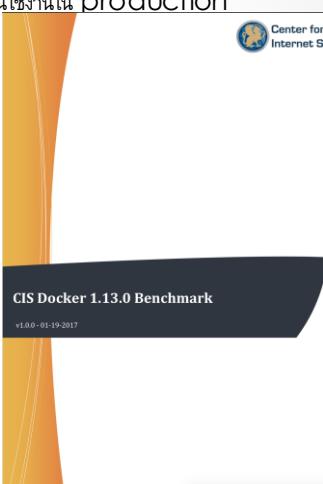
- Docker Bench for Security
- Purpose: ทำการตรวจสอบช่องโหว่ใน image ก่อนใช้งานใน production environment

```
ubuntu@ip-10-23-0-234:/opt/docker-bench-security$ docker run -it --net host --pid host --cap-add audit_control \
--entrypoint /bin/bash -e /opt/docker-bench-security/docker-bench-security \
> /var/lib/docker-bench-security \
> /var/run/docker.sock;/var/run/docker.sock:ro \
> /var/lib/systemd;/var/lib/systemd:ro \
> /var/lib/docker-bench-security/docker-bench-security \
> docker-bench-security
-----
# Docker Bench for Security v1.3.6
# Docker, Inc. (c) 2015-
# Checks for dozens of common best-practices around deploying Docker containers in production.
# Inspired by the CIS Docker Benchmark v1.2.0.

Initializing Tue Mar  3 12:24:34 UTC 2020

[INFO] 1 - Host Configuration
[INFO] 1.1 - General Configuration
[NOTE] 1.1.1 - Ensure the container host has been Hardened
[INFO] 1.1.2 - Ensure the container host is up to date
[INFO] 1.1.3 - Using 19.07.4, verify if it is up to date as deemed necessary
[INFO] 1.1.4 - Your operating system vendor may provide support and security maintenance for Docker
[INFO] 1.2 - Linux Hosts Specific Configuration
[INFO] 1.2.1 - Ensure a separate partition for containers has been created
[INFO] 1.2.2 - Ensure root users are allowed to control Docker daemon
[INFO] 1.2.3 - Ensure auditing is configured for the Docker daemon
[INFO] 1.2.4 - Ensure auditing is configured for Docker files and directories - /var/lib/docker
[INFO] 1.2.5 - Ensure auditing is configured for Docker files and directories - /etc/docker
[INFO] 1.2.6 - Ensure auditing is configured for Docker files and directories - docker.service
[INFO] 1.2.7 - Ensure auditing is configured for Docker files and directories - docker.socket
[INFO] 1.2.8 - File not found
[INFO] 1.2.9 - Ensure auditing is configured for Docker files and directories - /etc/default/docker
[INFO] 1.2.10 - Ensure auditing is configured for Docker files and directories - /etc/sysconfig/docker
[INFO] 1.2.11 - File not found
[INFO] 1.2.12 - Ensure auditing is configured for Docker files and directories - /usr/bin/runc
[INFO] 1.2.13 - File not found
```

Center for Internet Security



Docker: The Next-Gen of Virtualization



260

Workshop: Docker Security

- Docker Bench for Security
- Purpose: ทำการตรวจสอบของไฟล์ใน image ก่อนใช้งานใน production environment

```
ubuntu@ip-10-0-1-254:~/Docker-Bench-for-Security$ docker run -it --net host --pid host --cap-add audit_control \
> -e DOCKER_CONTENT_TRUST \
> -v /var/lib/docker \
> -v /var/run/docker.sock:/var/run/docker.sock:ro \
> -v /usr/lib/systemd:/usr/lib/systemd:ro \
> -v /etc/docker/docker_bench_security \
> docker/bench-security

Docker Bench for Security v1.9.5
# Docker, Inc. (c) 2015-
# Checks for dozens of common best-practices around deploying Docker containers in production.
# Inspired by the CIS Docker Benchmark v1.2.6.

Initializing Tue Mar 3 12:24:34 UTC 2020

[INFO] 1 - Host Configuration

[INFO] 1.1 - General Configuration
[NOTE] 1.1.1 - Ensure the container host has been Hardened
[INFO] 1.1.1.1 - Using 19.03.6, verify it is up to date as deemed necessary
[INFO] 1.1.1.2 - Your operating system vendor may provide support and security maintenance for Docker

[INFO] 1.2 - Linux Hosts Specific Configuration
[INFO] 1.2.1 - Ensure a separate partition for containers has been created
[INFO] 1.2.1.1 - Docker daemon and Docker files are allowed to control Docker daemon
[INFO] 1.2.1.2 - Docker daemon runs with root privileges
[INFO] 1.2.2 - Ensure auditing is configured for the Docker daemon
[INFO] 1.2.2.1 - Docker daemon runs with root privileges
[INFO] 1.2.2.2 - Ensure auditing is configured for Docker files and directories - /var/lib/docker
[INFO] 1.2.2.3 - Ensure auditing is configured for Docker files and directories - /etc/docker
[INFO] 1.2.2.4 - Ensure auditing is configured for Docker files and directories - /etc/docker/service
[INFO] 1.2.2.5 - Ensure auditing is configured for Docker files and directories - /etc/docker/socket
[INFO] 1.2.2.6 - Ensure auditing is configured for Docker files and directories - /etc/default/docker
[INFO] 1.2.2.7 - Ensure auditing is configured for Docker files and directories - /etc/sysconfig/docker
[INFO] 1.2.2.8 - Ensure auditing is configured for Docker files and directories - /etc/docker/demon.json
[INFO] 1.2.2.9 - Ensure auditing is configured for Docker files and directories - /var/bin/containedr
[INFO] 1.2.2.10 - File not found
[INFO] 1.2.2.11 - Ensure auditing is configured for Docker files and directories - /usr/sbin/runc
[INFO] 1.2.2.12 - File not found

[INFO] 1.2.3 - Ensure auditing is configured for Docker files and directories - /usr/sbin/runc
[INFO] 1.2.4 - File not found
```

1.1 Ensure a separate partition for containers has been created (Scored)

Profile Applicability:

- Level 1 - Linux Host OS

Description:

All Docker containers and their data and metadata is stored under `/var/lib/docker` directory. By default, `/var/lib/docker` would be mounted under `/` or `/var` partitions based on availability.

Rationale:

Docker depends on `/var/lib/docker` as the default directory where all Docker related files, including the images, are stored. This directory might fill up fast and soon Docker and the host could become unusable. So, it is advisable to create a separate partition (logical volume) for storing Docker files.

Audit:

At the Docker host execute the below command:

```
grep '/var/lib/docker' /etc/fstab
```

This should return the partition details for `/var/lib/docker` mount point.

Remediation:

For new installations, create a separate partition for `/var/lib/docker` mount point. For systems that were previously installed, use the Logical Volume Manager (LVM) to create partitions.

Docker: The Next-Gen of Virtualization



261

Workshop: Docker Security

- Docker Bench for Security
- Purpose: ทำการตรวจสอบของไฟล์ใน image ก่อนใช้งานใน production environment

```
ubuntu@ip-10-0-1-254:~/Docker-Bench-for-Security$ docker run -it --net host --pid host --cap-add audit_control \
> -e DOCKER_CONTENT_TRUST \
> -v /var/lib/docker \
> -v /var/run/docker.sock:/var/run/docker.sock:ro \
> -v /usr/lib/systemd:/usr/lib/systemd:ro \
> -v /etc/docker/docker_bench_security \
> docker/bench-security

Docker Bench for Security v1.9.5
# Docker, Inc. (c) 2015-
# Checks for dozens of common best-practices around deploying Docker containers in production.
# Inspired by the CIS Docker Benchmark v1.2.6.

Initializing Tue Mar 3 12:24:34 UTC 2020

[INFO] 1 - Host Configuration

[INFO] 1.1 - General Configuration
[NOTE] 1.1.1 - Ensure the container host has been Hardened
[INFO] 1.1.1.1 - Using 19.03.6, verify it is up to date as deemed necessary
[INFO] 1.1.1.2 - Your operating system vendor may provide support and security maintenance for Docker

[INFO] 1.2 - Linux Hosts Specific Configuration
[INFO] 1.2.1 - Ensure a separate partition for containers has been created
[INFO] 1.2.1.1 - Docker daemon and Docker files are allowed to control Docker daemon
[INFO] 1.2.1.2 - Docker daemon runs with root privileges
[INFO] 1.2.2 - Ensure auditing is configured for the Docker daemon
[INFO] 1.2.2.1 - Docker daemon runs with root privileges
[INFO] 1.2.2.2 - Ensure auditing is configured for Docker files and directories - /var/lib/docker
[INFO] 1.2.2.3 - Ensure auditing is configured for Docker files and directories - /etc/docker
[INFO] 1.2.2.4 - Ensure auditing is configured for Docker files and directories - /etc/docker/service
[INFO] 1.2.2.5 - Ensure auditing is configured for Docker files and directories - /etc/docker/socket
[INFO] 1.2.2.6 - Ensure auditing is configured for Docker files and directories - /etc/default/docker
[INFO] 1.2.2.7 - Ensure auditing is configured for Docker files and directories - /etc/sysconfig/docker
[INFO] 1.2.2.8 - Ensure auditing is configured for Docker files and directories - /etc/docker/demon.json
[INFO] 1.2.2.9 - Ensure auditing is configured for Docker files and directories - /var/bin/containedr
[INFO] 1.2.2.10 - File not found
[INFO] 1.2.2.11 - Ensure auditing is configured for Docker files and directories - /usr/sbin/runc
[INFO] 1.2.2.12 - File not found

[INFO] 1.2.3 - Ensure auditing is configured for Docker files and directories - /usr/sbin/runc
[INFO] 1.2.4 - File not found
```

1.6 Ensure auditing is configured for Docker files and directories - /var/lib/docker (Scored)

Profile Applicability:

- Level 1 - Linux Host OS

Description:

Audit `/var/lib/docker`.

Rationale:

Apart from auditing your regular Linux file system and system calls, audit all Docker related files and directories. Docker daemon runs with `root` privileges. Its behavior depends on some key files and directories. `/var/lib/docker` is one such directory. It holds all the information about containers. It must be audited.

Audit:

Verify that there is an audit rule corresponding to `/var/lib/docker` directory.

For example, execute below command:

```
auditctl -l | grep '/var/lib/docker'
```

This should list a rule for `/var/lib/docker` directory.

Remediation:

Add a rule for `/var/lib/docker` directory.

For example,

Add the line as below in `/etc/audit/audit.rules`:

```
-w /var/lib/docker -k docker
```

Then, restart the audit daemon. For example,

```
service auditd restart
```

Docker: The Next-Gen of Virtualization



262

Workshop: Docker Security

- Docker Bench for Security
- Purpose: ทำการตรวจสอบของไฟล์ใน image ก่อนใช้งานใน production environment

```
ubuntu@ip-10-0-1-234:~/Docker-Bench-for-Security$ docker run -it --net host --pid host --cap-add audit_control \
-e DOCKER_CONTENT_TRUST=$DOCKER_CONTENT_TRUST \
-v /var/lib/systemd:/lib/systemd \
-v /var/run/docker.sock:/var/run/docker.sock:ro \
-v /usr/lib/systemd:/usr/lib/systemd:ro \
-v /etc/docker/containers:/var/lib/docker_bench_security \
-d docker-bench-security

# Docker Bench for Security v1.3.5
# Docker, Inc. (c) 2015-
# Checks for dozens of common best-practices around deploying Docker containers in production.
# Inspired by the CIS Docker Benchmark v1.2.6.

Initializing Tue Mar 3 12:24:34 UTC 2020

[INFO] 1 - Host Configuration

[INFO] 1.1 - General Configuration
[NOTE] 1.1.1 - Ensure the container host has been Hardened
[INFO] 1.1.2 - Ensure Docker is up to date
[INFO] 1.1.3 - * Using 19.03.6, verify it is up to date as deemed necessary
[INFO] 1.1.4 - * Your operating system vendor may provide support and security maintenance for Docker

[INFO] 1.2 - Linux Hosts Specific Configuration
[WARN] 1.2.1 - Ensure a separate partition for containers has been created
[INFO] 1.2.2 - * Ensure only trusted users are allowed to control Docker daemon
[dockerx:1001:ubuntu,1001]
[WARN] 1.2.3 - Ensure auditing is configured for the Docker daemon
[PASS] 1.2.4 - Ensure auditing is configured for Docker files and directories - /var/lib/docker
[INFO] 1.2.5 - Ensure auditing is configured for Docker files and directories - /etc/docker
[INFO] 1.2.6 - Ensure auditing is configured for Docker files and directories - docker.service
[INFO] 1.2.7 - * File not found
[INFO] 1.2.8 - Ensure auditing is configured for Docker files and directories - docker.socket
[INFO] 1.2.9 - Ensure auditing is configured for Docker files and directories - /etc/sysconfig/docker
[INFO] 1.2.10 - Ensure auditing is configured for Docker files and directories - /etc/docker/daemon.json
[INFO] 1.2.11 - Ensure auditing is configured for Docker files and directories - /usr/bin/containerd
[INFO] 1.2.12 - Ensure auditing is configured for Docker files and directories - /usr/sbin/runc
[INFO] 1.2.13 - * File not found

ubuntu@ip-10-0-1-234:~$ sudo more /etc/audit/rules.d/audit.rules
## First rule - delete all
-D

## Increase the buffers to survive stress events.
## This make this bigger for busy systems
-b 8192

## This determine how long to wait in burst of events
--backlog_wait_time 0

## Set failure mode to syslog
-f 1

-w /usr/bin/docker -p wa
-w /var/lib/docker -p wa
-w /etc/docker -p wa
-w /lib/systemd/system/docker.service -p wa
-w /lib/systemd/system/docker.socket -p wa
-w /etc/default/docker -p wa
-w /etc/docker/daemon.json -p wa
-w /usr/bin/docker-containerd -p wa
-w /usr/bin/docker-runc -p wa
-w /usr/bin/docker -k docker

ubuntu@ip-10-0-1-234:~$
```

Docker: The Next-Gen of Virtualization



263

Workshop: Docker Security

- Docker Bench for Security
- Purpose: ทำการตรวจสอบของไฟล์ใน image ก่อนใช้งานใน production environment

```
ubuntu@ip-10-0-1-234:~$ docker run -it --net host --pid host --cap-add audit_control \
-e DOCKER_CONTENT_TRUST=$DOCKER_CONTENT_TRUST \
-v /var/lib/systemd:/lib/systemd \
-v /var/run/docker.sock:/var/run/docker.sock:ro \
-v /usr/lib/systemd:/usr/lib/systemd:ro \
-v /etc/docker/containers:/var/lib/docker_bench_security \
-d docker-bench-security

# Docker Bench for Security v1.3.5
# Docker, Inc. (c) 2015-
# Checks for dozens of common best-practices around deploying Docker containers in production.
# Inspired by the CIS Docker Benchmark v1.2.6.

Initializing Tue Mar 3 14:42:09 UTC 2020

[INFO] 1 - Host Configuration

[INFO] 1.1 - General Configuration
[NOTE] 1.1.1 - Ensure the container host has been Hardened
[INFO] 1.1.2 - Ensure Docker is up to date
[INFO] 1.1.3 - * Using 19.03.6, verify it is up to date as deemed necessary
[INFO] 1.1.4 - * Your operating system vendor may provide support and security maintenance for Docker

[INFO] 1.2 - Linux Hosts Configuration
[WARN] 1.2.1 - Ensure a separate partition for containers has been created
[INFO] 1.2.2 - Ensure only trusted users are allowed to control Docker daemon
[dockerx:1001:ubuntu,1001]
[WARN] 1.2.3 - Ensure auditing is configured for the Docker daemon
[PASS] 1.2.4 - Ensure auditing is configured for Docker files and directories - /var/lib/docker
[INFO] 1.2.5 - Ensure auditing is configured for Docker files and directories - /etc/docker
[INFO] 1.2.6 - Ensure auditing is configured for Docker files and directories - docker.service
[INFO] 1.2.7 - * File not found
[INFO] 1.2.8 - Ensure auditing is configured for Docker files and directories - docker.socket
[INFO] 1.2.9 - Ensure auditing is configured for Docker files and directories - /etc/sysconfig/docker
[INFO] 1.2.10 - Ensure auditing is configured for Docker files and directories - /etc/docker/daemon.json
[INFO] 1.2.11 - Ensure auditing is configured for Docker files and directories - /usr/bin/containerd
[INFO] 1.2.12 - Ensure auditing is configured for Docker files and directories - /usr/sbin/runc
[INFO] 1.2.13 - * File not found
```

Docker: The Next-Gen of Virtualization



264

Workshop: Docker Security

- Image Scan with Anchore
- Purpose: ทำการตรวจสอบช่องโหว่ใน image ก่อนใช้งานใน production environment

The screenshot shows the Anchore Container Analysis documentation page. At the top, there's a search bar and a navigation bar with links for 'Docs' (version 2.2), 'Enterprise Quick Start', 'Open-Source Quick Start', 'View FAQ', and 'Start Reading'. Below the header, the title 'Anchore Container Analysis' is displayed, followed by a sub-header 'Welcome to the Anchore Docs! Ready to get started?'. There are four buttons at the bottom of this section: 'Enterprise Quick Start', 'Open-Source Quick Start', 'View FAQ', and 'Start Reading'.

This screenshot shows the 'Integrations' section of the Anchore documentation. It features three main integration points: 'Kubernetes' (with a Helm icon), 'Jenkins' (with a Jenkins icon), and 'CircleCI' (with a CircleCI icon). Each section provides a brief description of how to integrate Anchore into that platform.

Docker: The Next-Gen of Virtualization

265

Workshop: Docker Security

- Image Scan with Anchore
- Purpose: ทำการตรวจสอบช่องโหว่ใน image ก่อนใช้งานใน production environment

```
ubuntu@ip-10-0-1-234:~/anchore-volume$ anchore-cli image add kong:1.5.1-alpine
Image Digest: sha256:7ed3b0acae0d60484662c1b11d239f908ff709cc77fa4
Parent Digest: sha256:c205c779547fc02a0e4ec4ca3c5c9a7d7474f93194e44aaef02a7ff09f8
Analysis Status: not_analyzed
Image Type: docker
Analyzed At: None
Image ID: f606047631c199e02d38967fec2f2abbd4ea8f7221f5cf70845509f9b168
Spec Mode: None
Distro: None
Distro Version: None
Size: 1000000000
Architecture: None
Layer Count: None

Full Tag: docker.io/kong:1.5.1-alpine
Tag Detected At: 2020-03-03T15:55:47Z

ubuntu@ip-10-0-1-234:~/anchore-volume$ anchore-cli image add bitnami/postgresql:latest
Image Digest: sha256:3ee3d50aeb7a1286195c77080ff864f6b79fd7bc4e5aee87c2f1ac93bc9d
Parent Digest: sha256:3ee3d50aeb7a1286195c77080ff864f6b79fd7bc4e5aee87c2f1ac93bc9d
Analysis Status: not_analyzed
Image Type: docker
Analyzed At: 2020-03-03T15:55:28Z
Image ID: 903a0a070b72e22c2926f76ddde150base9445445169f919dccc778a769
Dockerfile Mode: Guesses
Distro: debian
Size: 204668800
Architecture: amd64
Layer Count: 10

Full Tag: docker.io/bitnami/postgresql:latest
Tag Detected At: 2020-03-03T15:55:18Z

ubuntu@ip-10-0-1-234:~/anchore-volume$ anchore-cli image add labdoker/alpine:latest
Image Digest: sha256:b1f184a6e5676399e8686cd402e9ff72780f141768bcb3f79dcce19ff9f16decc9
Parent Digest: sha256:c3f104a6e5676399e8686cd402e9ff72780f141768bcb3f79dcce19ff9f16decc9
Analysis Status: not_analyzed
Image Type: docker
Analyzed At: None
Image ID: 05593d3d928576da37aa9bc44d975bc5f120208da1d80c0079andf03d766ea1
Dockerfile Mode: None
Distro: None
Distro Version: None
Size: 1000000000
Architecture: None
Layer Count: None

Full Tag: docker.io/labdoker/alpine:latest
Tag Detected At: 2020-03-03T15:55:04Z
```

Docker: The Next-Gen of Virtualization

266

Workshop: Docker Security

- Image Scan with Anchore
- Purpose: ทำการตรวจสอบช่องโหว่บน image ก่อนใช้งานใน production environment

```

ubuntu@ip-10-0-1-234:~/anchore-volume$ anchore-cli image list
Full Tag                               Image Digest
docker.io/bitnami/postgresql:latest    sha256:3eed3d580ab7a128619e77d0b9f064f6b70fd7bd7b2c4e5aae875c2f51acc93bc9d
docker.io/kong:1.5.1-alpine             sha256:e80c29afff7eb839ecac6c8d04481e6652ca1b31d12539f038f7f09cc7f94a
docker.io/labdockerdocker/alpine:latest sha256:f1f68446e3676389ec861c692e97f7b03f1a178a5bc3f79dc198ff1f168cc9
docker.io/labdockerdocker/alpine:web:latest sha256:a7aa7c078120ed126487aecbf49d0b1054fffff13185851dc91137c9accb04333
[ubuntu@ip-10-0-1-234:~/anchore-volume$ anchore-cli image list
Full Tag                               Image Digest
docker.io/bitnami/postgresql:latest    sha256:3eed3d580ab7a128619e77d0b9f064f6b70fd7bd7b2c4e5aae875c2f51acc93bc9d
docker.io/kong:1.5.1-alpine             sha256:e80c29afff7eb839ecac6c8d04481e6652ca1b31d12539f038f7f09cc7f94a
docker.io/labdockerdocker/alpine:latest sha256:f1f68446e3676389ec861c692e97f7b03f1a178a5bc3f79dc198ff1f168cc9
docker.io/labdockerdocker/alpine:web:latest sha256:a7aa7c078120ed126487aecbf49d0b1054fffff13185851dc91137c9accb04333
ubuntu@ip-10-0-1-234:~/anchore-volume$ 

```



```

ubuntu@ip-10-0-1-234:~/anchore-volume$ anchore-cli image content labdockerdocker/alpine:latest
Package        Version   License
alpine-baselayout  3.1.0    GPL-2.0
alpine-keys     2.1      MIT
apk-tools       2.28.3   GPL-2
busybox         1.29.3   GPL-2.0
ca-certificates-cacert 201901088  MPI-2.0 GPL-2.0-or-later
libc-utils      0.7.1    BSD
libcrypto1.1    1.1.1b   OpenSSL
libs11.1        1.1.1b   OpenSSL
libtls-standalone 2.7.4    ISC
musl            1.1.20   MIT
xz-utils         5.2.4.1  MIT BSD GPLv2+
scandef          1.2.3    GPL-2.0
ssl_client       1.29.3   GPL-2.0
zlib            1.2.11   ZLIB
ubuntu@ip-10-0-1-234:~/anchore-volume$ 

```

Docker: The Next-Gen of Virtualization



267

Workshop: Docker Security

- Image Scan with Anchore
- Purpose: ทำการตรวจสอบช่องโหว่บน image ก่อนใช้งานใน production environment

```

ubuntu@ip-10-0-1-234:~/anchore-volume$ anchore-cli image vuln kong:1.5.1-alpine all
ubuntu@ip-10-0-1-234:~/anchore-volume$ anchore-cli image vuln bitnami/postgresql:latest all
Vulnerability ID      Package              Severity       Fix      CVE Refs               Vulnerability URL
CVE-2004-971          libbz2-0.9.3       Negligible   None   CVE-2004-971 https://security-tracker.debian.org/tracker/CVE-2004-971
CVE-2004-971          libbz2-0.9.3+os-1.17-3   Negligible   None   CVE-2004-971 https://security-tracker.debian.org/tracker/CVE-2004-971
CVE-2004-971          libbz2-0.9.3-1.17-3     Negligible   None   CVE-2004-971 https://security-tracker.debian.org/tracker/CVE-2004-971
CVE-2004-971          libberbis-0.3-1.17-3     Negligible   None   CVE-2004-971 https://security-tracker.debian.org/tracker/CVE-2004-971
CVE-2004-971          libberbis-0.3-1.17-3+os    Negligible   None   CVE-2004-971 https://security-tracker.debian.org/tracker/CVE-2004-971
CVE-2006-2541          tar-1.38w+rgen2       Negligible   None   CVE-2006-2541 https://security-tracker.debian.org/tracker/CVE-2006-2541
CVE-2007-5686          login-1.4..5..1..1    Negligible   None   CVE-2007-5686 https://security-tracker.debian.org/tracker/CVE-2007-5686
CVE-2007-5686          passwd-1.4..5..1..1    Negligible   None   CVE-2007-5686 https://security-tracker.debian.org/tracker/CVE-2007-5686
CVE-2007-6785          libbz2-1.1..1.15..0+deb10u2   Negligible   None   CVE-2007-6785 https://security-tracker.debian.org/tracker/CVE-2007-6785
CVE-2007-6785          openessl-1.1.1d..0+deb10u2   Negligible   None   CVE-2007-6785 https://security-tracker.debian.org/tracker/CVE-2007-6785
CVE-2010-9228          libs11-1.1.1d..0+deb10u2   Negligible   None   CVE-2010-9228 https://security-tracker.debian.org/tracker/CVE-2010-9228
CVE-2010-9228          openssl-1.1.1d..0+deb10u2   Negligible   None   CVE-2010-9228 https://security-tracker.debian.org/tracker/CVE-2010-9228
CVE-2010-4651          libbz2-1.10..1.28..10    Negligible   None   CVE-2010-4651 https://security-tracker.debian.org/tracker/CVE-2010-4651
CVE-2010-4651          libbz2-1.10..1.28..10+os     Negligible   None   CVE-2010-4651 https://security-tracker.debian.org/tracker/CVE-2010-4651
CVE-2010-4651          libbz2-1.28..2.28..10    Negligible   None   CVE-2010-4651 https://security-tracker.debian.org/tracker/CVE-2010-4651
CVE-2010-4651          locales-1.12..1.28..10    Negligible   None   CVE-2010-4651 https://security-tracker.debian.org/tracker/CVE-2010-4651
CVE-2010-4651          locales-1.12..1.28..10+os  Negligible   None   CVE-2010-4651 https://security-tracker.debian.org/tracker/CVE-2010-4651
CVE-2010-4652          libbz2-1.10..1.28..10    Negligible   None   CVE-2010-4652 https://security-tracker.debian.org/tracker/CVE-2010-4652
CVE-2010-4652          libbz2-1.10..1.28..10+os  Negligible   None   CVE-2010-4652 https://security-tracker.debian.org/tracker/CVE-2010-4652
CVE-2010-4652          libbz2-2.28..2.28..10    Negligible   None   CVE-2010-4652 https://security-tracker.debian.org/tracker/CVE-2010-4652
CVE-2010-4652          libbz2-2.28..2.28..10+os  Negligible   None   CVE-2010-4652 https://security-tracker.debian.org/tracker/CVE-2010-4652
CVE-2010-4653          locales-2.18..2.28..10  Negligible   None   CVE-2010-4653 https://security-tracker.debian.org/tracker/CVE-2010-4653
CVE-2010-4653          locales-2.18..2.28..10+os Negligible   None   CVE-2010-4653 https://security-tracker.debian.org/tracker/CVE-2010-4653
CVE-2010-4654          locales-2..2.28..10     Negligible   None   CVE-2010-4654 https://security-tracker.debian.org/tracker/CVE-2010-4654
CVE-2010-4654          locales-2..2.28..10+os   Negligible   None   CVE-2010-4654 https://security-tracker.debian.org/tracker/CVE-2010-4654
CVE-2010-4655          locales-2..2.28..10     Negligible   None   CVE-2010-4655 https://security-tracker.debian.org/tracker/CVE-2010-4655
CVE-2010-4655          locales-2..2.28..10+os   Negligible   None   CVE-2010-4655 https://security-tracker.debian.org/tracker/CVE-2010-4655
CVE-2010-4656          locales-2..2.28..10     Negligible   None   CVE-2010-4656 https://security-tracker.debian.org/tracker/CVE-2010-4656
CVE-2010-4656          locales-2..2.28..10+os   Negligible   None   CVE-2010-4656 https://security-tracker.debian.org/tracker/CVE-2010-4656
CVE-2011-3374          apt-1.8..2             Negligible   None   CVE-2011-3374 https://security-tracker.debian.org/tracker/CVE-2011-3374
CVE-2011-3374          libpopt-pkg5-1.1..2..2    Negligible   None   CVE-2011-3374 https://security-tracker.debian.org/tracker/CVE-2011-3374
CVE-2011-3374          libpopt-pkg5-1.1..2..2+os  Negligible   None   CVE-2011-3374 https://security-tracker.debian.org/tracker/CVE-2011-3374
CVE-2011-4116          perl-sys-6.28..1..0     Negligible   None   CVE-2011-4116 https://security-tracker.debian.org/tracker/CVE-2011-4116
CVE-2012-4235          login-1.4..5..1..1    Negligible   None   CVE-2012-4235 https://security-tracker.debian.org/tracker/CVE-2012-4235
CVE-2013-4235          passwd-1..6..1..1     Negligible   None   CVE-2013-4235 https://security-tracker.debian.org/tracker/CVE-2013-4235
CVE-2013-4392          libsystemd-241-7..0+deb10u3 Negligible   None   CVE-2013-4392 https://security-tracker.debian.org/tracker/CVE-2013-4392

```

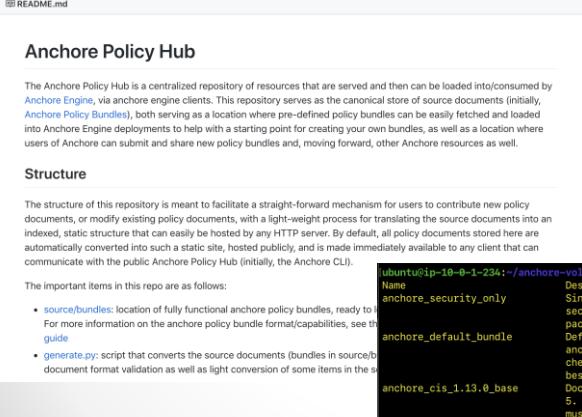
Docker: The Next-Gen of Virtualization



268

Workshop: Docker Security

- Image Scan with Anchore
- Purpose: ทำการตรวจสอบของใน image ก่อนใช้งานใน production environment



```
ubuntu@ip-10-0-1-234:~/anchore-volume$ anchore-cli policy hub list
  Name          Description
  anchor_security_only  Single policy, simple whitelist bundle for performing security checks, including example blacklist known malicious packages by name.
  anchor_default_bundle  Default policy bundle that comes installed with vanilla anchore-engine deployments. Mixture of light vulnerability checks, dockerfiles checks, and warning triggers for common best practices.
  anchor_cis_1.13.0_base  Docker CIS 1.13.0 image content checks, from section 4 and 5. NOTE: some parameters (generally are named 'example...') must be modified as they require site-specific settings
```

Docker: The Next-Gen of Virtualization 

269

Workshop: Docker Security

- Apparmor
- Purpose: จำกัดการทำงานบางอย่างบน container ที่ไม่จำเป็นเพื่อลดความเสี่ยงในการใช้งานเกินขอบเขตของ application

```
ubuntu@ip-10-0-1-104:~$ docker container run -dt --name sectestrestrict --security-opt "apparmor=restrict-apparmor" labdockers/alpine:latest sh
6d621be7b91b3f15a09e9bc1a0e1a9373cc9b8740a017ff8fb163283081b
ubuntu@ip-10-0-1-104:~$ docker container exec -it sectestrestrict sh
Error: No such container: sectestdefault
ubuntu@ip-10-0-1-104:~$ docker container exec -it sectestrestrict sh
// # ping 1.1.1.1
PING 1.1.1.1 (1.1.1.1) 56 data bytes
ping: can't create raw socket: Permission denied
// # apk update
ERROR: Unable to lock database: Permission denied
ERROR: Failed to open apk database: Permission denied
// # apk add curl
ERROR: Unable to lock database: Permission denied
ERROR: Failed to open apk database: Permission denied
// # curl https://www.google.com
sh: curl: not found
// # touch /tmp/testcreatefile1
touch: /tmp/testcreatefile1: Permission denied
// # touch /tmp/testcreatefile2
touch: /tmp/testcreatefile2: Permission denied
// # touch /etc/testcreatefile3
touch: /etc/testcreatefile3: Permission denied
// # touch /proc/testcreatefile4
touch: /proc/testcreatefile4: No such file or directory
// # cp /tmp/testcreatefile /proc/testprohibitfile
cp: can't stat '/tmp/testcreatefile': No such file or directory
// # ls -hl /proc | grep test
// # exit
ubuntu@ip-10-0-1-104:~$ docker container stop sectestrestrict && docker container rm sectestrestrict
sectestrestrict
sectestrestrict
ubuntu@ip-10-0-1-104:~$
```

Docker: The Next-Gen of Virtualization



270

Workshop: Docker Security

- Content Trust
- Purpose: ทำให้ enable feature "DOCKER_CONTENT_TRUST" และใช้งาน image แบบ sign/unsign

```
root@labdocker:~# docker push praparn/alpine:sign
The push refers to a repository [docker.io/praparn/alpine]
6102f0d2ad33: Layer already exists
sign: digest: sha256:65242e8220a341cec40628caae77eb4acd2fc252329aa853526fde15a4a1d85 size: 528
Signing and pushing trust metadata
You are about to create a new root signing key passphrase. This passphrase
will be used to protect the most sensitive key in your signing system. Please
choose a long, complex passphrase and be careful to keep the password and the
key file itself secure and backed up. It is highly recommended that you use a
password manager to generate the passphrase and keep it safe. There will be no
way to recover this key. You can find the key in your config directory.
Enter passphrase for new root key with ID ca6ab4b:
Repeat passphrase for new root key with ID ca6ab4b:
Enter passphrase for new repository key with ID 68d88cd (docker.io/praparn/alpine):
Repeat passphrase for new repository key with ID 68d88cd (docker.io/praparn/alpine):
```

Docker: The Next-Gen of Virtualization



271

Registry

• • •

Docker: The Next-Gen of Virtualization



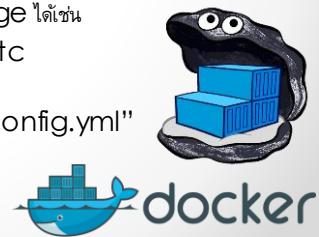
272

Registry

- Registry เป็น container แบบหนึ่งที่ใช้สำหรับจัดเก็บ image ที่สั่งขึ้นไว้บน private server โดยไม่จำเป็นต้องผ่าน public internet (hub.docker.com) ทำให้สามารถจัดเก็บ image ภายในองค์กรได้อย่างปลอดภัย
- Registry เป็นพื้นที่สำหรับจัดเก็บ image ที่ถูกสร้างขึ้นโดยแบ่งออกตาม image name และ tag version ที่กำหนดไว้

Ex: labdocker/alpineweb:latest

- กรณีการใช้งาน registry ระหว่าง Server ด้วยกันสามารถกำหนด TLS security ระหว่างกันได้
- สามารถเลือก storage backend อื่นๆในการจัดเก็บ image ได้ เช่น Etc: s3 (aws), azure, gcs (google cloud) etc
- มี option gc (garbage collect) ด้วย option "bin/registry garbage-collect [--dry-run] config.yml"



Docker: The Next-Gen of Virtualization

273

Workshop: Registry

- Part1: Registry on single docker-machine
- เริ่มใช้งาน registry โดย download image registry ตามตัวอย่าง


```
docker image pull registry:2.6.2 (registry version 2.0)
```
- Start container เพื่อเริ่มใช้งาน


```
docker container run -d -p 5000:5000 \
--restart=always --name registry \
-e REGISTRY_STORAGE_DELETE_ENABLED=true \
-mount type=bind,source=/home/docker,target=/var/lib/registry \
registry:2.7.1
```
- ดำเนินการ tag image และทดสอบ push image docker เข้า registry


```
docker image tag labdocker/alpine:latest \
localhost:5000/alpine:latest
```

```
docker push localhost:5000/alpine:latest
```
- *ต้องบันทึก digest image ไว้เพื่อใช้งานอย่างหลัง

Docker: The Next-Gen of Virtualization



274

Workshop: Registry

- ในการตรวจสอบ registry image file เพื่อให้เกิดความง่ายต่อการใช้งาน docker ได้จัดเตรียม HTTP API สำหรับการเรียกใช้งานบน registry ผ่าน curl/customize application ซึ่งรองรับการใช้งาน อาทิเช่น
- การ List รายชื่อ image ที่จัดเก็บไว้บัน

```
curl -X GET http://localhost:5000/v2/_catalog
```

- ตรวจสอบ revision image tag ที่จัดเก็บไว้ในแต่ละ image

```
curl -X GET http://localhost:5000/v2/<name>/tags/list
```

```
curl -X GET http://localhost:5000/v2/alpine/tags/list
```

- สามารถทำการลบ image ที่จัดเก็บไว้โดยอ้างอิงจาก digest

```
curl -X DELETE  
http://localhost:5000/v2/alpine/manifests/<digest>
```

Docker: The Next-Gen of Virtualization



275

Workshop: Registry

- ตรวจสอบ image ทาง physical file จาก docker-machine

```
cd /home/docker  
cd /docker/registry/v2
```

- ลบ image file ออกจาก docker-machine

```
docker image rm localhost:5000/alpine:latest
```

- ทำการดึง image ใหม่มาจากรегистรี

```
docker image pull localhost:5000/alpine:latest
```

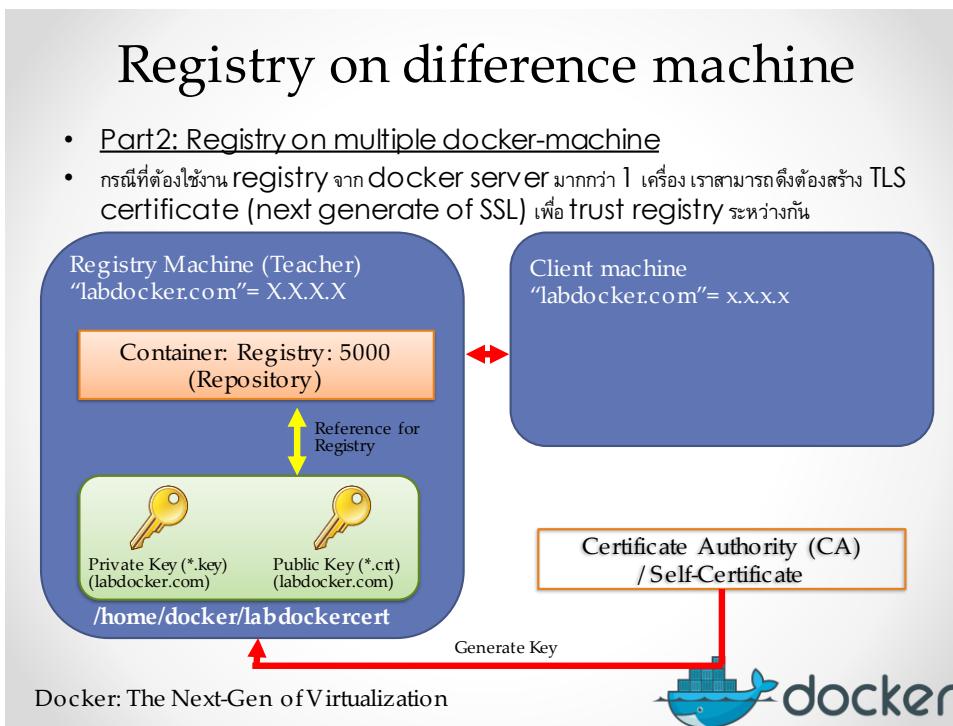
- ทำการลบ container registry เพื่อยกเลิกการใช้งาน

```
docker container stop registry  
docker container rm registry
```

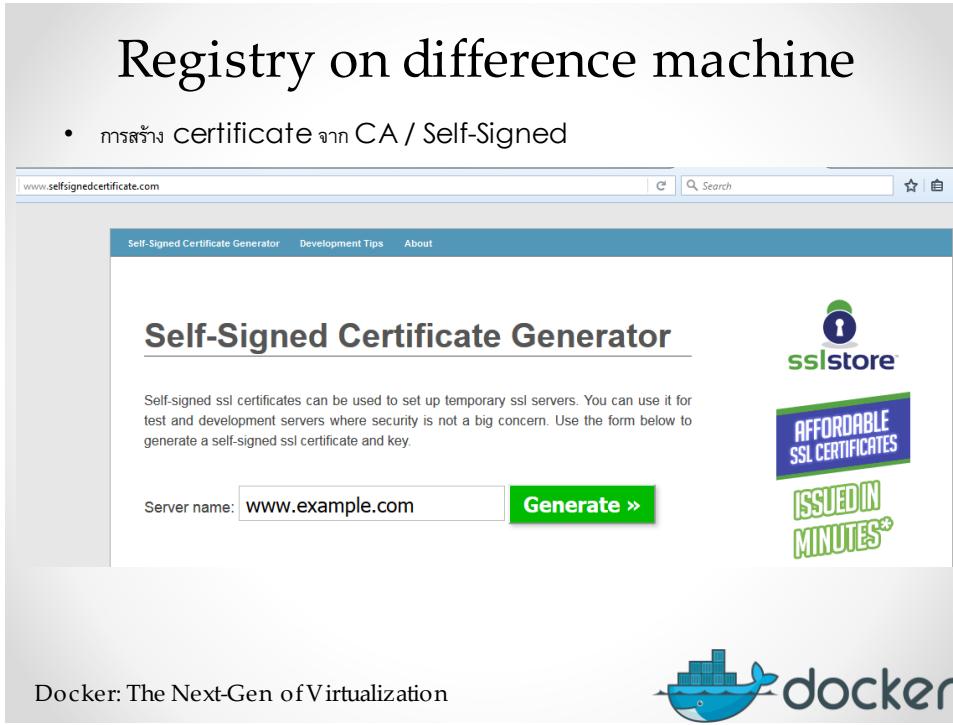
Docker: The Next-Gen of Virtualization



276



277



278

Registry on difference machine

- เพิ่ม domain "labdocker.com" ลงใน /etc/hosts (labdocker, labdocker2)

```
# The following lines are desirable for IPv6 capable hosts
# (added automatically by netbase upgrade)

::1      ip6-localhost ip6-loopback
fe00::0  ip6-localnet
ff00::0  ip6-mcastprefix
ff02::1  ip6-allnodes
ff02::2  ip6-allrouters
ff02::3  ip6-allhosts
192.168.99.100 labdocker.com
```

```
ping labdocker.com (192.168.99.100): 56 data bytes
64 bytes from 192.168.99.100: seq=0 ttl=64 time=0.110 ms
64 bytes from 192.168.99.100: seq=1 ttl=64 time=0.108 ms
64 bytes from 192.168.99.100: seq=2 ttl=64 time=0.105 ms
```
--- labdocker.com ping statistics ---
3 packets transmitted, 3 packets received, 0% packet loss
round-trip min/avg/max = 0.105/0.107/0.110 ms
docker@labdocker:~$
```

Docker: The Next-Gen of Virtualization



279

## Registry on difference machine

- สำหรับกรณีเครื่อง Ubuntu ที่ไม่มีชั้น self-certificate ให้ทำการ trust root ca ดังนี้

```
sudo mkdir /usr/local/share/ca-certificates/labdocker.com:5000
sudo cp labdocker.com.crt /usr/local/share/ca-certificates/labdocker.com:5000
sudo update-ca-certificates
sudo service docker restart
```

```
ubuntu@ip-10-0-1-182:~$ ls ~/docker-workshop-112018-cloud/Workshop-2-3-Registry
instruction.txt labdocker.com.crt labdocker.com.key
ubuntu@ip-10-0-1-182:~$ cd ~/docker-workshop-112018-cloud/Workshop-2-3-Registry
ubuntu@ip-10-0-1-182:~/docker-workshop-112018-cloud/Workshop-2-3-Registry$ sudo mkdir /usr/local/share/ca-certificates/labdocker.com:5000
ubuntu@ip-10-0-1-182:~/docker-workshop-112018-cloud/Workshop-2-3-Registry$ sudo cp labdocker.com.crt /usr/local/share/ca-certificates/labdocker.com:5000
ubuntu@ip-10-0-1-182:~/docker-workshop-112018-cloud/Workshop-2-3-Registry$ sudo update-ca-certificates
Updating certificates in /etc/ssl/certs...
1 added, 0 removed, 0 renamed, done.
Running hooks in /etc/ca-certificates/update.d...
done.
ubuntu@ip-10-0-1-182:~/docker-workshop-112018-cloud/Workshop-2-3-Registry$ sudo service docker restart
ubuntu@ip-10-0-1-182:~/docker-workshop-112018-cloud/Workshop-2-3-Registry$
```

Docker: The Next-Gen of Virtualization



280

# Registry on difference machine

- เริ่มใช้งาน registry โดย start container ดังนี้ (labdocker)

```
docker container run -d -p 5000:5000 \
--restart=always --name registrylab \
--mount type=bind,source=$(pwd),target=/var/lib/registry \
--mount type=bind,source=$(pwd),target=/certs \
--mount type=bind,source=$(pwd),target=/auth \
-e "REGISTRY_AUTH=hpasswd" \
-e "REGISTRY_AUTH_HTPASSWD_REALM=Registry Realm" \
-e REGISTRY_AUTH_HTPASSWD_PATH=/auth/hpasswd \
-e REGISTRY_STORAGE_DELETE_ENABLED=true \
-e REGISTRY_HTTP_TLS_CERTIFICATE=/certs/labdockerc.com.crt \
-e REGISTRY_HTTP_TLS_KEY=/certs/labdockerc.com.key \
registry:2.7.1
```

- ดำเนินการ tag image, login และทดสอบ push image docker เข้า registry

```
docker image tag labdocker/alpine:latest labdockerc.com:5000/alpine:1.0
docker login -u <username> -p <password>
docker image push labdockerc.com:5000/alpine:1.0
```

Docker: The Next-Gen of Virtualization



281

# Registry on difference machine

- ในการตรวจสอบ registry image file สำหรับ registry ที่มีการ enable TLS เรียนรู้อย่างต้องเพิ่มพารามิเตอร์ “--cacert” เพื่อรับ Certificate สำหรับการใช้งานและเปลี่ยน protocol เป็น https
- ทำการ List รายการ image ที่จัดเก็บไว้ในบันได
 

```
curl -u <username:password> --cacert /Share_DockerToolbox/labdockerc.com.crt -X GET https://labdockerc.com:5000/v2/_catalog
```
- ตรวจสอบ revision image tag ที่จัดเก็บไว้ในแต่ละ image
 

```
curl -u <username:password> --cacert /home/docker/labdockercert/labdockerc.com.crt -X GET https://labdockerc.com:5000/v2/alpine/tags/list
```
- สามารถทำการลบ image ที่จัดเก็บไว้โดยอ้างอิงจาก digest
 

```
curl -u <username:password> --cacert /home/docker/labdockercert/labdockerc.com.crt -X DELETE https://labdockerc.com:5000/v2/alpine/manifests
```

Docker: The Next-Gen of Virtualization



282

# Registry on difference machine

- ตั้ง image ผ่าน labdocker 2

```
ubuntu@ip-10-0-1-239:~/docker-workshop-112018-cloud/Workshop-2-3-Registry$ docker login labdocker.com:5000
Username: docker
Password:
WARNING! Your password will be stored unencrypted in /home/ubuntu/.docker/config.json.
Configure a credential helper to remove this warning. See
https://docs.docker.com/engine/reference/commandline/login/#credentials-store

Login Succeeded
ubuntu@ip-10-0-1-239:~/docker-workshop-112018-cloud/Workshop-2-3-Registry$ docker image pull labdocker.com:5000/alpine:1.0
1.0: Pulling from alpine
Digest: sha256:9289226e401a9d18f0ea01f8azf38d328ef039db4e1edcc45c630314a0457d5b
Status: Downloaded newer image for labdocker.com:5000/alpine:1.0
ubuntu@ip-10-0-1-239:~/docker-workshop-112018-cloud/Workshop-2-3-Registry$
```

- Optional: สำหรับเพิ่มรันเว็บเพื่อตรวจสอบ Container

```
-v `pwd`/auth:/auth \
-e "REGISTRY_AUTH=htpasswd" \
-e "REGISTRY_AUTH_HTPASSWD_REALM=Registry
Realm" \
```

- ทำการลบ Container registry เพื่อยกเลิกการใช้งาน

```
docker container stop registry
docker container rm -v registry
```

Ref. <https://docs.docker.com/registry/deploying/>

Docker: The Next-Gen of Virtualization



283

The dashboard displays various metrics for the Docker Trusted Registry. Under 'Host Status', there are three pie charts for RAM (2047MB/14.7GB), Storage (7.7GB/9.7GB), and CPU (1.13400%). Under 'Container Status', there are four cards for Admin Server, Auth Server, Load Balancer, and Log Aggregator, each showing RAM, Network, and CPU usage over time.

## Docker Trust Registry (DTR)

- Commercial support
- GUI for managing everything
- Integrated LDAP authentication
- GC schedule available

Docker: The Next-Gen of Virtualization

284

# Open Source Registry (Experiment)

- <http://port.us.org>

Docker: The Next-Gen of Virtualization



285

# Artifact Registry

- Sonatype

Nexus Repository Manager 3 > Private Registry for Docker

Available in Nexus Repository OSS and Nexus Repository Pro

Docker containers and their usage have revolutionized the way applications and the underlying operating system are packaged and deployed to development, testing and production systems. The creation of the [Open Container Initiative](#), and the involvement of a large number of stakeholders, guarantees that the ecosystem of tools around the lightweight containers and their usage will continue to flourish.

Docker Hub is the original registry for Docker container images and it is being joined by more and more other publicly available registries such as the [Google Container Registry](#) and others.

Nexus Repository Manager Pro and Nexus Repository OSS support Docker registries as the Docker repository format for hosted and proxy repositories. You can expose these repositories to the client-side tools directly or as a repository group, which is a repository that merges and exposes the contents of multiple repositories in one convenient URL. This allows you to reduce time and bandwidth usage for accessing Docker images in a registry as well as share your images within your organization in a hosted repository. Users can then launch containers based on those images, resulting in a completely private Docker registry with all the features available in the repository manager.

Docker: The Next-Gen of Virtualization



286

# Artifact Registry

- Harbor CNCF Registry (Incubator)

**Harbor**  
Cloud Native Computing Foundation (CNCF)  
Provisioning - Container Registry

An open source trusted cloud native registry project that stores, signs, and scans content.

|              |                                                                                                                                                       |
|--------------|-------------------------------------------------------------------------------------------------------------------------------------------------------|
| Website      | <a href="https://goharbor.io/">https://goharbor.io/</a>                                                                                               |
| Repository   | <a href="https://github.com/goharbor/harbor">https://github.com/goharbor/harbor</a>                                                                   |
| GitHub       | 0 ⚡ 7,027                                                                                                                                             |
| LinkedIn     | <a href="https://www.linkedin.com/company/cloud-native-computing-foundation/">https://www.linkedin.com/company/cloud-native-computing-foundation/</a> |
| Twitter      | @project_harbor                                                                                                                                       |
| First Commit | 3 years ago                                                                                                                                           |
| Contributors | 121                                                                                                                                                   |
| Headquarters | San Francisco, California                                                                                                                             |

**Tweets** by @project\_harbor

Project Harbor (@project\_harbor) [#currentlyreading](#) [#Insight](#) from @Maks\_Postumeni on how he built H.A. Harbor with #Ansible and #Rancher

Docker: The Next-Gen of Virtualization



287

# Swarm

• • •

Docker: The Next-Gen of Virtualization



288

## Swarm: Conceptual

- Swarm ถือเป็น native tool ในการจัดการ docker-engine หลายเครื่องให้รวมเป็น resource pool เดียวกัน เพื่อให้ง่ายต่อการบริหารจัดการ
- เพิ่มขีดความสามารถในการทำงานร่วมกันของ docker-engine
- Hybrid Swarm Cluster (Windows / Linux)
- Fail Over / High Availability (HA)
- Micro Service



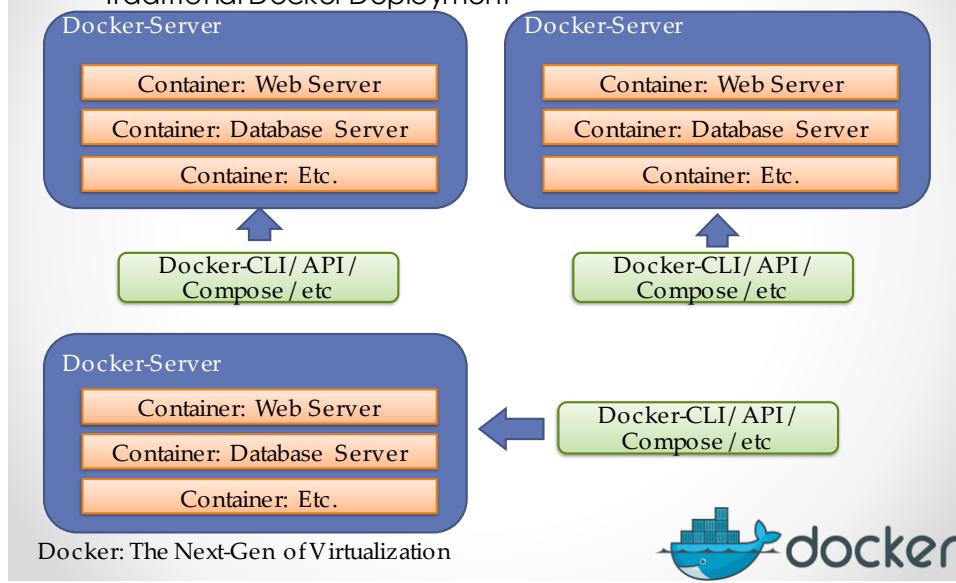
Docker: The Next-Gen of Virtualization



289

## Swarm: Conceptual

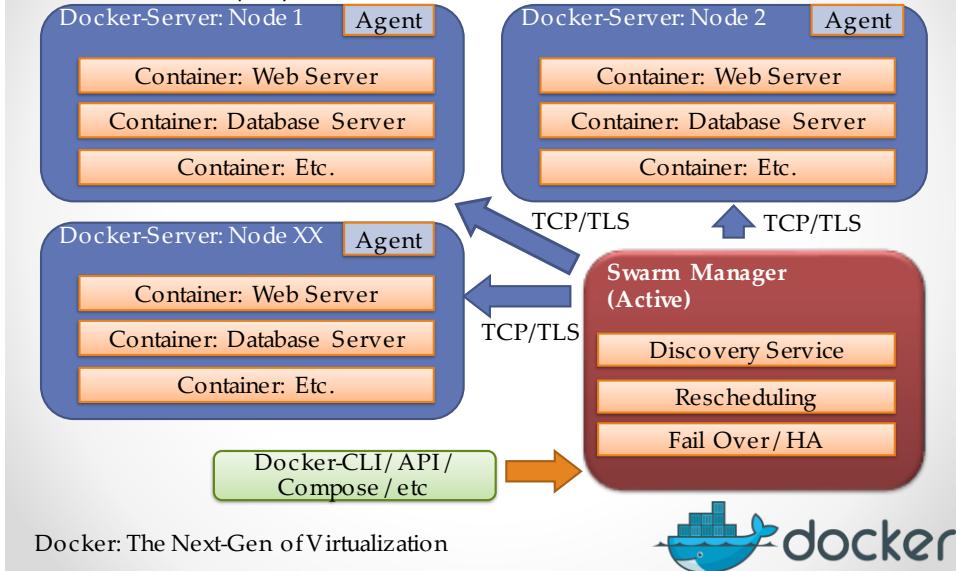
- Traditional Docker Deployment



290

# Swarm: Conceptual

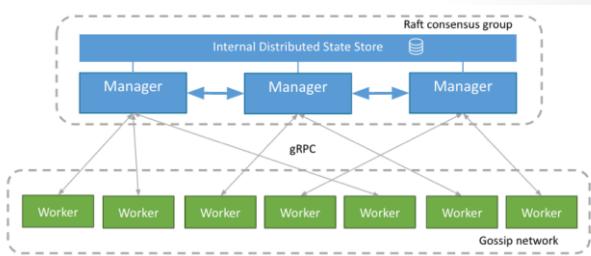
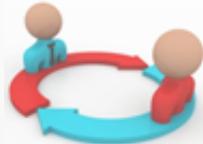
- Swarm Deployment



291

# Swarm Mode Architecture

- Swarm Mode (build-in swarm) เป็นฟังก์ชันที่มาพร้อม docker-engine ตั้งแต่ แรก และสามารถบริหารจัดการ Swarm ได้ด้วยตัวเองรวมถึง build-in TLS certificate เพื่อใช้ ทำงานระหว่าง node ทันที (Security On the Box)
- Swarm Role
  - Manager Node
  - Worker Node



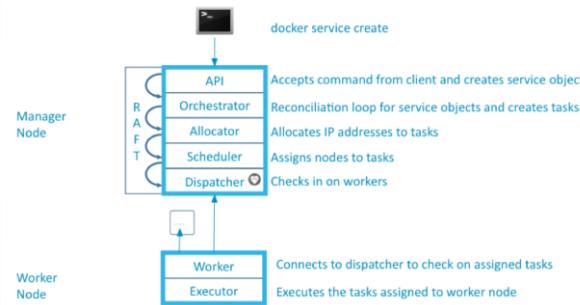
Docker: The Next-Gen of Virtualization



292

# Swarm Mode Architecture

## Node Breakdown

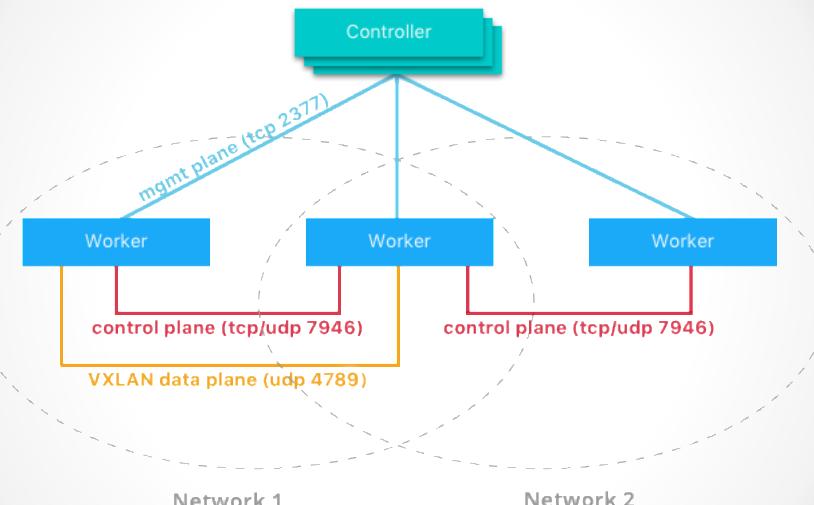


Docker: The Next-Gen of Virtualization



293

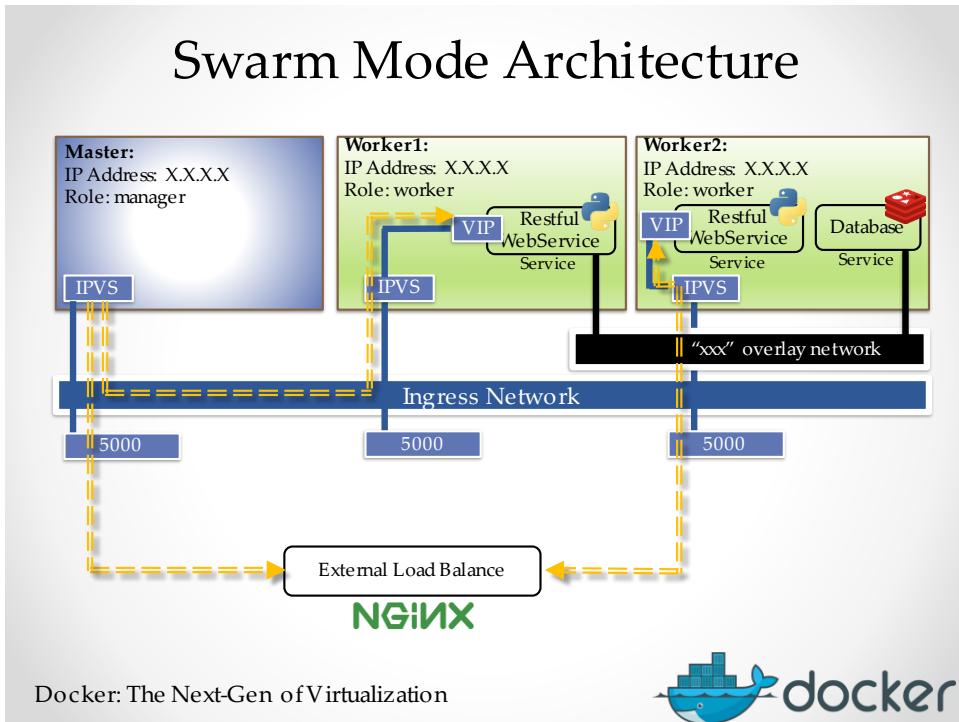
# Swarm Mode Architecture



Docker: The Next-Gen of Virtualization



294



295

# Swarm Mode Architecture

<https://blog.docker.com/2016/07/docker-built-in-orchestration-ready-for-production-docker-1-12-goes-ga/>

The core team also wanted to give a special thanks to one of our animal maintainers and Docker Captain, [Chenert Kannabilli](#), who through his own undertaking, drove the amazing [DockerSwarm2000](#) initiative which rallied the entire community around scaling an RC of 1.12 with swarm mode to nearly 2.4K active hosts and 102K containers. This was achieved through our work on the Docker daemon to support unprivileged access to their memory map shapes and in how we map to Raspberry Pis, to various clouds to VMs from x86 architectures to ARM-based systems. Through this evaluation using live data, we identified that built-in orchestration in Docker has already-in its first release-doubled Docker's [orchestration scale](#) in just a half a year. While this validates the scalability of the architecture, there is still headroom for greater performance optimization in the future.



**Nathan LeClaire** @upthecyberpunks 48m  
.@docker #swarm team forming like Voltron to inspect the #DockerSwarm2K results @stevvooee @alazzardl @mikegoelzer



**Nathan LeClaire** @upthecyberpunks  
@chanwit @dongluochen

Jul 30, 2016, 7:56 AM

Docker: The Next-Gen of Virtualization



296

# Swarm Mode Architecture

swarmzilla / swarm3k

Code Issues Pull requests Projects Pulse Graphs Watch 11 Star 21 Fork 21

SwarmZilla 3000 Collaborative Project

112 commits 1 branch 0 releases 21 contributors

Branch: master New pull request Find file Clone or download

chanwit committed on GitHub got 3185 nodes

Latest commit 763@dad a day ago

README.md got 3185 nodes a day ago

dashboard2.png Add files via upload 26 days ago

events.png Add files via upload 26 days ago

monitoring.md Update monitoring.md 2 days ago

README.md

**Swarm3k - Friday, 28th October 2016 3.00PM UTC**

SwarmZilla 3000 Collaborative Project

Docker: The Next-Gen of Virtualization 

297

# Swarm Mode Architecture

| Name          | Company                           | Number of Nodes Expected to Contribute |
|---------------|-----------------------------------|----------------------------------------|
| swarm3k       | SwarmZilla 3000                   | 100                                    |
| arthurwong    | my own boss                       | 10                                     |
| apmashimo     | PetaMD                            | 50                                     |
| avneron       | Rackspice                         | 100                                    |
| blueduckatsea | Internet Thailand                 | 500                                    |
| bpaschal      | n/a                               | 10                                     |
| coderock      | Neverlock                         | 10                                     |
| chenmengtao   | Demoware                          | 1000                                   |
| cristianbu    | Jahil                             | 50                                     |
| dplimont      | OVH                               | 500                                    |
| dukejones     | Cubatrix                          | 10                                     |
| eloykoz       | Ayara Clicker                     | 10                                     |
| erikspina     | Personal                          | 20                                     |
| ericspina     | SPRINTER                          | 630                                    |
| erickson      | Personal                          | 30                                     |
| esherickson   | N/A                               | 10                                     |
| evanm         | HotelQuickly                      | 200                                    |
| evanm         | N/A                               | 25                                     |
| evanm         | Packet                            | 100                                    |
| evanm         | Containerize This! The Conference | 10                                     |
| evanm         | FirePress                         | 10                                     |
| evanm         | TRAXx                             | 10                                     |
| evanm         | Personal                          | 10                                     |
| evanm         | Hab                               | 10                                     |
| evanm         | Myself J.                         | 20                                     |
| evanm         | Emerging Technology Advisors      | 100                                    |
| evanm         | Personal                          | 20                                     |
| evanm         | ThamFlow                          | 50                                     |
| evanm         | Personal                          | 10                                     |
| evanm         | Personal                          | 10                                     |
| evanm         | New Technology                    | 100                                    |
| evanm         | NexwayGroup                       | 30                                     |

About Monitoring Swarm3k Metrics, Logs and Events by Docker & Logstash Schedule: October 28, 2016, 3 PM UTC

Swarm Managers: 3

Swarm Nodes: 4700

Swarm Tasks: 47433

Swarm Task Errors: N/A

Task State Overview: 99.9% (0.00%)

Swarm Task Status: 100% (0.00%)

System load avg: 2.40

Number of Nodes: 47433

Docker: The Next-Gen of Virtualization 

298

## K8S vs Docker what is the difference ?

| Topic                        | K8S                                                                                                            | Docker/Swarm                                                                                         |
|------------------------------|----------------------------------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------|
| Architecture                 | Open-system (Base on cluster manager "Borg" for support complex workload)                                      | <b>Swarm:</b> 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"                                                                                              | <b>Docker:</b> By container name<br><b>Swarm:</b> By service/stack name                              |
| Level of workload management | Service Level: (Simple)<br>Replication Level: (Auto healing)<br>Deployment Level: (Auto healing + Roll Update) | <b>Docker:</b> N/A<br><b>Swarm:</b> 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                                                                                  |

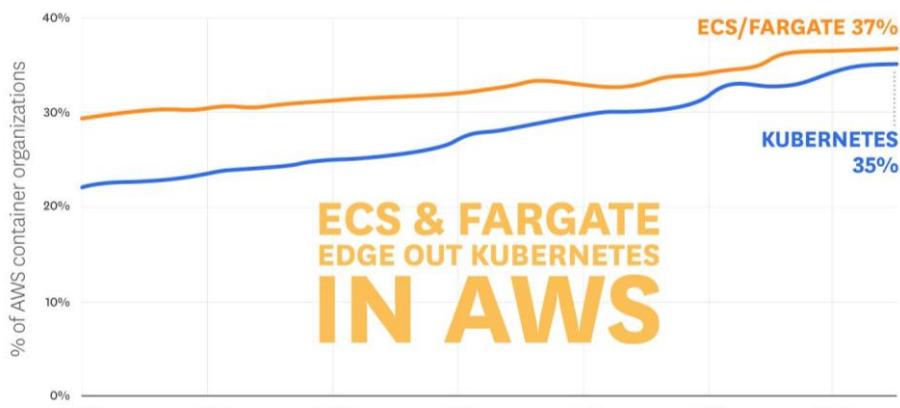
Docker: The Next-Gen of Virtualization



299

## K8S vs Docker what is the difference ?

Share of AWS Container Environments



Source: Datadog

Docker: The Next-Gen of Virtualization



300

## Swarm Init/Join

- เริ่มการสร้าง swarm ด้วยคำสั่ง docker swarm init [OPTIONS] เพื่อให้ docker เริ่มทำงานใน swarm mode

```
docker swarm init --listen-addr <ip address>:<port>
```

```
docker swarm init --advertise-addr 10.0.1.104:2377 --task-history-limit 2
```

```
...p_112018 — Terminal MAC Pro — -bash ...ssh -i k8s_lab ubuntu@13.229.126.99 ...— ssh -i k8s_lab ubuntu@13.229.98.3 sh -i k8s_lab ubuntu
[ubuntu@ip-10-0-1-104:~]$ docker swarm init --advertise-addr 10.0.1.104:2377 --task-history-limit 2
Swarm initialized: current node (3tnvltyyohbekgvgvbjijovy) is now a manager.

To add a worker to this swarm, run the following command:

 docker swarm join --token SWMTKN-1-1onpobqw0hilpc5i90sz803tlhyi1h4k84uok10y814bvhqf2-0ag405byoqpingqhojy6r866tv 10.0.1.104:2377

To add a manager to this swarm, run 'docker swarm join-token manager' and follow the instructions.

[ubuntu@ip-10-0-1-104:~]$ docker swarm join-token manager
To add a manager to this swarm, run the following command:

 docker swarm join --token SWMTKN-1-1qnpojqw0hilpc5i90sz803tlhyi1h4k84uok10y814bvhqf2-3u9rqxfwhbhw52qwqeeiewybl 10.0.1.104:2377
[ubuntu@ip-10-0-1-104:~]$
```

Docker: The Next-Gen of Virtualization



301

## Swarm Init/Join

- Option:
  - cert-expiry duration: กำหนดระยะเวลาในการ expire certificate (default: 2160 hours = 90 Days)
  - dispatcher-heartbeat duration: กำหนดช่วงเวลาในการ check heartbeat ภายใน swarm (Default: 5 seconds)
  - external-ca value: กำหนดให้ใช้งาน CA certificate ภายนอกเพื่อที่ trust node ภายใน swarm
  - force-new-cluster: บังคับการสร้าง swarm cluster ใหม่
  - advertise-addr value: ระบุ ip address ที่ใช้ระหว่าง swarm node (default: 0.0.0.0:2377)
  - task-history-limit <int>: กำหนดการจัดเก็บ history task ย้อนหลังบน swarm (default: 10)
  - availability:<"active"/"pause"/"drain">
  - Etc.

Docker: The Next-Gen of Virtualization



302

## Swarm Init/Join

- ทำต่อ join node เข้าไปยัง swarm cluster ด้วยคำสั่ง

```
docker swarm join --token<token><ip of swarm manager: port>
```

```
...p.112018 — Terminal MAC Pro — -bash | ...ssh -i k8s_lab ubuntu@13.229.126.99 | ...ssh -i k8s_lab ubuntu@13.229.98.3 | ...sh -i k8s_lab ubuntu@18.136.196.130 | +
[ubuntu@ip-10-0-1-163: ~]$ docker swarm join --token SiMTKN-1-1qnpbqw0hilpc5i90sz803tlhyi1h4k84uok10y814bvhqf2-0ag405byoqpiqgqhoyle86tv 10.0.1.104:2377
This node joined a swarm as a worker.
[ubuntu@ip-10-0-1-163: ~]$
```

```
...p.112018 — Terminal MAC Pro — -bash | ...ssh -i k8s_lab ubuntu@13.229.126.99 | ...ssh -i k8s_lab ubuntu@13.229.98.3 | ...sh -i k8s_lab ubuntu@18.136.196.130 | +
[ubuntu@ip-10-0-1-62: ~]$ docker swarm join --token SiMTKN-1-1qnpbqw0hilpc5i90sz803tlhyi1h4k84uok10y814bvhqf2-0ag405byoqpiqgqhoyle86tv 10.0.1.104:2377
This node joined a swarm as a worker.
[ubuntu@ip-10-0-1-62: ~]$
```

Docker: The Next-Gen of Virtualization



303

## Swarm Init/Join

- Option:
  - token <value>: กำหนด token เพื่อใช้ในการ trust swarm cluster ร่วมกัน
  - listen-addr <value>: ระบุ ip address ที่ใช้คุยกับ node (default: 0.0.0.0:2377)
  - advertise-addr <value>: ระบุ ip address ที่ประกาศให้ผู้อื่นมาคุยกับในกรณีเป็น swarm-manager
  - availability:<"active"/"pause"/"drain">
  - Etc.

Docker: The Next-Gen of Virtualization



304

## Swarm Init/Join

- ตรวจสอบ Token ก่อนการ Join Swarm (Worker/Manager)

```
docker swarm join-token <option><worker/manager>
```

```
...p_112018 — Terminal MAC Pro — bash | ...ssh -i k8s.lab ubuntu@13.229.126.99 | ...ssh -i k8s.lab ubuntu@13.229.98.3 ... | ...h -i k8s.lab ubuntu@13.229.98.3
ubuntu@ip-10-0-1-104:~$ docker swarm join-token worker
To add a worker to this swarm, run the following command:

 docker swarm join --token SWMTKN-1-1qnpobqw0hilpc5i90sz803tlhyi1h4k84uok10y814bvhqf2-0ag405byoqpiqgqhoeyr866tv 10.0.1.104:2377

ubuntu@ip-10-0-1-104:~$ docker swarm join-token manager
To add a manager to this swarm, run the following command:

 docker swarm join --token SWMTKN-1-1qnpobqw0hilpc5i90sz803tlhyi1h4k84uok10y814bvhqf2-3u9rqxfwhbhw52qwqeeiewybl 10.0.1.104:2377
ubuntu@ip-10-0-1-104:~$
```

- Option:

- Q, --Q : แสดงเฉพาะ token เท่านั้น
- rotate: ทำการเปลี่ยน token ใหม่

Docker: The Next-Gen of Virtualization



305

## Swarm Init/Join

- ตรวจสอบ node ที่ทำงานภายใต้ Swarm cluster ด้วยคำสั่ง docker node ls

```
docker node ls
```

```
...rshhop_112018 — Terminal MAC Pro — bash | ...~ — ssh -i k8s.lab ubuntu@13.229.126.99 | ...~ — ssh -i k8s.lab ubuntu@13.229.98.3 ... | ...~ — ssh -i k8s.lab ubuntu@18.136.19
ubuntu@ip-10-0-1-104:~$ docker node ls
ID HOSTNAME STATUS AVAILABILITY MANAGER STATUS ENGINE VERSION
4g6pqogaae5co8zxe8w6pxfd6 ip-10-0-1-62 Ready Active Leader 18.06.1-ce
3tnvltyohbekgygvb1jjoyv * ip-10-0-1-104 Ready Active Active 18.06.1-ce
ua0ajjfjd4d1mksqzwr80oy2ru ip-10-0-1-163 Ready Active Active 18.06.1-ce
ubuntu@ip-10-0-1-104:~$
```

Docker: The Next-Gen of Virtualization



306

## Swarm Init/Join

- Option:
  - promote <ID>: ทำการ promote node จาก worker กลับเป็น manager
  - demote <ID>: ทำการ demote node จาก manager กลับเป็น worker มากดิ
  - Inspect <ID>: ตรวจสอบค่าคอนฟิกที่รันอยู่ของ node
  - ls: แสดง node ทั้งหมดที่รันอยู่ภายใน docker swarm cluster
  - ps <ID>: แสดงงานทั้งหมดที่รันอยู่ภายใต้ node
  - rm <ID>: ลบ node ออกจาก swarm cluster
  - update <ID>: ทำการ update node information
    - --av ailability <status>: อัปเดตสถานะของ node (active/pause/drain)
    - --label-add <value>: สร้าง custom label เพื่อใช้อ้างอิงให้กับ node สำหรับการรัน service
    - --label-remove <value>: ลบ custom label
    - --role <role>: role of node (worker/manager)

Docker: The Next-Gen of Virtualization



307

## Workshop: Swarm Mode

- ทำการตรวจสอบเครื่องใน LAB ที่ได้รับมอบหมายแต่ละกลุ่มซึ่ง

Name: Training\_DockerZeroToHero\_StudentG1\_1  
 (IP: 10.0.1.X), Role: Master  
 Name: Training\_DockerZeroToHero\_StudentG1\_2  
 (IP: 10.0.1.X), Role: Worker1  
 Name: Training\_DockerZeroToHero\_StudentG1\_3  
 (IP: 10.0.1.X), Role: Worker2

- เริ่ม initial swarm ที่เครื่อง Master

```
docker swarm init --secret labdocker --listen-addr<private ip of Master>:2377
```

Docker: The Next-Gen of Virtualization



308

# Workshop: Swarm Mode

- ทำการ Join swarm-node1, swarm-node2 เข้าสู่ swarm cluster

```
docker swarm join --token SWMTKN-1-
1qnpobqw0hilpc5i90sz803tlhyi1h4k84uok10y814bvhyqf2-
0ag405byoqpiqgqhoy6r866tv 10.0.1.104:2377
```

- ตรวจสอบสถานะของ Swarm หลังจาก join node เข้าไปเรียบร้อยแล้ว

```
docker node ls
```

```
...rkshop_112018 — Terminal MAC Pro — -bash ... ~ ssh -i k8s_lab ubuntu@13.229.126.99 ... ~ ssh -i k8s_lab ubuntu@13.229.98.3 ... ~ ssh -i k8s_lab ubuntu@18.136.196.130 ...
ubuntu@ip-10-0-1-104:~$ docker node ls
ID HOSTNAME STATUS AVAILABILITY MANAGER STATUS ENGINE VERSION
4g6pqogaae5co8zx8w6pxfd6 ip-10-0-1-62 Ready Active Active
3tnvltyohbekgygsb1jjoyv * ip-10-0-1-104 Ready Active Leader
ua0ajjfjd4d1mkszwr80oy2ru ip-10-0-1-163 Ready Active
ubuntu@ip-10-0-1-104:~$
```



309

## Swarm Service

- สำหรับการสร้าง container ใน swarm ให้การเปลี่ยนแปลงรูปแบบใหม่ที่น่าสนใจ โดยกำหนดการรัน container ดูเหมือนกับการรัน "docker service" เพื่อจัดการแทนการรัน container แทนคำสั่ง "docker run" โดยเราสามารถกำหนดค่าพารามิเตอร์ได้เช่นกัน

```
docker service <action> (create/inspect/tasks/scale/ls/rm/update)
```

```
Ex: docker service create -dt --name nodejs \
labdocker/alpineweb:latest node hello.js
```

```
...018 — Terminal MAC Pro — -bash ... ~ i k8s_lab ubuntu@13.229.126.99 ... ~ i k8s_lab ubuntu@13.229.98.3 ... ~ k8s_lab ubuntu@18.136.196.130 ...
ubuntu@ip-10-0-1-104:~$ docker service create --name nodejs -dt labdocker/alpineweb:latest node hello.js
4i4gu0o15f819johbgn994f7
ubuntu@ip-10-0-1-104:~$
```



310

# Swarm Service

```

Options:
 --config config Specify configurations to expose to the service
 --constraint list Placement constraints
 --container-label list Container labels
 --credential-spec credential-spec
 -d, --detach Exit immediately instead of waiting for the service to converge (default true)
 --dns list Set custom DNS servers
 --dns-option list Set DNS options
 --dns-search list Set custom DNS search domains
 --endpoint-mode string Endpoint mode (vip or dnsrr) (default "vip")
 --entrypoint command Overwrite the default ENTRYPOINT of the image
 -e, --env list Set environment variables
 --env-file list Read in a file of environment variables
 --group list Set one or more supplementary user groups for the container
 --health-cmd string Command to run to check health
 --health-interval duration Time between running the check (ms|s|m|h)
 --health-retries int Consecutive failures needed to report unhealthy
 --health-start-period duration Start period for the container to initialize before counting retries towards unstable (ms|s|m|h)
 --health-timeout duration Maximum time to allow one check to run (ms|s|m|h)
 --help Print usage
 --host list Set one or more custom host-to-IP mappings (host:ip)
 --hostname string Container hostname
 -l, --label list Service labels
 --limit-cpu decimal Limit CPUs
 --limit-memory bytes Limit Memory
 --log-driver string Logging driver for service
 --log-opt list Logging driver options
 --mode string Service mode (replicated or global) (default "replicated")
 --mount mount Attach a filesystem mount to the service
 --name string Service name

```

- Reference: [https://docs.docker.com/engine/reference/commandline/service\\_create/](https://docs.docker.com/engine/reference/commandline/service_create/)

Docker: The Next-Gen of Virtualization



311

# Swarm Service

```

--name string Service name
--network network Network attachments
--no-healthcheck Disable any container-specified HEALTHCHECK
--no-resolve-image Do not query the registry to resolve image digest and supported platforms
--placement-pref pref Add a placement preference
-p, --publish port Publish a port as a node port
-q, --quiet Suppress progress output
--read-only Mount the container's root filesystem as read only
--replicas uint Number of tasks
--reserve-cpu decimal Reserve CPUs
--reserve-memory bytes Reserve Memory
--restart-condition strg Restart when condition is met ("none"|"on-failure"|"any") (default "any")
--restart-delay duration Delay between restart attempts (ns|us|ms|s|m|h) (default 5s)
--restart-max-attempts uint Maximum number of restarts before giving up
--restart-window duration Window used to evaluate the restart policy (ns|us|ms|s|m|h)
--rollback-delay duration Delay between task rollbacks (ns|us|ms|s|m|h) (default 0s)
--rollback-failure-action string Action on rollback failure ("pause"|"continue") (default "pause")
--rollback-max-failure-ratio float Failure rate to tolerate during a rollback (default 0)
--rollback-monitor duration Duration after each task rollback to monitor for failure (ns|us|ms|s|m|h) (default 5s)
--rollback-order string Rollback order ("start-first"|"stop-first") (default "stop-first")
--rollback-parallelism uint Maximum number of tasks rolled back simultaneously (0 to roll back all at once) (default 1)
--secret secret Specify secrets to expose to the service
--stop-grace-period duration Time to wait before force killing a container (ns|us|ms|s|m|h) (default 10s)
--stop-signal string Signal to stop the container
-t, --tty Allocate a pseudo-terminal
--update-delay duration Delay between updates (ns|us|ms|s|m|h) (default 8s)
--update-failure-action string Action on update failure ("pause"|"continue"|"rollback") (default "pause")
--update-max-failure-ratio float Failure rate to tolerate during an update (default 0)
--update-monitor duration Duration after each task update to monitor for failure (ns|us|ms|s|m|h) (default 5s)
--update-order string Update order ("start-first"|"stop-first") (default "stop-first")
--update-parallelism uint Maximum number of tasks updated simultaneously (0 to update all at once) (default 1)
-u, --user string Username or UID (format: <name>|uid:<group>|gid:<group>)
--with-registry-auth Send registry authentication details to swarm agents
-w, --workdir string Working directory inside the container

```

- Reference: [https://docs.docker.com/engine/reference/commandline/service\\_create/](https://docs.docker.com/engine/reference/commandline/service_create/)

Docker: The Next-Gen of Virtualization



312

# Swarm Service

docker service scale nodejs=5

```
...shop:112018 — Terminal MAC Pro — bash | ... ~ ssh -i k8s_lab ubuntu@13.229.126.99 | ... ~ ssh -i k8s_lab ubuntu@13.229.98.3 ... | ... ssh -i k8s_lab ubuntu@18.136.196.130 ...
ubuntu@ip-10-0-1-104:~$ docker service create --name nodejs -dt labdocker/alpineweb:latest node hello.js
414gu0o15f819jgohbgn994f7
ubuntu@ip-10-0-1-104:~$ docker service ls
ID NAME MODE REPLICAS IMAGE
414gu0o15f81 nodejs replicated 1/1 labdocker/alpineweb:latest
ubuntu@ip-10-0-1-104:~$ docker service ps nodejs
ID NAME IMAGE NODE DESIRED STATE CURRENT STATE ERROR PORTS
h6u61t0564af nodejs.1 labdocker/alpineweb:latest ip-10-0-1-104 Running Running about a minute ago
ubuntu@ip-10-0-1-104:~$ docker service scale nodejs=5
nodejs scaled to 5
overall progress: 5 out of 5 tasks
1/5: running [=====>]
2/5: running [=====>]
3/5: running [=====>]
4/5: running [=====>]
5/5: running [=====>]
verify: Service converged
ubuntu@ip-10-0-1-104:~$ docker service ps nodejs
ID NAME IMAGE NODE DESIRED STATE CURRENT STATE ERROR PORTS
h6u61t0564af nodejs.1 labdocker/alpineweb:latest ip-10-0-1-104 Running Running 2 minutes ago
r5j8bpawetm nodejs.2 labdocker/alpineweb:latest ip-10-0-1-104 Running Running 26 seconds ago
w60ihy8kbs2 nodejs.3 labdocker/alpineweb:latest ip-10-0-1-104 Running Running 16 seconds ago
gjgkorz84q7q nodejs.4 labdocker/alpineweb:latest ip-10-0-1-62 Running Running 16 seconds ago
jubx7aup73b nodejs.5 labdocker/alpineweb:latest ip-10-0-1-62 Running Running 16 seconds ago
ubuntu@ip-10-0-1-104:~$
```

Docker: The Next-Gen of Virtualization



313

# Swarm Service

docker service update -dt --reserve-cpu 1 --limit-cpu 1 nodejs

docker service inspect nodejs | more

```
...shop:112018 — Terminal MAC Pro — bash | ... ~ ssh -i k8s_lab ubuntu@13.229.126.99 | ... ~ ssh -i k8s_lab ubuntu@13.229.98.3 ... | ... ssh -i k8s_lab ubuntu@18.136.196.130 ...
ubuntu@ip-10-0-1-104:~$ docker service update -dt --reserve-cpu 1 --limit-cpu 1 nodejs
nodejs
ubuntu@ip-10-0-1-104:~$ docker service inspect nodejs | more
[
 {
 "ID": "414gu0o15f819jgohbgn994f7",
 "Version": {
 "Index": 44
 },
 "CreatedAt": "2018-10-21T16:25:07.16972436Z",
 "UpdatedAt": "2018-10-21T16:29:50.119830764Z",
 "Spec": {
 "Name": "nodejs",
 "Labels": {},
 "TaskTemplate": {
 "ContainerSpec": {
 "Image": "labdocker/alpineweb:latest@sha256:a7aa70c78120ed126487aecbf49db1054f1ff131858516c91137c3accbb4a333",
 "Args": [
 "node",
 "hello.js"
],
 "Init": false,
 "TTY": true,
 "StopGracePeriod": 10000000000,
 "DNSConfig": {}
 }
 }
 }
]
```

Docker: The Next-Gen of Virtualization



314

# Swarm Service

```

 "Spec": {
 ...
 },
 "UpdateStatus": {
 "State": "updating",
 "StartedAt": "2018-10-21T16:29:50.119815463Z",
 "Message": "update in progress"
 }
 }
ubuntu@ip-10-0-1-104:~$

```

```

...shop_112018 — Terminal MAC Pro — -bash ...: ~ — ssh -i k8s_lab ubuntu@13.229.126.99 ...: ~ — ssh -i k8s_lab ubuntu@13.229.98.3 ...
ubuntu@ip-10-0-1-104:~$ docker service inspect nodejs | grep update
 "Message": "update completed"
ubuntu@ip-10-0-1-104:~$

```

Docker: The Next-Gen of Virtualization 

315

# Swarm Service

**docker service ps nodejs**

```

...shop_112018 — Terminal MAC Pro — -bash ...: ~ — ssh -i k8s_lab ubuntu@13.229.126.99 ...: ~ — ssh -i k8s_lab ubuntu@13.229.98.3 ... — ssh -i k8s_lab ubuntu@18.136.196.130 ...
ubuntu@ip-10-0-1-104:~$ docker service ps nodejs
ID NAME IMAGE NODE DESIRED STATE CURRENT STATE ERROR PORTS
lch1jjspui2 nodejs.1 labdockerc/alpineweb:latest ip-10-0-1-104 Running Running about a minute ago
h6u6lt7o564f _ nodejs.1 labdockerc/alpineweb:latest ip-10-0-1-104 Shutdown Shutdown about a minute ago
ubuntu@ip-10-0-1-104:~$

```

**docker service rm nodejs**

```

...shop_112018 — Terminal MAC Pro — -bash ...: ~ — ssh -i k8s_lab ubuntu@13.229.126.99 ...: ~ — ssh -i k8s_lab ubuntu@13.229.98.3 ... — ssh -i k8s_lab ubuntu@18.136.196.130 ...
ubuntu@ip-10-0-1-104:~$ docker service ps nodejs
ID NAME IMAGE NODE DESIRED STATE CURRENT STATE ERROR PORTS
lch1jjspui2 nodejs.1 labdockerc/alpineweb:latest ip-10-0-1-104 Running Running 2 minutes ago
h6u6lt7o564f _ nodejs.1 labdockerc/alpineweb:latest ip-10-0-1-104 Shutdown Shutdown 2 minutes ago
ubuntu@ip-10-0-1-104:~$ docker service rm nodejs
nodejs
ubuntu@ip-10-0-1-104:~$ docker service ps nodejs
no such service: nodejs
ubuntu@ip-10-0-1-104:~$

```

Docker: The Next-Gen of Virtualization 

316

# Swarm Service

```
Usage: docker service update [OPTIONS] SERVICE
Update a service

Options:
--args command Service command args
--config-add config Add or update a config file on a service
--config-rm list Remove a configuration file
--constraint-add list Add or update a placement constraint
--constraint-rm list Remove a constraint
--container-label-add list Add or update a container label
--container-label-rm list Remove a container label by its key
--credential-spec credential-spec Credential spec for managed service account (Windows only)
-d, --detach Exit immediately instead of waiting for the service to converge (default true)
--dns-add list Add or update a custom DNS server
--dns-option-add list Add or update a DNS option
--dns-option-rm list Remove a DNS option
--dns-rm list Remove a custom DNS server
--dns-search-add list Add or update a custom DNS search domain
--dns-search-rm list Remove a DNS search domain
--endpoint-mode string Endpoint mode (vip or dnsrr)
--entrypoint command Overwrite the default ENTRYPOINT of the image
--env-add list Add or update an environment variable
--env-rm list Remove an environment variable
--force Force update even if no changes require it
--group-add list Add an additional supplementary user group to the container
--group-rm list Remove a previously added supplementary user group from the container
--health-cmd string Command to run to check health
```

- Reference: [https://docs.docker.com/engine/reference/commandline/service\\_update/](https://docs.docker.com/engine/reference/commandline/service_update/)

Docker: The Next-Gen of Virtualization



317

# Swarm Service

```
--health-interval duration Time between running the check (ms|s|m|h)
--health-retries int Consecutive failures needed to report unhealthy
--health-start-period duration Start period for the container to initialize before counting retries towards unstable (ms|s|m|h)
--health-timeout duration Maximum time to allow one check to run (ms|s|m|h)
--help Print usage
--host-add list Add or update a custom host-to-IP mapping (host:ip)
--host-rm list Remove a custom host-to-IP mapping (host:ip)
--hostname string Container hostname
--image string Service image tag
--label-add list Add or update a service label
--label-rm list Remove a label by its key
--limit-cpu decimal Limit CPUs
--limit-memory bytes Limit Memory
--log-driver string Logging driver for service
--log-opt list Logging driver options
--mount-add mount Add or update a mount on a service
--mount-rm path Remove a mount by its target path
--network-add network Add a network
--network-rm list Remove a network
--no-healthcheck Disable using container-specified HEALTHCHECK
--no-resolve-image Do not query the registry to resolve image digest and supported platforms
--placement-pref-add pref Add a placement preference
--placement-pref-rm pref Remove a placement preference
--publish-add port Add or update a published port
--publish-rm port Remove a published port by its target port
-q, --quiet Suppress progress output
--read-only Mount the container's root filesystem as read only
--replicas uint Number of tasks
--reserve-cpu decimal Reserve CPUs
--reserve-memory bytes Reserve Memory
```

- Reference: [https://docs.docker.com/engine/reference/commandline/service\\_update/](https://docs.docker.com/engine/reference/commandline/service_update/)

Docker: The Next-Gen of Virtualization



318

# Swarm Service

```
--read-only Mount the container's root filesystem as read only
--replicas uint Number of tasks
--reserve-cpu decimal Reserve CPUs
--reserve-memory bytes Reserve Memory
--restart-condition string Restart when condition is met ("none"|"on-failure"|"any")
--restart-delay duration Delay between restart attempts (ns|us|ms|s|m|h)
--restart-max-attempts uint Maximum number of restarts before giving up
--restart-window duration Window used to evaluate a restart policy (ns|us|ms|s|m|h)
--rollback Rollback to previous specification
--rollback-delay duration Delay between each task rollback (ns|us|ms|s|m|h)
--rollback-failure-action string Action on rollback failure ("pause"|"continue")
--rollback-max-failure-ratio float Failure rate to tolerate during a rollback
--rollback-monitor duration Duration after each task rollback to monitor for failure (ns|us|ms|s|m|h)
--rollback-order string Rollback order ("start-first"|"stop-first")
--rollback-parallelism uint Maximum number of tasks rolled back simultaneously (0 to roll back all at once)
--secret-add secret Add or update a secret on a service
--secret-rm list Remove a secret
--stop-grace-period duration Time to wait before force killing a container (ns|us|ms|s|m|h)
--stop-signal string Signal to stop the container
-t, --tty Allocate a pseudo-TTY
--update-delay duration Delay between updates (ns|us|ms|s|m|h)
--update-failure-action string Action on update failure ("pause"|"continue"|"rollback")
--update-max-failure-ratio float Failure rate to tolerate during an update
--update-monitor duration Duration after each task update to monitor for failure (ns|us|ms|s|m|h)
--update-order string Update order ("start-first"|"stop-first")
--update-number-of-tasks uint Maximum number of tasks updated simultaneously (0 to update all at once)
-u, --user string Username or UID (format: <name>[<uid>:<group>|<gid>])
--with-registry-auth string Send registry authentication details to swarm agents
-w, --workdir string Working directory inside the container
```

- Reference: [https://docs.docker.com/engine/reference/commandline/service\\_update/](https://docs.docker.com/engine/reference/commandline/service_update/)

Docker: The Next-Gen of Virtualization



319

# Workshop: Swarm Service

- สร้าง service nodejs เพื่อเริ่มทำงานบน Swarm

```
docker service create -dt --name nodejs \
labdocker/alpineweb:latest node hello.js
```

- ตรวจสอบสถานะของ Service หลังจากการสร้าง

```
docker service ls
docker service ps nodejs
```

```
...shop_112016 — Terminal MAC Pro — bash | ...:~— ssh -i k8s_lab ubuntu@13.229.126.99 | ...:~— ssh -i k8s_lab ubuntu@13.229.98.3 ... | ...:~— ssh -i k8s_lab ubuntu@18.136.196.130 ...
ubuntu@ip-10-0-1-104:~$ docker service create -dt --name nodejs \
> labdocker/alpineweb:latest node hello.js

sm7d0yzuedh7kpv4o3zvbtvy1
ubuntu@ip-10-0-1-104:~$
ubuntu@ip-10-0-1-104:~$ docker service ls
ID NAME MODE REPLICAS IMAGE
sm7d0yzuedh7 nodejs replicated 1/1 labdocker/alpineweb:latest
ubuntu@ip-10-0-1-104:~$ docker service ps nodejs
ID NAME IMAGE NODE DESIRED STATE CURRENT STATE ERROR PORTS
pg53791w4q8a nodejs.1 labdocker/alpineweb:latest ip-10-0-1-104 Running Running 7 seconds ago
ubuntu@ip-10-0-1-104:~$
```

Docker: The Next-Gen of Virtualization



320

# Orchestrator Assignment

- Swarm สามารถควบคุมการจ่ายงานให้จาก manager ไปยัง worker ได้ หลากหลายเทคนิค เพื่อตอบสนองความต้องการของผู้ใช้งาน
- Docker stack deploy (Compose version 3.4)
  - Replicated (Specific number of container by manual)
  - Global (1 container per node in swarm)
- Assign by default node constrain (-constrain)
  - node.id
  - node.hostname
  - node.role
  - node.labels (user define label)
  - engine.labels
- Assign by service placement preference (--placement-pref)
  - Spread on node.label (user define label)

Docker: The Next-Gen of Virtualization



321

# Orchestrator Assignment

- node constrain
  - Running with constrain with node
  - node.id

```
ubuntu@ip-10-0-1-104:~$ docker node ls
ID HOSTNAME STATUS AVAILABILITY MANAGER STATUS ENGINE VERSION
4pp6popaaa5co8zxew8w0pxfd6 ip-10-0-1-42 Ready Active 18.06.1-ce
3tnvityyohbekgvgvsh1ijovy * ip-10-0-1-104 Ready Active Leader 18.06.1-ce
uaa8afjfd4d1nsks0zer80w0zru ip-10-0-1-163 Ready Active 18.06.1-ce
ubuntu@ip-10-0-1-104:~$ docker service create --dt --constraint 'node.id==3tnvityyohbekgvgvsh1ijovy' \
> --name nodejs --replicas=5 labdocker/alpineweb:latest node hello.js
v52nswafowr2xq6lb9rpzu7r
ubuntu@ip-10-0-1-104:~$ docker service ps nodejs
ID NAME IMAGE NODE DESIRED STATE CURRENT STATE ERROR PORTS
18 nodejs.1 labdocker/alpineweb:latest ip-10-0-1-104 Running Running less than a second ago
párlexixi88 nodejs.2 labdocker/alpineweb:latest ip-10-0-1-104 Running Running less than a second ago
17z39prijb5f* nodejs.3 labdocker/alpineweb:latest ip-10-0-1-104 Running Running less than a second ago
oadf44nbhd5d* nodejs.4 labdocker/alpineweb:latest ip-10-0-1-104 Running Running less than a second ago
12g4d3bkkc70 nodejs.5 labdocker/alpineweb:latest ip-10-0-1-104 Running Running less than a second ago
```

Docker: The Next-Gen of Virtualization



322

# Orchestrator Assignment

- Remove constrain/Add constrain
  - node.hostname

```
orkshop_112018 — Terminal MAC Pro — bash 04: ~ — ssh -i k8s_lab ubuntu@13.229.126.99 63: ~ — ssh -i k8s_lab ubuntu@13.229.98.3 ... — ssh -i k8s_lab ubuntu@18.136.196.130 ...
ubuntu@ip-10-0-1-104:~$ docker service update --dt --constraint-rm 'node.id==3tnvityohbekgvgvbslijjovy' --constraint-add 'node.hostname==ip-10-0-1-104' nodejs
nodejs
(node)ubuntu@ip-10-0-1-104:~$ docker service ps nodejs
ID NAME IMAGE NODE DESIRED STATE CURRENT STATE ERROR PORTS
nodejs.1 nodejs.1 labdocke/alpineweb:latest ip-10-0-1-104 Running Running less than a second ago
nodejs.2 nodejs.2 labdocke/alpineweb:latest ip-10-0-1-104 Running Running less than a second ago
nodejs.3 nodejs.3 labdocke/alpineweb:latest ip-10-0-1-104 Ready Ready 2 seconds ago
nodejs.4 nodejs.4 labdocke/alpineweb:latest ip-10-0-1-104 Running Running 3 minutes ago
nodejs.5 nodejs.5 labdocke/alpineweb:latest ip-10-0-1-104 Running Running 3 minutes ago
(node)ubuntu@ip-10-0-1-104:~$ docker service inspect nodejs | grep update
Message: "update completed"
(node)ubuntu@ip-10-0-1-104:~$ docker service ps nodejs
ID NAME IMAGE NODE DESIRED STATE CURRENT STATE ERROR PORTS
nodejs.1 nodejs.1 labdocke/alpineweb:latest ip-10-0-1-104 Running Running 24 seconds ago
nodejs.2 nodejs.2 labdocke/alpineweb:latest ip-10-0-1-104 Shutdown Shutdown 24 seconds ago
nodejs.3 nodejs.3 labdocke/alpineweb:latest ip-10-0-1-104 Running Running about 4 minutes ago
nodejs.4 nodejs.4 labdocke/alpineweb:latest ip-10-0-1-104 Shutdown Shutdown 48 seconds ago
nodejs.5 nodejs.5 labdocke/alpineweb:latest ip-10-0-1-104 Shutdown Shutdown 48 seconds ago
(node)ubuntu@ip-10-0-1-104:~$
```

Docker: The Next-Gen of Virtualization



323

## Workshop: node constrain

- สร้าง service nodejs เพื่อเริ่มทำงานบน Swarm

```
docker service create -dt --constraint
'node.id==6bei4mbj5pd7yduyhp375b7z4' --name nodejs --
replicas=5 labdocke/alpineweb:latest node hello.js
```

```
orkshop_112018 — Terminal MAC Pro — bash 04: ~ — ssh -i k8s_lab ubuntu@13.229.126.99 63: ~ — ssh -i k8s_lab ubuntu@13.229.98.3 ... — ssh -i k8s_lab ubuntu@18.136.196.130 ...
ubuntu@ip-10-0-1-104:~$ docker node ls
ID HOSTNAME STATUS AVAILABILITY MANAGER STATUS ENGINE VERSION
4g6pqopaae5c0zx8w6pxfd6 ip-10-0-1-62 Ready Active 18.06.1-ce
3tnvityohbekgvgvbslijjovy * ip-10-0-1-104 Ready Active Leader 18.06.1-ce
uaa8jff4d1msqwr80y02ru ip-10-0-1-163 Ready Active 18.06.1-ce
ubuntu@ip-10-0-1-104:~$ docker service create -dt --constraint 'node.id==3tnvityohbekgvgvbslijjovy' \
> --name nodejs --replicas=5 labdocke/alpineweb:latest node hello.js
v52nswafourz2xq61bprpu0r
ubuntu@ip-10-0-1-104:~$ docker service ps nodejs
ID NAME IMAGE NODE DESIRED STATE CURRENT STATE ERROR PORTS
TS
nodejs.1 nodejs.1 labdocke/alpineweb:latest ip-10-0-1-104 Running Running less than a second ago
nodejs.2 nodejs.2 labdocke/alpineweb:latest ip-10-0-1-104 Running Running less than a second ago
nodejs.3 nodejs.3 labdocke/alpineweb:latest ip-10-0-1-104 Running Running less than a second ago
nodejs.4 nodejs.4 labdocke/alpineweb:latest ip-10-0-1-104 Running Running less than a second ago
nodejs.5 nodejs.5 labdocke/alpineweb:latest ip-10-0-1-104 Running Running less than a second ago
```

Docker: The Next-Gen of Virtualization



324

# Orchestrator Assignment

- node constrain (custom label)

```
docker node update --label-add 'storage=sas' swarm-mng
docker node update --label-add 'storage=nvdimm' swarm-node1
docker node update --label-add 'storage=sata' swarm-node2
```

```
docker node inspect swarm-mng | grep storage
docker node inspect swarm-node1 | grep storage
docker node inspect swarm-node2 | grep storage
```

```
ubuntu@ip-10-0-1-104:~$ docker node ls
ID HOSTNAME STATUS AVAILABILITY MANAGER STATUS ENGINE VERSION
4g6engqas5cc08zxe8edcoxfd6 ip-10-0-1-2 Ready Active
3t611qghnkbygqv811j0... * ip-10-0-1-104 Ready Active
ubn0j5jd4klnkwqr800y2ru ip-10-0-1-163 Ready Active
ubuntu@ip-10-0-1-104:~$ docker node update --label-add 'storage=sas' ip-10-0-1-104
ip-10-0-1-104
ubuntu@ip-10-0-1-104:~$ docker node update --label-add 'storage=nvdimm' ip-10-0-1-163
ip-10-0-1-163
ubuntu@ip-10-0-1-104:~$ docker node update --label-add 'storage=sata' ip-10-0-1-62
ip-10-0-1-62
ubuntu@ip-10-0-1-104:~$ docker node inspect ip-10-0-1-104|grep storage
"storage": "sas"
ubuntu@ip-10-0-1-104:~$ docker node inspect ip-10-0-1-163|grep storage
"storage": "nvdimm"
ubuntu@ip-10-0-1-104:~$ docker node inspect ip-10-0-1-62|grep storage
"storage": "sata"
ubuntu@ip-10-0-1-104:~$
```

Docker: The Next-Gen of Virtualization



325

# Orchestrator Assignment

- custom label

```
docker service create --name xxx --add-label 'xxx=xxx'
```

```
docker service create -dt --constraint 'node.labels.storage==sas' --
name nodejs --replicas=5 labdockeralpineweb:latest node hello.js
```

```
ubuntu@ip-10-0-1-104:~$ docker service create -dt --constraint 'node.labels.storage==sas' --name nodejs --replicas=5 labdockeralpineweb:latest node hello.js
p16f8569yokx23bgm312txo91
ubuntu@ip-10-0-1-104:~$ docker service ps nodejs
ID IMAGE NODE DESIRED STATE CURRENT STATE ERROR PORTS
vkh4rygb02z1 nodejs.1 labdockeralpineweb:latest ip-10-0-1-104 Running Running 7 seconds ago
7hkco9pn2z3q nodejs.2 labdockeralpineweb:latest ip-10-0-1-104 Running Running 6 seconds ago
s1lcqv1dn1nt nodejs.3 labdockeralpineweb:latest ip-10-0-1-104 Running Running 6 seconds ago
3axpawaukgmt nodejs.4 labdockeralpineweb:latest ip-10-0-1-104 Running Running 7 seconds ago
j2cwlwj4qmcbu nodejs.5 labdockeralpineweb:latest ip-10-0-1-104 Running Running 7 seconds ago
ubuntu@ip-10-0-1-104:~$
```

Docker: The Next-Gen of Virtualization



326

## Workshop: custom label

- สร้าง docker service โดยระบุ constraint จาก custom label ที่สร้างขึ้น

```
docker service create --dt --constraint
'node.labels.storage==sas' --name nodejs --replicas=5
labdocker/alpineweb:latest node hello.js
```

```
...orkshop_112018 — Terminal MAC Pro - bash ...04: ~ — ssh -i k8s_lab ubuntu@13.229.126.99 ...63: ~ — ssh -i k8s_lab ubuntu@13.229.98.3 ... ~ — ssh -i k8s_lab ubuntu@18.136.196.1
ubuntu@ip-10-0-1-104:~$ docker service create --dt --constraint 'node.labels.storage==sas' --name nodejs --replicas=5 labdocker/alpineweb:latest node hello.js
p16f8558yokz3bpg3l2txo91
ubuntu@ip-10-0-1-104:~$ docker service ps nodejs
ID NAME IMAGE
vkh4tygb2z1 nodejs.1 labdocker/alpineweb:latest
7nko9pzb03a nodejs.2 labdocker/alpineweb:latest
s1cqvldinm nodejs.3 labdocker/alpineweb:latest
3axgwaukmgf nodejs.4 labdocker/alpineweb:latest
izcwjlegbnu nodejs.5 labdocker/alpineweb:latest
ubuntu@ip-10-0-1-104:~$ docker node ls
ID HOSTNAME
4q6ppqgaae5co8zxe86pxfd6 ip-10-0-1-62
3tnvlyyyhbekgyvbljovy * ip-10-0-1-104
ua8ajfd4d1nskzqr80yo2ru ip-10-0-1-163
ubuntu@ip-10-0-1-104:~$
```

Docker: The Next-Gen of Virtualization



327

## Orchestrator Assignment

- service placement preference (--placement-pref)

```
docker node update --label-add 'physical=DELLPE820' swarm-mng
docker node update --label-add 'physical=DELLPE820' swarm-node1
docker node update --label-add 'physical=HP' swarm-node2
```

```
docker node inspect swarm-mng|grep physical
docker node inspect swarm-node1|grep physical
docker node inspect swarm-node2|grep physical
```

```
...orkshop_112018 — Terminal MAC Pro - bash ...04: ~ — ssh -i k8s_lab ubuntu@13.229.126.99 ...63: ~ — ssh -i k8s_lab ubuntu@13.229.98.3 ...
ubuntu@ip-10-0-1-104:~$ docker node ls
ID HOSTNAME
4q6ppqgaae5co8zxe86pxfd6 ip-10-0-1-62
3tnvlyyyhbekgyvbljovy * ip-10-0-1-104
ua8ajfd4d1nskzqr80yo2ru ip-10-0-1-163
ubuntu@ip-10-0-1-104:~$ docker node update --label-add 'physical=DELLPE820'
ip-10-0-1-104
ubuntu@ip-10-0-1-104:~$ docker node update --label-add 'physical=DELLPE820' ip-10-0-1-163
ip-10-0-1-163
ubuntu@ip-10-0-1-104:~$ docker node update --label-add 'physical=HP' ip-10-0-1-62
ip-10-0-1-62
ubuntu@ip-10-0-1-104:~$ docker node inspect ip-10-0-1-104|grep physical
"physical": "DELLPE820"
ubuntu@ip-10-0-1-104:~$ docker node inspect ip-10-0-1-163|grep physical
"physical": "DELLPE820",
ubuntu@ip-10-0-1-104:~$ docker node inspect ip-10-0-1-62|grep physical
"physical": "HP",
ubuntu@ip-10-0-1-104:~$
```

Docker: The Next-Gen of Virtualization



328

# Orchestrator Assignment

- service placement preference

```
docker service create -dt --placement-pref 'spread=node.labels.physical' \
--name nodejs --replicas=10 labdocker/alpineweb:latest node hello.js
```

```
...orkshop_112018 — Terminal MAC Pro — bash ...04: ~ — ssh -i k8s_lab ubuntu@13.229.126.99 ...63: ~ — ssh -i k8s_lab ubuntu@13.229.98.3~ — ssh -i k8s_lab ubuntu@18.136
ubuntu@ip-10-0-1-104:~$ docker service create -dt --placement-pref 'spread=node.labels.physical' \
> --name nodejs --replicas=10 labdocker/alpineweb:latest node hello.js
n36z17qvwkj2euaus95qxcxwpt
ubuntu@ip-10-0-1-104:~$ docker service ps nodejs
ID NAME IMAGE NODE DESIRED STATE CURRENT STATE ERROR PORTS
zlp6rg0ip7o8 nodejs.1 labdocker/alpineweb:latest ip-10-0-1-62 Running Running 2 seconds ago
pygyve34w7b1 nodejs.2 labdocker/alpineweb:latest ip-10-0-1-104 Running Running 2 seconds ago
w4vvxv0f1zix nodejs.3 labdocker/alpineweb:latest ip-10-0-1-62 Running Running 2 seconds ago
w5isyp10p61 nodejs.4 labdocker/alpineweb:latest ip-10-0-1-62 Running Running 2 seconds ago
tf2bmq7bj1q nodejs.5 labdocker/alpineweb:latest ip-10-0-1-62 Running Running 2 seconds ago
6f0tjy1yyry3 nodejs.6 labdocker/alpineweb:latest ip-10-0-1-62 Running Running 2 seconds ago
73idk3fa30pc nodejs.7 labdocker/alpineweb:latest ip-10-0-1-62 Running Running 2 seconds ago
kq2bqw9q4o9 nodejs.8 labdocker/alpineweb:latest ip-10-0-1-163 Running Running 2 seconds ago
nlziswaf4p92 nodejs.9 labdocker/alpineweb:latest ip-10-0-1-163 Running Running 2 seconds ago
ozn25d53dexf nodejs.10 labdocker/alpineweb:latest ip-10-0-1-104 Running Running 2 seconds ago
ubuntu@ip-10-0-1-104:~$ docker service rm nodejs
nodejs
ubuntu@ip-10-0-1-104:~$
```



Docker: The Next-Gen of Virtualization

329

# Workshop: Orchestrator

- สร้าง docker service โดยระบุใน swarm กระจายภายใต้ label

```
docker service create -dt --placement-pref
'spread=node.labels.physical' \
--name nodejs --replicas=10 labdocker/alpineweb:latest node hello.js
```

```
...orkshop_112018 — Terminal MAC Pro — bash ...04: ~ — ssh -i k8s_lab ubuntu@13.229.126.99 ...63: ~ — ssh -i k8s_lab ubuntu@13.229.98.3~ — ssh -i k8s_lab ubuntu@18.136
ubuntu@ip-10-0-1-104:~$ docker service create -dt --placement-pref 'spread=node.labels.physical' \
> --name nodejs --replicas=10 labdocker/alpineweb:latest node hello.js
n36z17qvwkj2euaus95qxcxwpt
ubuntu@ip-10-0-1-104:~$ docker service ps nodejs
ID NAME IMAGE NODE DESIRED STATE CURRENT STATE ERROR PORTS
zlp6rg0ip7o8 nodejs.1 labdocker/alpineweb:latest ip-10-0-1-62 Running Running 2 seconds ago
pygyve34w7b1 nodejs.2 labdocker/alpineweb:latest ip-10-0-1-104 Running Running 2 seconds ago
w4vvxv0f1zix nodejs.3 labdocker/alpineweb:latest ip-10-0-1-62 Running Running 2 seconds ago
w5isyp10p61 nodejs.4 labdocker/alpineweb:latest ip-10-0-1-62 Running Running 2 seconds ago
tf2bmq7bj1q nodejs.5 labdocker/alpineweb:latest ip-10-0-1-62 Running Running 2 seconds ago
6f0tjy1yyry3 nodejs.6 labdocker/alpineweb:latest ip-10-0-1-62 Running Running 2 seconds ago
73idk3fa30pc nodejs.7 labdocker/alpineweb:latest ip-10-0-1-62 Running Running 2 seconds ago
kq2bqw9q4o9 nodejs.8 labdocker/alpineweb:latest ip-10-0-1-163 Running Running 2 seconds ago
nlziswaf4p92 nodejs.9 labdocker/alpineweb:latest ip-10-0-1-163 Running Running 2 seconds ago
ozn25d53dexf nodejs.10 labdocker/alpineweb:latest ip-10-0-1-104 Running Running 2 seconds ago
ubuntu@ip-10-0-1-104:~$ docker service rm nodejs
nodejs
ubuntu@ip-10-0-1-104:~$
```



Docker: The Next-Gen of Virtualization

330

# Config and Secret

- Make secret data and configuration great again !
- การทำงานของ microservice ที่รวมกันเป็น application stack จะมี ส่วนของข้อมูลคอนฟิกต่างๆที่จำเป็นต้องใช้งานในระบบ เช่น
  - Root password of database
  - Environment variable
  - Custom variable
  - Path of mount volume data
  - Etc
- docker config จะใช้เพื่อจัดเก็บข้อมูลคอนฟิกต่างๆของ container
- docker secret ใช้จัดเก็บข้อมูลที่เป็นความลับโดยการเข้ารหัสข้อมูล

Docker: The Next-Gen of Virtualization



331

# Config and Secret

- config can add/update with service anytime
  - Linux container: /<config name>
  - Windows container: c:\<config name>

```
echo "config value" | docker config create<config name> -
```

```
docker config create<config name><config file>
```

```
docker config rm <config name>
```

- Apply config to service

```
docker service<create --config /update --config-add> \
source=<config name>,target=<path to map config>
```

```
docker service<rm/ --config-rm><config name>
```

Docker: The Next-Gen of Virtualization



332

# Config and Secret

- secret is all same as configure except that secret is encrypt all data (encode base64)
  - Linux container: /run/secrets/<secret name>
  - Windows container: c:\ProgramData\ Docker\Secret

```
echo "config value" | docker secret create <secret name> -
```

```
docker secret create <secret name> <secret file>
```

```
docker secret rm <secret name>
```

- Apply config to service

```
docker service <create --secret/update --secret-add> <secret name>
```

```
docker service <rm /update --secret-rm> source=<secret name>
```

Docker: The Next-Gen of Virtualization



333

# Workshop: Config and Secret

- สร้าง nginx service เพื่อรองรับการให้บริการ https (TLS 1.2) ผ่าน config and secret

```
docker config create nginx.conf /Share_DockerToolbox/nginx.conf
```

```
docker secret create labdocker.com.crt \
/Share_DockerToolbox/labdocker.com.crt
```

```
docker secret create labdocker.com.key \
/Share_DockerToolbox/labdocker.com.key
```

```
...orkshop_112018 — Terminal MAC Pro — -bash ...04: ~ — ssh -i k8s_lab ubuntu@13.229.126.99 ...63: ~ — ssh -i k8s_lab ubuntu@13.229.98.3 ...
[ubuntu@ip-10-0-1-104:~$ docker config create nginx.conf ~/docker_workshop_112018/Workshop-2-4-Swarm/nginx.conf
bni17w6tmbks3911ri3ryvvxz
[ubuntu@ip-10-0-1-104:~$ docker config ls
ID NAME CREATED UPDATED
bni17w6tmbks3911ri3ryvvxz nginx.conf 3 seconds ago 3 seconds ago
ubuntu@ip-10-0-1-104:~$ docker secret create labdocker.com.crt ~/docker_workshop_112018/Workshop-2-4-Swarm/labdocker.com.crt
pt2dhhjirroqd7gb6ig9bv62n
ubuntu@ip-10-0-1-104:~$ docker secret create labdocker.com.key ~/docker_workshop_112018/Workshop-2-4-Swarm/labdocker.com.key
nu1ifusm2zb1815112s8peonb
[ubuntu@ip-10-0-1-104:~$ docker secret ls
ID NAME DRIVER CREATED UPDATED
pt2dhhjirroqd7gb6ig9bv62n labdocker.com.crt
nu1ifusm2zb1815112s8peonb labdocker.com.key
[ubuntu@ip-10-0-1-104:~$ docker service create -dt \

```

Docker: The Next-Gen of Virtualization



334

# Workshop: Config and Secret

- nginx.conf

```
server {
 listen 443 ssl;
 server_name localhost;
 ssl_certificate /run/secrets/labdocker.com.crt;
 ssl_certificate_key /run/secrets/labdocker.com.key;
 location / {
 root /usr/share/nginx/html;
 index index.html index.htm;
 }
}
```

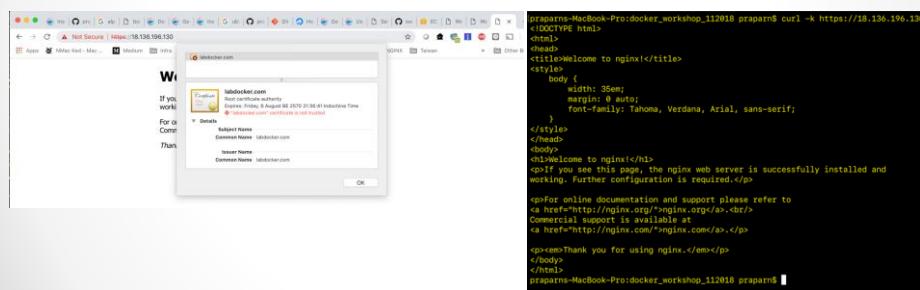
Docker: The Next-Gen of Virtualization



335

# Workshop: Config and Secret

```
docker service -dt create \
--name nginx \
--secret labdocker.com.key \
--secret labdocker.com.crt \
--config source=nginx.conf,target=/etc/nginx/nginx.conf \
-p 443:443 labdocker/nginx:latest
```



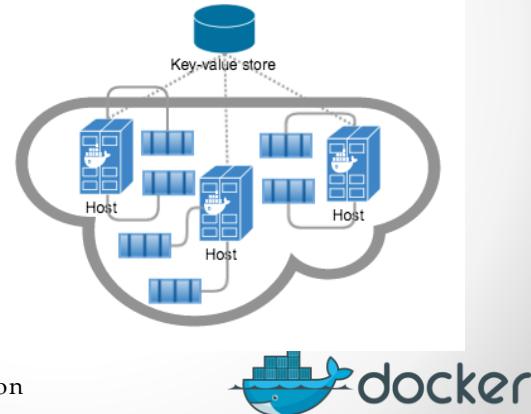
Docker: The Next-Gen of Virtualization



336

# Overlay and Ingress Network

- เพื่อให้ทุกๆ node ภายใน swarm cluster สามารถมองเห็นกันได้ docker ได้เตรียมระบบ network แบบ overlay
- การทำงานของ overlay จะอาศัยกลไกของ key value store เป็นหลักในการตรวจสอบ (Discovery / Network status / IP Address etc)
- Default swarm จะสร้าง overlay network ชื่อ “ingress” เพื่อรับการใช้งาน service

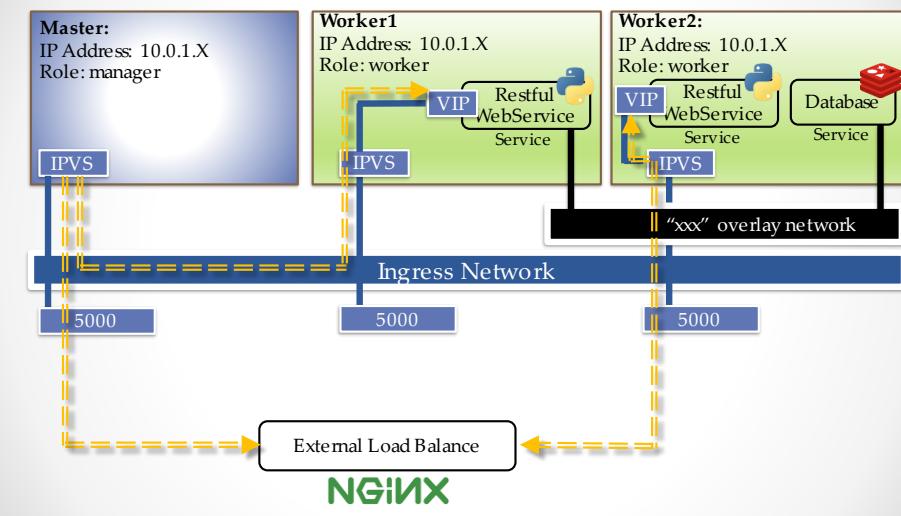


Docker: The Next-Gen of Virtualization



337

# Overlay and Ingress Network

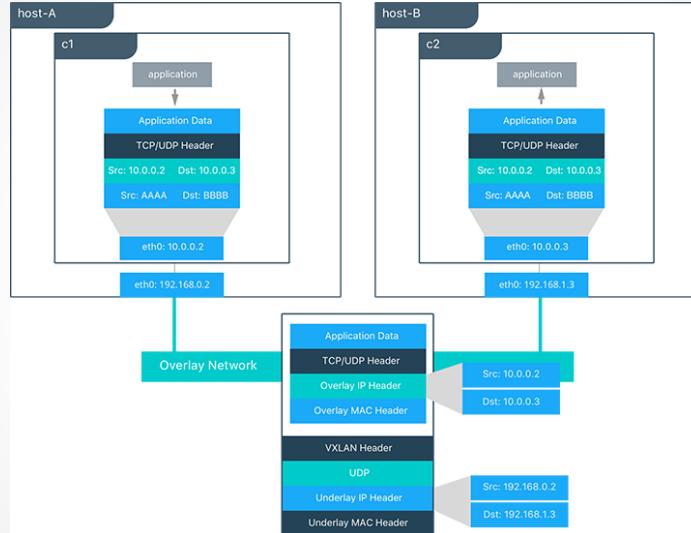


Docker: The Next-Gen of Virtualization



338

# Overlay and Ingress Network



Docker: The Next-Gen of Virtualization



339

## Workshop: Overlay and Ingress

- สร้าง overlay network บน swarm manager

```
docker network create --driver overlay --
subnet=192.168.100.0/24 swarmnet
```

```
...orkshop.112018 — Terminal MAC Pro — bash .. 04: ~ — ssh -i k8s_lab ubuntu@13.229.126.99 .. 63: ~ — ssh
ubuntu@ip-10-0-1-104:~$ docker network create --driver overlay --subnet=192.168.100.0/24 swarmnet
lwxhyfrildn83atwsqxd9ior
ubuntu@ip-10-0-1-104:~$ docker network ls
NETWORK ID NAME DRIVER SCOPE
cdb776996466 bridge bridge local
7f907ae7a4c9 docker_gwbridge bridge local
d44f5de24dd1 host host local
hd63tliqe1os ingress overlay swarm
alef67647608 none null local
lwxhyfrildn8 swarmnet overlay swarm
ubuntu@ip-10-0-1-104:~$
```

Docker: The Next-Gen of Virtualization



340

# Workshop: Overlay and Ingress

- สร้าง new service เพื่อใช้งาน swarm network

```
docker service -dt create --name nodejs --replicas=2 --
network-add swarmnet -p 3000:3000
labdocker/alpineweb:latest node hello.js
```

```
...orkshop_112018 — Terminal MAC Pro — bash 04: ~ ssh -i k8s_lab ubuntu@13.229.98.99 ... 63: ~ ssh -i k8s_lab ubuntu@13.229.98.3 ... 70: ~ ssh -i k8s_lab ubuntu@18.136.159 ...
ubuntu@ip-10-0-1-104:~$ docker service create --name nodejs \
> --replicas=2 --network swarmnet -p 3000:3000 \
> labdocker/alpineweb:latest node hello.js
42w42@vcxgu0k81qmslar6
ubuntu@ip-10-0-1-104:~$ docker service ps nodejs
ID NAME IMAGE
t1022pjovfjs.1 labdocker/alpineweb:latest
51l1bdb8z00y nodejs.2 labdocker/alpineweb:latest
ubuntu@ip-10-0-1-104:~$
```

```
ubuntu@ip-10-0-1-104:~$ docker ps
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
bfc3f7a2e264 labdocker/alpineweb:latest "node hello.js node ..." About a minute ago Up About a minute 3000/tcp nodejs.1.t1022pjovfjsxgtfficvr9mf
ubuntu@ip-10-0-1-104:~$ docker inspect nodejs.1.t1022pjovfjsxgtfficvr9mf |grep IPAddress
 "SecondaryIPAddresses": null,
 "IPAddress": "",
 "IPAddress": "10.255.0.8",
 "IPAddress": "192.168.100.6",
ubuntu@ip-10-0-1-104:~$
```

```
ubuntu@ip-10-0-1-103:~$ docker ps
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
cf75d3a7074b labdocker/alpineweb:latest "node hello.js node ..." 2 minutes ago Up 2 minutes 3000/tcp nodejs.2.51t1bdb8z00yaq761ac6ozpy0
ubuntu@ip-10-0-1-103:~$ docker inspect nodejs.2.51t1bdb8z00yaq761ac6ozpy0 |grep IPAddress
 "SecondaryIPAddresses": null,
 "IPAddress": "",
 "IPAddress": "10.255.0.9",
 "IPAddress": "192.168.100.7",
ubuntu@ip-10-0-1-103:~$
```

Docker: The Next-Gen of Virtualization 

341

# Workshop: Overlay and Ingress

- ตรวจสอบ IP Address ของ service และทดสอบ Ping ชั้ม node

```
docker service inspect nodejs|more
```

```
...orkshop_112018 — Terminal MAC Pro — bash 04: ~ ssh -i k8s_lab ubuntu@13.229.126.99 ... 63: ~ ssh -i k8s_lab ubuntu@13.229.98.3 ... 70: ~ ssh -i k8s_lab ubuntu@18.136.159 ...
[{"ID": "4ze1b0k4vcxgu0k81qmslar6", "Version": {"Index": 233}, "Created": "2018-10-21T17:12:30.830228563Z", "Updated": "2018-10-21T17:12:30.831796988Z", "Spec": {"Name": "nodejs", "Labels": {}, "TaskTemplate": {"ContainerSpec": {"Image": "labdocker/alpineweb:latest@sha256:a7aa70c78120ed126487aecbf49db1054ff1ff13185881dc91137c3acbb4a333", "Args": ["node", "hello.js"], "Init": false, "Restart": true, "StopGracePeriod": 10000000000, "DNSConfig": {}, "Isolation": "default"}}, "VirtualIPs": [{"NetworkID": "hd63211ng18s27mu9w4ilkui", "Address": "10.255.0.7/16"}, {"NetworkID": "1wvhyfrldm83atesqdb9ior", "Address": "192.168.100.5/24"}]}]
```

Docker: The Next-Gen of Virtualization 

342

# Workshop: Overlay and Ingress

- ตรวจสอบ IP Address ของ service และทดสอบ Ping ข้าม node

```
ubuntu@ip-10-0-1-104:~$ docker ps
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
bf3f7a2e264 labdocker/alpineweb:latest "node hello.js node -" 4 minutes ago Up 4 minutes 3000/tcp nodejs.1.t1022pjovfjsrxgtfficvr9mf

ubuntu@ip-10-0-1-104:~$ docker exec -it nodejs.1.t1022pjovfjsrxgtfficvr9mf ping 192.168.100.5
PING 192.168.100.5 (192.168.100.5): 56 data bytes
64 bytes from 192.168.100.5: seq=0 ttl=64 time=0.116 ms
64 bytes from 192.168.100.5: seq=1 ttl=64 time=0.076 ms
```
--- 192.168.100.5 ping statistics ---
2 packets transmitted, 2 packets received, 0% packet loss
round-trip min/avg/max = 0.076/0.094/0.116 ms

ubuntu@ip-10-0-1-104:~$ docker exec -it nodejs.1.t1022pjovfjsrxgtfficvr9mf ping 192.168.100.7
PING 192.168.100.7 (192.168.100.7): 56 data bytes
64 bytes from 192.168.100.7: seq=0 ttl=64 time=0.114 ms
64 bytes from 192.168.100.7: seq=1 ttl=64 time=0.226 ms
```
--- 192.168.100.7 ping statistics ---
2 packets transmitted, 2 packets received, 0% packet loss
round-trip min/avg/max = 0.226/0.678/1.114 ms

ubuntu@ip-10-0-1-104:~$
```

```
ubuntu@ip-10-0-1-163:~$ docker ps
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
cf7503a7074b labdocker/alpineweb:latest "node hello.js node -" 7 minutes ago Up 7 minutes 3000/tcp nodejs.2.51tldbd8z00yaq761ac60zpy6

ubuntu@ip-10-0-1-163:~$ docker exec -it nodejs.2.51tldbd8z00yaq761ac60zpy6 ping 192.168.100.6
PING 192.168.100.6 (192.168.100.6): 56 data bytes
64 bytes from 192.168.100.6: seq=0 ttl=64 time=0.180 ms
64 bytes from 192.168.100.6: seq=1 ttl=64 time=0.062 ms
```
--- 192.168.100.6 ping statistics ---
2 packets transmitted, 2 packets received, 0% packet loss
round-trip min/avg/max = 0.062/0.081/0.180 ms

ubuntu@ip-10-0-1-163:~$ docker exec -it nodejs.2.51tldbd8z00yaq761ac60zpy6 ping 192.168.100.6
PING 192.168.100.6 (192.168.100.6): 56 data bytes
64 bytes from 192.168.100.6: seq=0 ttl=64 time=0.244 ms
64 bytes from 192.168.100.6: seq=1 ttl=64 time=0.237 ms
```
--- 192.168.100.6 ping statistics ---
2 packets transmitted, 2 packets received, 0% packet loss
round-trip min/avg/max = 0.237/0.240/0.244 ms

ubuntu@ip-10-0-1-163:~$
```

Docker: The Next-Gen of Virtualization



343

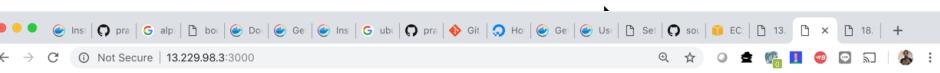
# Workshop: Overlay and Ingress

- เรียกใช้บริการผ่าน http port ด้วย browser/curl

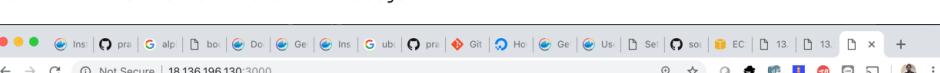
```
praparns-MacBook-Pro:docker_workshop_112018 praparn$ curl http://18.136.196.130:3000
Hello World Container Docker for Nodejs
praparns-MacBook-Pro:docker_workshop_112018 praparn$ curl http://13.229.126.99:3000
Hello World Container Docker for Nodejs
praparns-MacBook-Pro:docker_workshop_112018 praparn$ curl http://13.229.98.3:3000
Hello World Container Docker for Nodejs
praparns-MacBook-Pro:docker_workshop_112018 praparn$
```



Hello World Container Docker for Nodejs



Hello World Container Docker for Nodejs



Hello World Container Docker for Nodejs

Docker: The Next-Gen of Virtualization



344

# HA Manager Role

- Add redundancy swarm manager in swarm cluster
  - ท่านำ change role node จาก worker ➔ manager

```
docker node update --role manager <node id>
```

```
...orkshop_112018 — Terminal MAC Pro — bash | ..04: ~ — ssh -i k8s_lab ubuntu@13.229.126.99 | ..63: ~ — ssh -i k8s_lab ubuntu@13.229.98.3 |
[ubuntu@ip-10-0-1-104: ~]$ docker node ls
ID HOSTNAME STATUS AVAILABILITY MANAGER STATUS ENGINE VERSION
4g6pqogaae5co8zxe8w6pxfd6 ip-10-0-1-62 Ready Active Leader 18.06.1-ce
3tnvltyohbekgygsvb1ijovy * ip-10-0-1-104 Ready Active Leader 18.06.1-ce
ua0ajfjd4d1mksqzwr80oy2ru ip-10-0-1-163 Ready Active Reachable 18.06.1-ce
[ubuntu@ip-10-0-1-104: ~]$ docker node update --role manager ip-10-0-1-163
[ubuntu@ip-10-0-1-104: ~]$ docker node ls
ID HOSTNAME STATUS AVAILABILITY MANAGER STATUS ENGINE VERSION
4g6pqogaae5co8zxe8w6pxfd6 ip-10-0-1-62 Ready Active Reachable 18.06.1-ce
3tnvltyohbekgygsvb1ijovy * ip-10-0-1-104 Ready Active Leader 18.06.1-ce
ua0ajfjd4d1mksqzwr80oy2ru ip-10-0-1-163 Ready Active Reachable 18.06.1-ce
[ubuntu@ip-10-0-1-104: ~]$ docker node update --role manager ip-10-0-1-62
[ubuntu@ip-10-0-1-104: ~]$ docker node ls
ID HOSTNAME STATUS AVAILABILITY MANAGER STATUS ENGINE VERSION
4g6pqogaae5co8zxe8w6pxfd6 ip-10-0-1-62 Ready Active Reachable 18.06.1-ce
3tnvltyohbekgygsvb1ijovy * ip-10-0-1-104 Ready Active Leader 18.06.1-ce
ua0ajfjd4d1mksqzwr80oy2ru ip-10-0-1-163 Ready Active Reachable 18.06.1-ce
[ubuntu@ip-10-0-1-104: ~]$
```

Docker: The Next-Gen of Virtualization



345

# HA Manager Role

- Add redundancy swarm manager in swarm cluster
  - Restart major manager

```
[ubuntu@ip-10-0-1-104: ~]$ docker node ls
ID HOSTNAME STATUS AVAILABILITY MANAGER STATUS ENGINE VERSION
4g6pqogaae5co8zxe8w6pxfd6 ip-10-0-1-62 Ready Active Reachable 18.06.1-ce
3tnvltyohbekgygsvb1ijovy * ip-10-0-1-104 Ready Active Leader 18.06.1-ce
ua0ajfjd4d1mksqzwr80oy2ru ip-10-0-1-163 Ready Active Reachable 18.06.1-ce
[ubuntu@ip-10-0-1-104: ~]$ sudo shutdown -r now
Connection to 13.229.224.202 closed by remote host.
Connection to 13.229.224.202 closed.
praparn-MacBook-Pro:~$ ssh praparn$
```

- Other will take role for manager within 5 min.

```
[ubuntu@ip-10-0-1-163: ~]$ docker node ls
ID HOSTNAME STATUS AVAILABILITY MANAGER STATUS ENGINE VERSION
4g6pqogaae5co8zxe8w6pxfd6 ip-10-0-1-62 Ready Active Reachable 18.06.1-ce
3tnvltyohbekgygsvb1ijovy * ip-10-0-1-104 Ready Active Leader 18.06.1-ce
ua0ajfjd4d1mksqzwr80oy2ru ip-10-0-1-163 Ready Active Reachable 18.06.1-ce
[ubuntu@ip-10-0-1-163: ~]$ docker node ls
ID HOSTNAME STATUS AVAILABILITY MANAGER STATUS ENGINE VERSION
4g6pqogaae5co8zxe8w6pxfd6 ip-10-0-1-62 Ready Active Reachable 18.06.1-ce
3tnvltyohbekgygsvb1ijovy * ip-10-0-1-104 Unknown Active Unreachable 18.06.1-ce
ua0ajfjd4d1mksqzwr80oy2ru * ip-10-0-1-163 Ready Active Leader 18.06.1-ce
[ubuntu@ip-10-0-1-163: ~]$
```

Docker: The Next-Gen of Virtualization



346

# HA Manager Role

- Remove

- Update node to worker2 and set “Drain” to node

```
ubuntu@ip-10-0-1-104:~$ docker node ls
ID HOSTNAME STATUS AVAILABILITY MANAGER STATUS ENGINE VERSION
4g6pqogaae5co8xe8w6pxfd6 ip-10-0-1-62 Ready Active Reachable 18.06.1-ce
3tnv1tyohbekgygvbsbjjovy * ip-10-0-1-104 Ready Active Reachable 18.06.1-ce
ua0ajfjd4d1mksqzr80oy2ru ip-10-0-1-163 Ready Active Leader 18.06.1-ce
ubuntu@ip-10-0-1-104:~$ docker node update --role worker ip-10-0-1-62
ip-10-0-1-62
ubuntu@ip-10-0-1-104:~$ docker node update --availability drain ip-10-0-1-62
ip-10-0-1-62
ubuntu@ip-10-0-1-104:~$ docker node ls
ID HOSTNAME STATUS AVAILABILITY MANAGER STATUS ENGINE VERSION
4g6pqogaae5co8xe8w6pxfd6 ip-10-0-1-62 Ready Drain Reachable 18.06.1-ce
3tnv1tyohbekgygvbsbjjovy * ip-10-0-1-104 Ready Active Reachable 18.06.1-ce
ua0ajfjd4d1mksqzr80oy2ru ip-10-0-1-163 Ready Active Leader 18.06.1-ce
ubuntu@ip-10-0-1-104:~$
```

- Leave swarm on node “labdocker”

```
...orkshop_112018 — Terminal MAC Pro — bash | -4: ~
ubuntu@ip-10-0-1-62:~$ docker swarm leave
Node left the swarm.
ubuntu@ip-10-0-1-62:~$
ubuntu@ip-10-0-1-104:~$ docker node ls
ID HOSTNAME STATUS AVAILABILITY MANAGER STATUS ENGINE VERSION
4g6pqogaae5co8xe8w6pxfd6 ip-10-0-1-62 Down Drain Reachable 18.06.1-ce
3tnv1tyohbekgygvbsbjjovy * ip-10-0-1-104 Ready Active Reachable 18.06.1-ce
ua0ajfjd4d1mksqzr80oy2ru ip-10-0-1-163 Ready Active Leader 18.06.1-ce
ubuntu@ip-10-0-1-104:~$ docker node rm ip-10-0-1-62
ip-10-0-1-62
ubuntu@ip-10-0-1-104:~$ docker node ls
ID HOSTNAME STATUS AVAILABILITY MANAGER STATUS ENGINE VERSION
3tnv1tyohbekgygvbsbjjovy * ip-10-0-1-104 Ready Active Reachable 18.06.1-ce
ua0ajfjd4d1mksqzr80oy2ru ip-10-0-1-163 Ready Active Leader 18.06.1-ce
ubuntu@ip-10-0-1-104:~$
```

Docker: The Next-Gen of Virtualization



347

# HA Manager Role

- Add redundancy swarm manager in swarm cluster
  - ทำการ join node เข้าไปเป็น manager swarm cluster ด้วยคำสั่ง

`docker swarm join --ca-hash<hash><ip of swarm manager: port>`

```
...orkshop_112018 — Terminal MAC Pro — bash | ... ~ ssh -i k8s_lab ubuntu@13.229.224.202 | ... ~ ssh -i k8s_lab ubuntu@13.229.98.3 | ... ~ ssh -i k8s_lab ubuntu@18.136.196
ubuntu@ip-10-0-1-62:~$ docker swarm leave
Node left the swarm.
ubuntu@ip-10-0-1-62:~$ docker swarm join --token SWMTKN-1-1qnpoqbqw0hilpc5i90szs803tlhyi1h4k84uok10y814bvhqf2-3u9rqxfwhbw52qwqeeiewybl 10.0.1.104:2377
This node joined a swarm as a manager.
ubuntu@ip-10-0-1-62:~$ docker node ls
ID HOSTNAME STATUS AVAILABILITY MANAGER STATUS ENGINE VERSION
mydkcrgwkurxyjiblkf1od17 * ip-10-0-1-62 Ready Active Reachable 18.06.1-ce
3tnv1tyohbekgygvbsbjjovy ip-10-0-1-104 Ready Active Reachable 18.06.1-ce
ua0ajfjd4d1mksqzr80oy2ru ip-10-0-1-163 Ready Active Leader 18.06.1-ce
ubuntu@ip-10-0-1-62:~$
```

- ตรวจสอบค่า token เพื่อ join swarm

```
[ubuntu@ip-10-0-1-62:~$ docker swarm join-token manager
To add a manager to this swarm, run the following command:

 docker swarm join --token SWMTKN-1-1qnpoqbqw0hilpc5i90szs803tlhyi1h4k84uok10y814bvhqf2-3u9rqxfwhbw52qwqeeiewybl 10.0.1.62:2377
ubuntu@ip-10-0-1-62:~$]
```

Docker: The Next-Gen of Virtualization



348

# Workshop: HA Manager Role

- ทำต่อ join new manager เข้าสู่ swarm cluster/update/drain

```
[ubuntu@ip-10-0-1-104:~$ docker node ls
ID HOSTNAME STATUS AVAILABILITY MANAGER STATUS ENGINE VERSION
4g6pqogaae5co8xe8w6pxfd6 ip-10-0-1-62 Ready Active Reachable 18.06.1-ce
3tnvltyohbekgygsb1jjoyv * ip-10-0-1-104 Ready Active Leader 18.06.1-ce
ua0ajfjd4dmksqzwr80oy2ru ip-10-0-1-163 Ready Active Reachable 18.06.1-ce
ubuntu@ip-10-0-1-104:~$

ubuntu@ip-10-0-1-104:~$ docker node ls
ID HOSTNAME STATUS AVAILABILITY MANAGER STATUS ENGINE VERSION
4g6pqogaae5co8xe8w6pxfd6 ip-10-0-1-62 Ready Active Reachable 18.06.1-ce
3tnvltyohbekgygsb1jjoyv * ip-10-0-1-104 Ready Active Reachable 18.06.1-ce
ua0ajfjd4dmksqzwr80oy2ru ip-10-0-1-163 Ready Active Leader 18.06.1-ce
ubuntu@ip-10-0-1-104:~$ docker node update --role worker ip-10-0-1-62
ip-10-0-1-62
ubuntu@ip-10-0-1-104:~$ docker node update --availability drain ip-10-0-1-62
ip-10-0-1-62
ubuntu@ip-10-0-1-104:~$ docker node ls
ID HOSTNAME STATUS AVAILABILITY MANAGER STATUS ENGINE VERSION
4g6pqogaae5co8xe8w6pxfd6 ip-10-0-1-62 Ready Drain Reachable 18.06.1-ce
3tnvltyohbekgygsb1jjoyv * ip-10-0-1-104 Ready Active Reachable 18.06.1-ce
ua0ajfjd4dmksqzwr80oy2ru ip-10-0-1-163 Ready Active Leader 18.06.1-ce
ubuntu@ip-10-0-1-104:~$
```

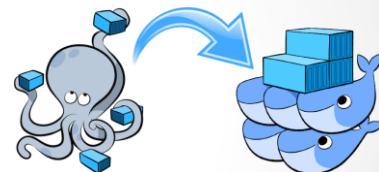
Docker: The Next-Gen of Virtualization



349

# Docker Stack Deploy (Compose)

- Compose syntax: (docker stack deploy)
  - services:**
    - XXX (service name): (Current Version 3.3)
      - image: <image name>
      - deploy: <SWARM>**
        - mode
          - global
          - replicated
        - resource:
          - limits:
            - cpus: 'X.X'
            - memory: XXXM
          - reservations:
            - cpus: 'X.X'
            - memory: 2=XXM
      - restart\_policy:
        - condition: on-failure/any
        - delay: XXs
        - max\_attempts: X
        - windows: XXs
      - update\_config:
        - parallelism: X
        - delay: XXs
        - failure\_action: Xms
        - max\_failure\_ratio: X



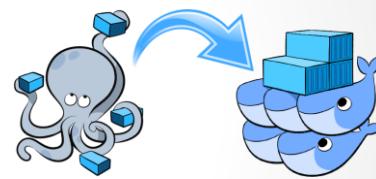
Docker: The Next-Gen of Virtualization



350

# Docker Stack Deploy (Compose)

- Not support for docker stack deploy
  - build
  - cgroup\_parent
  - container\_name
  - devices
  - dns
  - dns\_search
  - tmpfs
  - external\_links
  - links
  - network\_mode
  - security\_opt
  - stop\_signal
  - sysctls
  - userns\_mode



Docker: The Next-Gen of Virtualization



351

# Docker Stack Deploy (Compose)

- Compose syntax: (docker stack deploy)

```
docker stack deploy -c <compose file> <stack name>
```

```
docker stack ls
```

```
docker stack ps <stack name>
```

```
docker stack services <stack name>
```

```
docker stack rm <stack name>
```

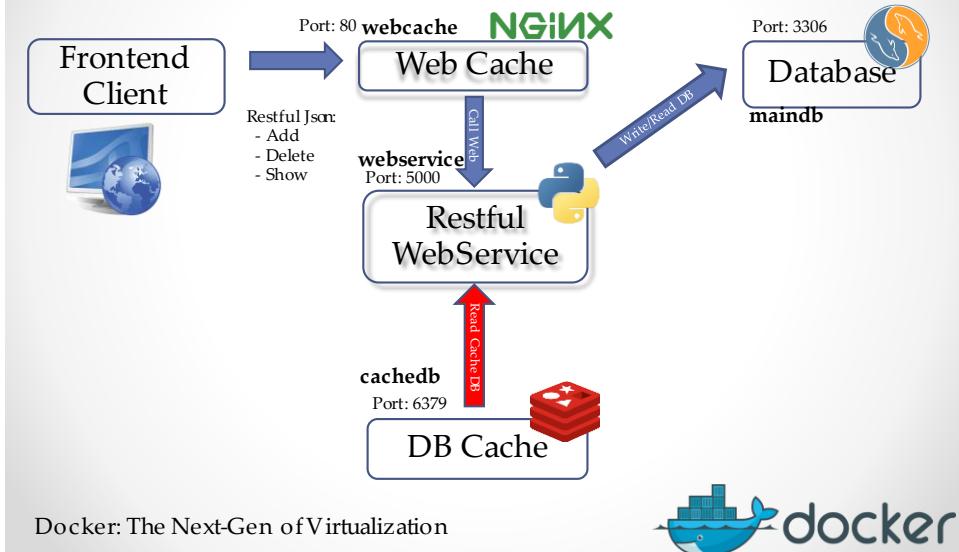
Docker: The Next-Gen of Virtualization



352

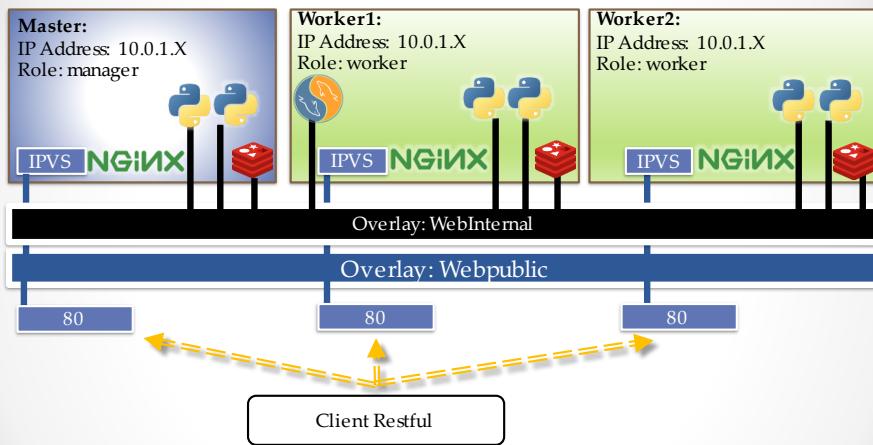
# Workshop: Swarm Compose

- Optimize for WorkLoad



353

# Workshop: Swarm Compose



354

# Workshop: Swarm Compose

```

1 version: '3.3'
2 services:
3 webcache:
4 image: labdocker/cluster:webcache
5 container_name: nginx
6 deploy:
7 mode: global
8 update_config:
9 parallelism: 1
10 delay: 10s
11 restart_policy:
12 condition: on-failure
13 delay: 30s
14 max_attempts: 5
15 window: 60s
16 endpoint_mode: vip
17 depends_on:
18 - webservice
19 - cachedb
20 - maindb
21 networks:
22 webpublic:
23 aliases:
24 - webcache
25 webinternal:
26 aliases:
27 - webcache
28 ports:
29 - "80:80"
30
31 webservice:
32 image: labdocker/cluster:webservice
33 container_name: webservice
34 deploy:
35 mode: replicated
36 replicas: 6
37 update_config:
38 parallelism: 2
39 delay: 10s
40 restart_policy:
41 condition: on-failure
42 delay: 30s
43 max_attempts: 3
44 window: 60s
45 endpoint_mode: vip
46 depends_on:
47 - cachedb
48 - maindb
49 networks:
50 webinternal:
51 aliases:
52 - webservice
53 ports:
54 - "5000:5000"

```

Docker: The Next-Gen of Virtualization



355

# Workshop: Swarm Compose

```

58 maindb:
59 image: labdocker/mariadb:latest
60 container_name: maindb
61 deploy:
62 mode: replicated
63 replicas: 1
64 endpoint_mode: vip
65 environment:
66 - MYSQL_ROOT_PASSWORD=password
67 networks:
68 webinternal:
69 aliases:
70 - maindb
71
72 cachedb:
73 image: labdocker/redis:latest
74 container_name: cachedb
75 deploy:
76 mode: global
77 endpoint_mode: vip
78 networks:
79 webinternal:
80 aliases:
81 - cachedb
82
83 networks:
84 webpublic:
85 driver: overlay
86 ipam:
87 driver: default
88 config:
89 - subnet: 192.168.100.0/24
90 webinternal:
91 driver: overlay
92 ipam:
93 driver: default
94 config:
95 - subnet: 192.168.101.0/24

```

Docker: The Next-Gen of Virtualization



356

# Workshop: Swarm Compose

```
[ubuntu@ip-10-0-1-104:~/docker_workshop_112018/Workshop-2-4-Swarm/python_restfulset$ docker stack deploy -c docker-compose_swarm.yml webservice
Ignoring deprecated options:
container_name: Setting the container name is not supported.

Creating network webservice_webpublic
Creating network webservice_webinternal
Creating service webservice_webcache
Creating service webservice_webservice
Creating service webservice_maindb
Creating service webservice_cachedb
ubuntu@ip-10-0-1-104:~/docker_workshop_112018/Workshop-2-4-Swarm/python_restfulset$]
```

Docker: The Next-Gen of Virtualization



357

# Workshop: Swarm Compose

| ID           | NAME                                         | IMAGE                                        | NODE                                         | DESIRED STATE               | CURRENT STATE          |
|--------------|----------------------------------------------|----------------------------------------------|----------------------------------------------|-----------------------------|------------------------|
| mcj0ux4x25uv | webservice_webcache.wzx0rvl5res39v0lxir2gfh  | labdocke/cluster:webcache                    | swarm-node2                                  | Running                     | Running 46 seconds ago |
| o1223n2c2htp | webservice_webcache.0k95omm28fef22thumppz3uf | labdocke/cluster:webcache                    | swarm-eng                                    | Running                     | Running 49 seconds ago |
| fj8q565u8mt  | webservice_webcache.q814dh13bz1sz4zmjpiy3qa  | labdocke/cluster:webcache                    | swarm-node1                                  | Running                     | Running 50 seconds ago |
| kw80prgn8b3v | webservice_webcache.wzx0rvl5res39v0lxir2gfh  | labdocke/cluster:webcache                    | swarm-node2                                  | Shutdown                    | Failed about a minute  |
| ago          | "task: non-zero exit (1)"                    | webservice_webcache.0k95omm28fef22thumppz3uf | labdocke/cluster:webcache                    | swarm-eng                   | Shutdown               |
| a4foun5bxs5  | "task: non-zero exit (1)"                    | webservice_webcache.0k95omm28fef22thumppz3uf | labdocke/cluster:webcache                    | swarm-node1                 | Failed about a minute  |
| ago          | "task: non-zero exit (1)"                    | webservice_webcache.q814dh13bz1sz4zmjpiy3qa  | labdocke/cluster:webcache                    | swarm-node2                 | Failed about a minute  |
| host2nccrvt  | "task: non-zero exit (1)"                    | webservice_webcache.q814dh13bz1sz4zmjpiy3qa  | labdocke/cluster:webcache                    | swarm-node1                 | Shutdown               |
| ago          | "task: non-zero exit (1)"                    | vhutislnb7np                                 | webservice_webcache.wzx0rvl5res39v0lxir2gfh  | swarm-node2                 | Failed about a minute  |
| ago          | "task: non-zero exit (1)"                    | ktjzx8thiq6k                                 | webservice_webcache.q814dh13bz1sz4zmjpiy3qa  | swarm-node1                 | Shutdown               |
| ago          | "task: non-zero exit (1)"                    | 7siao04nhvmh                                 | webservice_webcache.0k95omm28fef22thumppz3uf | swarm-eng                   | Failed about a minute  |
| ago          | "task: non-zero exit (1)"                    | axhd6tzu971                                  | webservice_cachedb.wzx0rvl5res39v0lxir2gfh   | labdocke/redis:latest       | Running                |
| ago          | "task: non-zero exit (1)"                    | t8jc8appm6wu                                 | webservice_cachedb.q814dh13bz1sz4zmjpiy3qa   | swarm-node1                 | Running                |
| e app        | "task: non-zero exit (1)"                    | 32d2fa3cjcui                                 | webservice_cachedb.0k95omm28fef22thumppz3uf  | labdocke/redis:latest       | Running                |
| e app        | "task: non-zero exit (1)"                    | mtbhxlkuwjg7                                 | webservice_webservice.1                      | swarm-node1                 | Running                |
| o            | "task: non-zero exit (1)"                    | nw2jky8drn42                                 | webservice_webservice.1                      | labdocke/cluster:webservice | swarm-eng              |
| ago          | "task: non-zero exit (1)"                    | ivnofusuyxx8                                 | webservice_maindb.1                          | labdocke/mysql:latest       | swarm-node2            |

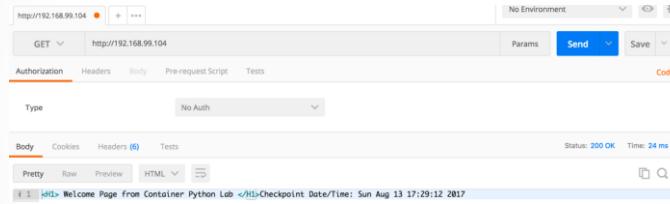
Docker: The Next-Gen of Virtualization



358

# Workshop: Swarm Compose

```
praparns-MacBook-Pro:~ praparns$ docker-machine ls
NAME ACTIVE DRIVER STATE IP ADDRESS PORT(S)
libdocker - virtualbox Running tcp://192.168.99.103:2375
v1.15/version: x509: certificate is valid for 192.168.99.100, not 192.168.99.103
swarm-eng virtualbox Running tcp://192.168.99.104:2376
swarm-node1 - virtualbox Running tcp://192.168.99.105:2376
swarm-node2 - virtualbox Running tcp://192.168.99.106:2376
v17.66.0-ce
praparns-MacBook-Pro:~ praparns$ export Server_IP=192.168.99.104
praparns-MacBook-Pro:~ praparns$ curl http://$Server_IP:$Server_Port/
<H1> Welcome Page from Container Python Lab </H1>Checkpoint Date/Time: Sun Aug 13 17:27:58 2017
```



Docker: The Next-Gen of Virtualization



359

# Workshop: Swarm Compose

```
POST http://192.168.99.104/users/insertuser
Content-Type: application/json
{
 "id": "1",
 "user": "Praporn Luangphoolap",
 "describe": "Slave"
}
```

```
GET http://192.168.99.104/users/1
Content-Type: application/json
{
 "id": "1",
 "user": "Praporn Luangphoolap",
 "describe": "Slave"
}
```

Docker: The Next-Gen of Virtualization



360

# Workshop: Swarm Compose

The screenshot shows a Postman interface with the following details:

- URL: `http://192.168.99.104/users/removeuser/1`
- Type: No Auth
- Status: 200 OK Time: 33 ms
- Body: `##### Record was deleted (Both Database Cache) #####`

Docker: The Next-Gen of Virtualization



361

# Portainer for Docker

• • •

Docker: The Next-Gen of Virtualization



362

# portainer for Docker

- portainer.io เป็น open source management ที่สามารถใช้บริหารจัดการ docker-machine ผ่านหน้า GUI web ได้อย่างมีประสิทธิภาพแบบ On premise
- การใช้งาน portainer จะใช้การบริหารจัดการผ่านหน้าเบราว์เซอร์ (x.x.x.x:9000)

Docker: The Next-Gen of Virtualization

363

# portainer for Docker

Docker: The Next-Gen of Virtualization

364

# portainer for Docker

The screenshot shows the Portainer dashboard for a Docker host named 'labdocker'. Key statistics include:

- Containers:** 1 running, 0 stopped
- Images:** 31
- Volumes:** 5
- Networks:** 4

Below the dashboard is a table of running containers:

Name	State	Quick actions	Stack	Image	IP Address	Published Ports	Ownership
vigilant_pare	running	<span>Containers</span>	-	portainer/portainer	172.17.0.2	9000:9000	public

At the bottom, there's a banner for Docker: "Docker: The Next-Gen of Virtualization" next to the Docker logo.

365

# portainer for Docker

The screenshot shows the 'Create container' form in Portainer. The container is named 'nginx' and is based on the image 'labdocker/nginx:latest'. It has port mapping from host port 80 to container port 80. The access control is set to 'Administrators'.

**Create container**

**Name:** nginx

**Image configuration:**

**Name:** labdocker/nginx:latest      **Registry:** DockerHub

**Always pull the image:**

**Ports configuration:**

**Port mapping:**  map additional ports

host	80	→	container	80	TCP	UDP	<input checked="" type="checkbox"/>
------	----	---	-----------	----	-----	-----	-------------------------------------

**Access control:**

**Enable access control:**

Administrators       Restricted

I want to restrict the management of this resource to administrators only      I want to restrict the management of this resource to a set of users and/or teams

**Actions:**

**Deploy the container**

Advanced container settings

At the bottom, there's a banner for Docker: "Docker: The Next-Gen of Virtualization" next to the Docker logo.

366

# portainer for Docker

192.168.99.100

NMac Ked - Mac OS... Medium jerkins vagrant Mesos Vue docker NGINX Taiwan Kubernetes PWA\_Progressive\_W... MYSQL\_Cluster

## Welcome to nginx!

If you see this page, the nginx web server is successfully installed and working. Further configuration is required.

For online documentation and support please refer to [nginx.org](http://nginx.org). Commercial support is available at [nginx.com](http://nginx.com).

Thank you for using nginx.

**Container list**

Name	State	Quick actions	Stack	Image	IP Address	Published Ports	Ownership
nginx	running	[Start] [Stop] [Kill] [Restart] [Pause] [Resume] [Remove]	-	labdockers/nginx:latest	172.17.0.3	80:80	administrators
vigilant_pare	running	[Start] [Stop] [Kill] [Restart] [Pause] [Resume] [Remove]	-	portainer/portainer	172.17.0.2	9000:9000	public

Docker: The Next-Gen of Virtualization 

367

# portainer for Docker

portainer.io

ACTIVE ENDPOINT: local

ENDPOINT ACTIONS

- Dashboard
- App Templates
- Containers
- Images
- Networks
- Volumes
- Events
- Engine

PORTAINER SETTINGS

- User management
- Endpoints
- Registries
- Settings

Container statistics

About statistics

This view displays real-time statistics about the container nginx as well as a list of the running processes inside this container.

Refresh rate: 5s

Memory usage: 10 MB

CPU usage: 1.0%

Network usage: RX on eth0, TX on eth0

Processes

UID	PID	PPID	C	STIME	TTY	TIME	CMD
root	1	0	-	22:28:31	-	-	/usr/sbin/nginx -g "daemon off;"

Container console

Console

Exec into container as default user using command sh

```
/usr/bin # ls
addressctl astring b chroot deluser fakeidfd fipd httpd killall5 loadfont ncacnnp nsh-client poweroff rdate readprofile sendmail setfont vifdhd
addresser obpsaved delgroup cron dead fdesc fipfd httpd killall5 loadfont ncacnsp nsh-client poweroff rdate readahead remove-shell setfont vifdhd
/user/bin #
```

Docker: The Next-Gen of Virtualization 

368

## Recapture for Day 2

- Dockerfile and Build
- Compose
- Registry
- Swarm Mode
  - Conceptual of Swarm
  - Swarm Mode Architecture
  - Swarm init/join cluster system
  - Swarm service
  - Orchestrator Assignment
  - Config and Secret
  - Network Overlay and Ingress
  - HA Manager Role
  - Docker Stack Deploy (Compose Swarm Mode)
- portainer for Docker
- Q&A

Docker: The Next-Gen of Virtualization



369

By Praparn Luengphoonlap  
Email: eva10409@gmail.com

Docker: The Next-Gen of Virtualization



370