Docker:Zero to Hero (Day 1)

By Praparn Luengphoonlap
Email: eva10409@gmail.com

Outline Day 1

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

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

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

Prerequisite

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

Prerequisite

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

Lab Resource

- Repository for lab

Lab Resource

- Software in lab

Lab Resource

- Download on Google Drive
  - https://tinyurl.com/feu5jhyr
- Download on GitHub
  - git clone https://github.com/praparn/docker-workshop-032023

Lab Resource

- https://tinyurl.com/y9svjua2

DockerCon2022

DockerCon 2022

DockerCon: What Makes a Successful CFP Submission

The DockerCon 2022 Call for Papers is now open! DockerCon is one of the largest developer events in the world, with over 80,000 developers registering for each of the last two events. At the core of DockerCon is the chance for members of the community to share their tips, tr...

dockercon, DockerCon 2022, Dockercon CFP

PETER MCKEE
Feb 03 2022

Keep Reading →

DockerCon: What Makes a Successful CFP Submission

The DockerCon 2022 Call for Papers is now open! DockerCon is one of the largest developer events in the world, with over 80,000 developers registering for each of the last two events. At the core of DockerCon is the chance for members of the community to share their tips, tr...

dockercon, DockerCon 2022, Dockercon CFP

PETER MCKEE
Feb 03 2022

Keep Reading →

dockercon

First: it helps to know your audience. At DockerCon, we draw attendees from all over the world with a range of experiences from newbie to guru. In general, however, the majority of the developers who attend DockerCon fall into these three categories:

- **New developers**, who are just getting started with containerization, using Docker, and building cloud native applications. Sessions introducing Docker, ones that put Docker into context of popular programming languages such as JavaScript, Python, Go, .NET and Java, and that help a developer get up and running are quite popular. In these cases, sharing the details you wish you knew when you were getting started make for outstanding sessions and are some of our most viewed content.
- **Experienced developers**, who use Docker but are looking for ways to get their applications built faster and grow their capabilities (both in terms of features as well as individual skills). Proposals for sessions that share best practices, case studies from real-world development experience, as well as ways to incorporate automation, scalability and security within a development inner loop. These all meet the needs of developers with more experience.
- **Expert developers,** who are digging into the command line, building Docker Images from scratch, and contributing to OSS projects. These developers are attracted to the “black belt” sessions that dive into the internals of the Docker Engine, extensibility and integration into other tools such as CI/CD, repositories, and CSPs. Examples that show how hard problems are solved, how the Docker Engine works, digging into OSS projects, and showing how you can “run with scissors” make for compelling content on cutting-edge topics.

docker

Workshop: Access Docker on AWS

Workshop: Access Docker on AWS

https://labs.play-with-docker.com

NMac Ked - Mac ... | Medium | Infra | NOVA | jerkins | vagrant | Mesos | Vue | docker | NGINX | Taiwan | Kubernetes | PWA_Pr

Contribute

Play with Docker

A simple, interac

03:59:39

CLOSE SESSION

Instances

+ ADD NEW INSTANCE

192.168.0.13
node1

c8ecrt5m_c8ecrudmrepg00av6lig

IP
192.168.0.13

OPEN PORT

Memory
0.80% (32.12MiB / 3.906GiB)

CPU
0.60%

SSH
ssh ip172-18-0-54-c8ecrt5mrepg00av6li0@direct.labs.pla

DELETE

EDITOR

```
# completely the user's responsibilites.                    #
#                                                           #
# The PWD team.                                             #
#############################################################
[node1] (local) root@192.168.0.13 ~
$ $ docker ps
CONTAINER ID   IMAGE     COMMAND   CREATED   STATUS    PORTS     NAMES
[node1] (local) root@192.168.0.13 ~
$ docker version
Client: Docker Engine - Community
 Version:           20.10.0
 API version:       1.41
 Go version:        go1.13.15
 Git commit:        7287ab3
 Built:             Tue Dec  8 18:54:00 2020
 OS/Arch:           linux/amd64
 Context:           default
 Experimental:      true
```

Landscape of the world now

Landscape of the world now

Key Trends

Security		Container Usage	
87%	of container images have high or critical vulnerabilities	Companies running more than 1,000 nodes could save over **$10 million annually**	
90%	of granted permissions are not used	**72%**	of containers live less than five minutes (up from 44%)
15%	of high and critical vulnerabilities are in use at runtime	**69%**	of CPU resources are unused (up from 34%)

Ref: https://sysdig.com/blog/2023-cloud-native-security-usage-report/

docker

Landscape of the world now

Landscape of the world now

Landscape of the world now

Landscape of the world now

docker

Docker Principle

...

What is docker ?

Existing Technology

Existing Technology

- Development Environment (Mix everything possible)

- Apache 2.20 Web Server
- PHP 5.5 Engine
- Larevel 4.1 Framework
- MaridDB 5.1
- Etc

(All-in-One Server)

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

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

Existing Technology

- <u>Production Environment (Best design)</u>
- Day 1: Application 1: Implement

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

- IIS 8
- .Net FrameWork 3.5

- MariaDB 5.1

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

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)

What is docker ?

Existing Technology

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

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

Existing Technology

- Development Environment (Clone from Production)

Virtual Machine reach limit:
- Resource insufficient (Almost from disk)
- Conflict version
- Huge of disk duplication
- Conflict & dependency still exist

Existing Technology

- <u>Production Environment</u>
- Day 1: Application 1: Implement

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

- IIS 8
- .Net FrameWork 3.5

- MariaDB 5.1

- Day 2: Application 2: Need to implement
  - Need PHP 7.0 ?
  - MariaDB 10.1.14 (Need search feature on 10.1)

- So… The problem still exist.

What is docker ?

Docker: The Next-Gen of Virtualization

What is docker ?

What is docker ?

Core Principles of the Docker Platform

INDEPENDENCE

OPENNESS

SIMPLICITY

dockercon17 EU

docker

What is docker ?

- Docker คือ open platform solution ที่ทำงานภายใต้คอนเซ็ปต์ของ container virtualize technology (operating -system –level virtualization)
- ผู้ใช้สามารถสร้างสภาพแวดล้อมเพื่อใช้ในพัฒนาโปรแกรมและส่งมอบเพื่อใช้งานในสภาพเดียวกัน
- Build, Ship, Run
- เหมาะสำหรับ Developer, DevOps, Architecture, Engineer
- เขียนด้วยภาษา Go
- รองรับการติดตั้งบนบนลินุกซ์ 64 บิต (kernel 3.1.0 and upper) (Official)
  - Ubuntu/ Debian (X64/ARM/ARM64)
  - CentOS (Obsolete)
  - Fedora
  - Red Hat Enterprise Linux
  - Microsoft Windows Server
  - Rasbian

Docker's product

Docker Hub

The world's leading service for finding and sharing container images with your team and the Docker community.

For developers and those experimenting with Docker, Docker Hub is your starting point into Docker containers. Create an account and start exploring the millions of images that are available from the community and verified publishers.

See Docker Hub

Docker Desktop

The preferred choice for millions of developers that are building containerized apps.

Docker Desktop is an application for MacOS and Windows machines for the building and sharing of containerized applications. Access Docker Desktop and follow the guided onboarding to build your first containerized application in minutes.

See Docker Desktop

Docker's feature

Container Runtime

Docker Engine powers millions of applications worldwide, providing a standardized packaging format for diverse applications.

See Container Runtime

Developer Tools

The fastest way to securely build, test, and share cloud-ready modern applications from your desktop.

See Developer Tools

Docker's feature

Kubernetes

Kubernetes has become the standard orchestration platform for containers. All the major cloud providers support it, making it the logical choice for organizations looking to move more applications to the cloud.

See Kubernetes

What is docker ?

What is docker ?

Docker: The Next-Gen of Virtualization

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/

What is docker ?

- DOCKER CE

Desktop

Platform	x86_64 / amd64
Docker Desktop for Mac (macOS)	✓
Docker Desktop for Windows	✓

Server

Docker provides `.deb` and `.rpm` packages from the following Linux distributions and architectures:

Platform	x86_64 / amd64	ARM	ARM64 / AARCH64
CentOS	✓		✓
Debian	✓	✓	✓
Fedora	✓		✓
Raspbian		✓	✓
Ubuntu	✓	✓	✓

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

Who use docker now ?

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

Who use docker now ?

PayPal

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

Container Life Cycle

- Developer
  - สร้าง template สำหรับโปรแกรม (build)
  - เริ่มรัน template สำหรับพัฒนาโปรแกรม (run)
  - พัฒนาเสร็จเรียบร้อย สั่ง save version เพื่อเตรียมนำขึ้นใช้งาน (commit)

- Operation
  - เริ่มรัน template ที่ได้รับจาก developer (run) บนเครื่อง production
  - หยุดการให้บริการโปรแกรม (stop, kill)
  - ลบโปรแกรมออกจากเครื่อง production (rm, rmi)

Container Life Cycle

What Cool of Docker ?

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

Separate of Concern

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

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

Benefit of docker for DevOps

- Build-Ship-Run

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

Docker Info

```
ubuntu@ip-10-200-20-58:~$ docker version
Client: Docker Engine - Community
 Version:           23.0.1
 API version:       1.42
 Go version:        go1.19.5
 Git commit:        a5ee5b1
 Built:             Thu Feb  9 19:46:56 2023
 OS/Arch:           linux/amd64
 Context:           default

Server: Docker Engine - Community
 Engine:
  Version:          23.0.1
  API version:      1.42 (minimum version 1.12)
  Go version:       go1.19.5
  Git commit:       bc3805a
  Built:            Thu Feb  9 19:46:56 2023
  OS/Arch:          linux/amd64
  Experimental:     false
 containerd:
  Version:          1.6.18
  GitCommit:        2456e983eb9e37e47538f59ea18f2043c9a73640
 runc:
  Version:          1.1.4
  GitCommit:        v1.1.4-0-g5fd4c4d
 docker-init:
  Version:          0.19.0
  GitCommit:        de40ad0
ubuntu@ip-10-200-20-58:~$
```

```
ubuntu@ip-10-200-20-58:~$ docker info
Client:
 Context:    default
 Debug Mode: false
 Plugins:
  buildx: Docker Buildx (Docker Inc.)
    Version:  v0.10.2
    Path:     /usr/libexec/docker/cli-plugins/docker-buildx
  compose: Docker Compose (Docker Inc.)
    Version:  v2.16.0
    Path:     /usr/libexec/docker/cli-plugins/docker-compose
  scan: Docker Scan (Docker Inc.)
    Version:  v0.23.0
    Path:     /usr/libexec/docker/cli-plugins/docker-scan

Server:
 Containers: 0
  Running: 0
  Paused: 0
  Stopped: 0
 Images: 0
 Server Version: 23.0.1
 Storage Driver: overlay2
  Backing Filesystem: extfs
  Supports d_type: true
  Using metacopy: false
  Native Overlay Diff: true
  userxattr: false
 Logging Driver: json-file
 Cgroup Driver: systemd
 Cgroup Version: 1
 Plugins:
  Volume: local
  Network: bridge host ipvlan macvlan null overlay
  Log: awslogs fluentd gcplogs gelf journald json-file local logentries splunk syslog
 Swarm: inactive
 Runtimes: io.containerd.runc.v2 runc
 Default Runtime: runc
 Init Binary: docker-init
 containerd version: 2456e983eb9e37e47538f59ea18f2043c9a73640
 runc version: v1.1.4-0-g5fd4c4d
 init version: de40ad0
 Security Options:
  apparmor
  seccomp
   Profile: builtin
 Kernel Version: 5.15.0-1028-aws
 Operating System: Ubuntu 20.04.5 LTS
 OSType: linux
 Architecture: x86_64
 CPUs: 2
 Total Memory: 1.898GiB
 Name: ip-10-200-20-58
 ID: d23af148-f020-4539-b5d4-f17ea62551f4
 Docker Root Dir: /var/lib/docker
 Debug Mode: false
 Registry: https://index.docker.io/v1/
 Experimental: false
 Insecure Registries:
  127.0.0.0/8
 Live Restore Enabled: false

ubuntu@ip-10-200-20-58:~$
```

Log4j CVE-2021-44228

docker blog Products Developers Pricing About Us P

Apache Log4j 2 CVE-2021-44228

JUSTIN CORMACK
Dec 11 2021

Update: 13 December 2021

As an update to CVE-2021-44228, the fix made in version 2.15.0 was incomplete in certain non-default configurations. An additional issue was identified and is tracked with **CVE-2021-45046**. For a more complete fix to this vulnerability, it's recommended to update to Log4j2 **2.16.0**.

Original post below has now been updated:

Ref:https://www.docker.com/blog/apache-log4j-2-cve-2021-44228/

What's New

Docker: The Next-Gen of Virtualization

What's New

Docker is Updating and Extending Our Product Subscriptions

SCOTT JOHNSTON
Aug 31 2021

Docker is used by millions of developers to build, share, and run any app, anywhere, and 55% of professional developers use Docker every day at work. In these work environments, the increase in outside attacks on software supply chains is accelerating developer demand for Docker's trusted content, including Docker Official Images and Docker Verified Publisher images. Finally, the rapid global growth in developers – to an estimated 45 million by 2030 – pushes us to scale sustainably so we may continue to provide an innovative, free Docker experience that developers love.

To meet these challenges, today we're announcing updates and extensions to our product subscriptions: **Personal, Pro, Team, and Business.** These updated product subscriptions provide the productivity and collaboration developers rely on with the scale, security, and trusted content businesses require, and do so in a manner sustainable for Docker.

What you need to know:

- We're introducing a **new** product subscription, Docker Business, for organizations using Docker at scale for application development and require features like secure software supply chain management, single sign-on (SSO), container registry access controls, and more.
- Our Docker Subscription Service Agreement includes a change to the terms for **Docker Desktop**:
  - Docker Desktop **remains free** for small businesses (fewer than 250 employees AND less than $10 million in annual revenue), personal use, education, and non-commercial open source projects.
  - It requires a paid subscription (**Pro**, **Team** or **Business**), starting at $5 per user per month, for professional use in larger businesses. You may directly purchase here, or share this post and our solution brief with your manager.
  - While the effective date of these terms is August 31, 2021, there is a grace period until January 31, 2022 for those that require a paid subscription to use Docker Desktop.
- Docker Pro, Docker Team, and Docker Business subscriptions **include** commercial use of Docker Desktop.
- The **existing** Docker Free subscription has been renamed Docker Personal.
- **No changes** to Docker Engine or any upstream open source Docker or Moby project.
- *Check out our FAQ for more information.*

Personal	Pro	Team	Business
1	1	5+	50+
Ideal for individual developers, education, open source communities, and small businesses.	Ideal for individual developers	Ideal for development teams	Ideal for medium and large businesses
$0	$5 /month	$7 /user/month minimum 5 seats	$21 /user/month
Includes Docker Desktop	Includes Docker Desktop	Includes Docker Desktop	Includes Docker Desktop

docker

What's New

docker docs | Search the docs | Home | Guides | Manuals | Reference | Samples | Contribute

/ Manuals / Docker Engine / Release notes / Engine 23.0

- Docker Desktop
- Docker Engine
  - Overview
  - Install
  - Storage
  - Networking
  - Working with Docker Engine
  - Logging
  - Security
  - Advanced concepts
  - Deprecated features
  - Release notes
    - **Engine 23.0**
    - Previous versions
- Docker Build

Docker Engine 23.0 release notes

Note

From Docker Engine version 23.0.0, Buildx is distributed in a separate package: `docker-buildx-plugin`. In earlier versions, Buildx was included in the `docker-ce-cli` package. When you upgrade to this version of Docker Engine, make sure you update all packages. For example, on Ubuntu:

```
$ sudo apt-get install docker-ce docker-ce-cli containerd.io docker-buildx-plugin docker-compose-plugin
```

Refer to the Docker Engine installation instructions for your operating system for more details on upgrading Docker Engine.

This page describes the latest changes, additions, known issues, and fixes for Docker Engine version 23.0.

For more information about:

- Deprecated and removed features, see Deprecated Engine Features.
- Changes to the Engine API, see Engine API version history.

Starting with the 23.0.0 release, Docker Engine moves away from using CalVer versioning, and starts using the SemVer versioning format. Changing the version format is a stepping-stone towards Go module compatibility, but the repository doesn't yet use Go modules, and still requires using a "+incompatible" version. Work continues towards Go module compatibility in a future release.

Ref: https://docs.docker.com/engine/release-notes/23.0/

What's New

23.0.0

2023-02-01

For a full list of pull requests and changes in this release, refer to the relevant GitHub milestones:

- docker/cli, 23.0.0 milestone
- moby/moby, 23.0.0 milestone

New

- Set Buildx and BuildKit as the default builder on Linux. moby/moby#43992
  - Alias `docker build` to `docker buildx build`. docker/cli#3314
  - The legacy builder can still be used by explicitly setting `DOCKER_BUILDKIT=0`.
  - There are differences in how BuildKit and the legacy builder handle multi-stage builds. For more information, see Multi-stage builds.
- Add support for pulling `zstd` compressed layers. moby/moby#41759, moby/moby#42862
- Add support for alternate OCI runtimes on Linux, compatible with the containerd runtime v2 API. moby/moby#43887, moby/moby#43993
- Add support for the containerd `runhcs` shim on Windows (off by default). moby/moby#42089
- Add `dockerd --validate` to check the daemon JSON config and exit. moby/moby#42393
- Add the ability to configure the daemon's HTTP proxy via flags or JSON config. moby/moby#42835
- Add support for RFC 3021 point-to-point networks (IPv4 /31s) and single hosts (IPv4 /32s). For networks with two or fewer addresses, IPAM won't reserve a network and broadcast address. moby/moby#42626
- Add support for setting `ipvlan_flag` and using the `l3s` `ipvlan_mode` in the `ipvlan` network driver. moby/moby#42542
- Add support for displaying the value of the `metacopy` option for the `overlay2` storage driver. moby/moby#43557
- Add support for describing Windows devices using the syntax `IDType://ID`. moby/moby#43368
- Add `RootlessKit`, `slirp4netns`, and `VPNKit` version reporting. moby/moby#42330
- Add experimental support for SwarmKit cluster volumes (CSI). moby/moby#41982
  - CLI: Add cluster volume (CSI) options to `docker volume`. docker/cli#3606
  - CLI: Add cluster volume (CSI) support to `docker stack`. docker/cli#3662
- Add support for SwarmKit jobs in `docker stack deploy`. docker/cli#2907
- Add the `docker stack config` command to output the merged and interpolated config files as utilized by `stack deploy`. docker/cli#3544
- Add a new `docker context show` command that prints the name of the current context. docker/cli#3567
- Add the `--format=json` shorthand variant of `--format="{{ json . }}"` to all commands supporting the `--format` flag. docker/cli#2936
- Add a `--quiet` option to `docker create` and `docker run` commands to suppress output when pulling an image. docker/cli#3377
- Add a `--force` option to `docker network rm` subcommand. Causes CLI to return a 0 exit code even if the network doesn't exist. Has no effect on the server-side procedure for removing a network. docker/cli#3547

What's New

- Add a `--signal` option to `docker stop` and `docker restart`. docker/cli#3614
- Add a `-v/--version` flag to `docker-proxy`. moby/moby#44703
- Plugins are now discovered in well-known user-level paths when the daemon is running in rootless mode. moby/moby#44778
- The daemon now handles common alternate JSON encodings in the JSON configuration file gracefully, and reports useful errors. moby/moby#44777, moby/moby#44832
  - UTF-8 with a byte order mark is accepted.
  - UTF-16 with a byte order mark is accepted.
  - Invalid UTF-8 is reported early and with a comprehensible error message.
- Allow use of `STOPSIGNAL` via `docker commit`. moby/moby#43369
- Add a new option to the `awslogs` log driver to allow skipping log stream creation in CloudWatch. moby/moby#42132
- Add a new option to the `awslogs` log driver to specify the log format that's sent to CloudWatch. moby/moby#42838
- Add a new option to the `fluentd` log driver to set the reconnection interval. moby/moby#43100
- Add new options-setters to the Go API client: `WithTLSClientConfigFromEnv()`, `WithHostFromEnv()`, and `WithVersionFromEnv()`. moby/moby#42224
- Add generation of shell command completion through a `docker completion` subcommand. docker/cli#3429
- API: Add a `Swarm` header to `GET /_ping` and `HEAD /_ping`, allowing single-request detection of Swarm support. moby/moby#42064
- API: Add a `signal` parameter to `POST /containers/{id}/stop` and `POST /containers/{id}/restart` to set the signal used. moby/moby#43206
- API: Add a `CreateMountPoint` parameter to `POST /containers/create`. moby/moby#43484
- API: Add a `shared-size` parameter to `GET /images/json` to enable shared-size computation of images. moby/moby#42531
- API: Add a `type` parameter to `GET /system/df`, to control what object types to are considered when computing disk usage. moby/moby#42559
- systemd: Use a systemd-managed containerd instead of a daemon-managed containerd. moby/moby#42373
- systemd: Start `docker.service` after `time-set.target`. moby/moby#43107

Removed

- Remove support for reading configuration from `~/.dockercfg`. docker/cli#2504
  - This location has been deprecated since 1.7.0.
  - Deprecation notice
- Remove the `-g` and `--graph` daemon options in favor of `--data-root`. docker/cli#3739
  - These options have been hidden and deprecated since 17.05.
  - Deprecation notice
- Remove client-side sorting of results, in favor of the order in which the search API returns. docker/cli#3470
- Remove warnings related to deprecated storage drivers from the CLI. Warnings are now handled by the daemon instead. docker/cli#3542
- Remove `Experimental` client field from `docker version`. docker/cli#3543

What's New

- Require explicit opt-in to use deprecated storage drivers, and don't automatically select them when upgrading. moby/moby#43378
- Remove deprecated support for `overlay` and `overlay2` storage drivers on backing filesystems without `d_type` support. moby/moby#43472
  - Deprecation notice
- Remove the deprecated `overrideKernelCheck` option from the `overlay2` storage driver. moby/moby#44279 Deprecation notice
- Remove support for the deprecated `io.containerd.runtime.v1.linux` OCI runtime. moby/moby#43695
- Remove LCOW (Linux Containers on Windows). moby/moby#42451, moby/moby#42499, moby/moby#42506, moby/moby#42511, moby/moby#42520, moby/moby#42683, moby/moby#42684, moby/moby#42685, moby/moby#43187
  - LCOW was introduced as a technical preview in 17.09 and deprecated in 20.10.
  - Deprecation notice
- Remove daemon options related to legacy overlay networks used with standalone Swarm.
  - Remove `--cluster-xx` options from `dockerd`. moby/moby#40383
  - Remove `host-discovery` and overlay networks with external k/v stores. moby/moby#42247
  - Deprecation notice
- Remove a deprecated `arm` platform fallback. `--platform linux/arm/vY` will now return a error when `arm/vY` isn't available instead of pulling the wrong image. moby/moby#44414
- Remove the deprecated `SetCustomHTTPHeaders()`, `CustomHTTPHeaders()` options-setters from the Go client API. moby/moby#42694
- Remove the deprecated `WithDialer()` option-setter from the Go client API. moby/moby#44022
  - Use `WithDialContext()` instead.
- Remove the daemon implementation of `opts.QuotedString`. The implementation has moved to the CLI. moby/moby#43250
- Remove separate daemon ID from trust-key in the daemon, and disable generating the trust-key. moby/moby#43555
- API: Remove the deprecated `KernelMemory` option from `POST /containers/create` on API version >= 1.42. moby/moby#43214
  - Deprecation notice

Deprecated

- Require Windows Server RS5 / LTSC 2019 (build 17763) as the minimum to run the daemon. moby/moby#43254
- Deprecate `BuilderSize` on API version >= 1.42. moby/moby#42608
- Deprecate `BuildCache.Parent` in favor of the newly introduced `BuildCache.Parents` on API version >= 1.42. moby/moby#43908
- Deprecate `pkg/urlutil`, moving the implementation to `builder/remotecontext/urlutil`. moby/moby#43477

Upgrades

- Upgrade Go to `1.19.5`. docker/cli#3958, moby/moby#44794
- Upgrade `rootlesskit` to `v0.14.4`. moby/moby#42708
- Upgrade `buildkit` to `v0.10.6`. moby/moby#43239
- Upgrade `buildx` to `v0.10.2`. docker/docker-ce-packaging#840

What's New

- Upgrade `swarmkit` to `v2.0.0-20230119195359-904c221ac281`. moby/moby#44858
- Upgrade `containerd` to `v1.6.16`. moby/moby#44766, moby/moby#44769, moby/moby#44881
- Upgrade `runc` to `v1.1.4`. moby/moby#44039
- Upgrade `hcsshim` `v0.9.6`. moby/moby#44658
- The `btrfs` storage driver now depends on Linux kernel headers (>= 4.12) instead of headers from btrfs-progs. moby/moby#44776

Security

- Change permissions on container `hostconfig.json` files to `0600` (was `0644`). moby/moby#41620
- Fix `--seccomp-profile` not accepting `unconfined` and renamed the default seccomp profile to `builtin`. moby/moby#42481
- Always build with seccomp support, and remove the `seccomp` build tag. moby/moby#42501
- Add seccomp support on `riscv64`. moby/moby#43553
- Add support for setting flags passed to `seccomp(2)` in seccomp profiles. moby/moby#42648
- Refactor seccomp types to reuse runtime-spec, and add support for `ErrnoRet`. moby/moby#42005
- Add support for `DefaultErrnoRet` in `seccomp` profiles. moby/moby#42604
- Add an explicit `DefaultErrnoRet` field to the default seccomp profile, with no behavior change. moby/moby#42649
- Block `socket` with `AF_VSOCK` in the default seccomp profile. moby/moby#44563
- Re-enable `process_vm_readv` and `process_vm_writev` in the default seccomp profile. moby/moby#42083
- Add syscalls related to PKU to the default seccomp profile. moby/moby#43812
- Allow `clock_settime64` with `CAP_SYS_TIME`. moby/moby#43775
- Allow `bpf` with `CAP_BPF` and `perf_event_open` with `CAP_PERFMON`. moby/moby#43988
- Explicitly set the `clone3` syscall to return `ENOSYS` in the default seccomp profile, in order to ensure `glibc` will correctly fallback to using `clone`. moby/moby#42681

Bug fixes and enhancements

- Promote `overlay2` to be the default storage driver (`btrfs` and `zfs` are now opt-in). moby/moby#42661
- Add a loading spinner to the `docker cp` command. docker/cli#2708
- Deprecate the `ElectAuthServer` function, and made it return the default registry without calling the `GET /info` API endpoint. docker/cli#2819
- Progress bars are no longer reversed when rolling back Swarm services. docker/cli#2940
- Use `net.JoinHostPort()` to fix formatting with IPv6 addresses. docker/cli#2972
- CLI error messages are now printed to `stderr`. docker/cli#3044
- Improve performance of `docker info` if a custom `--format` is used that only uses local information. With this change, the CLI only uses the daemon API if it detects that information from the daemon is needed. docker/cli#3179
- Remove the default value from the `--stop-signal` flag, as it may not reflect the actual default used by the daemon. docker/cli#3245
- Add Compose schema `3.10` to `docker stack`; allow omitting the `version` field (resulting in `latest`). docker/cli#3257

What's New

Status	Feature	Deprecated	Remove
Deprecated	Legacy builder for Linux images	v23.0.0	-
Deprecated	Legacy builder fallback	v23.0.0	-
Removed	Btrfs storage driver on CentOS 7 and RHEL 7	v20.10	v23.0.0
Removed	Support for encrypted TLS private keys	v20.10	v23.0.0
Removed	Kubernetes stack and context support	v20.10	v23.0.0
Deprecated	Pulling images from non-compliant image registries	v20.10	-
Removed	Linux containers on Windows (LCOW)	v20.10	v23.0.0
Deprecated	BLKIO weight options with cgroups v1	v20.10	-
Removed	Kernel memory limit	v20.10	v23.0.0
Removed	Classic Swarm and overlay networks using external key/value stores	v20.10	v23.0.0
Removed	Support for the legacy `~/.dockercfg` configuration file for authentication	v20.10	v23.0.0
Deprecated	CLI plugins support	v20.10	-
Deprecated	Dockerfile legacy `ENV name value` syntax	v20.10	-
Removed	`docker build --stream` flag (experimental)	v20.10	v20.10
Deprecated	`fluentd-async-connect` log opt	v20.10	-
Removed	Configuration options for experimental CLI features	v19.03	v23.0.0

Ref: https://docs.docker.com/engine/deprecated/

Docker Machine

...

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 Personal, Pro, Team Business)
  - 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 and upper
  - Docker for Ubuntu,Debian
  - Docker for CentOS
  - Docker for Red Hat Enterprise
  - Docker for Fedora

Linux vs Windows vs MAC OS

Not Secure | boot2docker.io

Apps | NMac Ked - Mac ... | Medium | Infra | NOVA | jerkins | vagrant | Mesos | Vue | docker | NGINX | Taiwan | Kubernetes

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.

docker docs | Search the docs | Guides | Product manuals | Glossary | Reference | Samples

- Docker Enterprise Edition
- Docker Cloud
- Docker Compose
- Docker for Mac
- Docker for Windows
- Docker ID accounts
- Docker Machine
  - Machine overview
  - Install Machine

Get started with Docker Machine and a local VM

Estimated reading time: 13 minutes

Let's take a look at using `docker-machine` to create, use and manage a Docker host inside of a local virtual machine.

Prerequisite Information

With the advent of Docker for Mac and Docker for Windows as replacements for Docker Toolbox, we recommend that you use these for your primary Docker workflows. You can use these applications to run Docker natively on your local system without using Docker Machine at all. (See Docker for Mac vs. Docker Toolbox for an explanation on the Mac side.)

For now, however, if you want to create *multiple* local machines, you still need Docker Machine to create and manage machines for multi-node experimentation. Both Docker for Mac and Docker for Windows include the newest version of Docker Machine, so when you install either of these, you get `docker-machine`.

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>
```

- สั่งสร้าง docker-machine เครื่องใหม่

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

Workshop: Pull Image

ทำการ download image ubuntu ลงมาที่เครื่อง boot2docker โดยใช้คำสั่งดังนี้

```
docker image pull labdocker/alpine:3.13.6
```

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

```
docker image pull labdocker/cadvisor:0.40.0
```

Compare with Ubuntu image

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

```
ubuntu@ip-10-0-1-90:~$ docker image ls
REPOSITORY            TAG       IMAGE ID       CREATED          SIZE
labdocker/alpineweb   latest    75f4bf61fa65   51 minutes ago   62.1MB
labdocker/alpine      3.13.6    12adea71a33b   2 days ago       5.61MB
labdocker/ubuntu      latest    fb52e22af1b0   3 days ago       72.8MB
labdocker/cadvisor    0.40.0    1ed215a61956   8 weeks ago      86.9MB
ubuntu@ip-10-0-1-90:~$
```

Workshop: Pull Image

ทำการตรวจสอบประวัติของ image ที่ใช้งานผ่านคำสั่ง docker image history

docker image history labdocker/alpine:3.13.6

docker image history labdocker/alpineweb:latest

```
ubuntu@ip-10-0-1-90:~$ docker image history labdocker/alpine:3.13.6
IMAGE          CREATED      CREATED BY                                      SIZE      COMMENT
12adea71a33b   2 days ago   /bin/sh -c #(nop)  CMD ["/bin/sh"]              0B
<missing>      2 days ago   /bin/sh -c #(nop) ADD file:ecdfb91a737d6c292…   5.61MB
ubuntu@ip-10-0-1-90:~$     docker image history labdocker/alpineweb:latest
IMAGE          CREATED          CREATED BY                                      SIZE      COMMENT
75f4bf61fa65   53 minutes ago   /bin/sh -c #(nop)  EXPOSE 3000                  0B
<missing>      53 minutes ago   /bin/sh -c #(nop)  HEALTHCHECK &{["CMD-SHELL…   0B
<missing>      53 minutes ago   /bin/sh -c #(nop)  ENTRYPOINT ["node" "hello…   0B
<missing>      53 minutes ago   /bin/sh -c #(nop) WORKDIR /nodejs               0B
<missing>      53 minutes ago   /bin/sh -c #(nop) COPY file:7a1f27eb22f7a835…   263B
<missing>      53 minutes ago   /bin/sh -c mkdir /nodejs                        0B
<missing>      53 minutes ago   /bin/sh -c apk update &&     apk add nodejs …   56.5MB
<missing>      2 hours ago      /bin/sh -c #(nop)  ENV NODE_VERSION=v14.15.4…   0B
<missing>      2 hours ago      /bin/sh -c #(nop)  LABEL Description=NodeJS/…   0B
<missing>      2 hours ago      /bin/sh -c #(nop)  LABEL maintainer=Praparn …   0B
<missing>      2 days ago       /bin/sh -c #(nop)  CMD ["/bin/sh"]              0B
<missing>      2 days ago       /bin/sh -c #(nop) ADD file:ecdfb91a737d6c292…   5.61MB
ubuntu@ip-10-0-1-90:~$
```

General Command in Docker

Command	Description
docker attach	Attach local standard input, output, and error streams to a running container
docker build	Build an image from a Dockerfile
docker builder	Manage builds
docker checkpoint	Manage checkpoints
docker commit	Create a new image from a container's changes
docker config	Manage Docker configs
docker container	Manage containers
docker context	Manage contexts
docker cp	Copy files/folders between a container and the local filesystem
docker create	Create a new container
docker diff	Inspect changes to files or directories on a container's filesystem
docker events	Get real time events from the server
docker exec	Run a command in a running container
docker export	Export a container's filesystem as a tar archive
docker history	Show the history of an image
docker image	Manage images

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

General Command in Docker

docker import	Import the contents from a tarball to create a filesystem image
docker info	Display system-wide information
docker inspect	Return low-level information on Docker objects
docker kill	Kill one or more running containers
docker load	Load an image from a tar archive or STDIN
docker login	Log in to a Docker registry
docker logout	Log out from a Docker registry
docker logs	Fetch the logs of a container
docker manifest	Manage Docker image manifests and manifest lists
docker network	Manage networks
docker node	Manage Swarm nodes
docker pause	Pause all processes within one or more containers
docker plugin	Manage plugins
docker port	List port mappings or a specific mapping for the container
docker ps	List containers
docker pull	Pull an image or a repository from a registry
docker push	Push an image or a repository to a registry
docker rename	Rename a container

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

General Command in Docker

docker restart	Restart one or more containers
docker rm	Remove one or more containers
docker rmi	Remove one or more images
docker run	Run a command in a new container
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
docker service	Manage services
docker stack	Manage Docker stacks
docker start	Start one or more stopped containers
docker stats	Display a live stream of container(s) resource usage statistics
docker stop	Stop one or more running containers
docker swarm	Manage Swarm
docker system	Manage Docker
docker tag	Create a tag TARGET_IMAGE that refers to SOURCE_IMAGE
docker top	Display the running processes of a container
docker trust	Manage trust on Docker images
docker unpause	Unpause all processes within one or more containers

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

General Command in Docker

docker update	Update configuration of one or more containers
docker version	Show the Docker version information
docker volume	Manage volumes
docker wait	Block until one or more containers stop, then print their exit codes

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

Image, Repository and Container

...

Docker Deployment

Docker Image

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

Docker Image

- เรียกดู image ที่อยู่ภายในเครื่อง

docker images /docker image ls

```
ubuntu@ip-10-0-1-90:~$ docker image ls
REPOSITORY                TAG              IMAGE ID       CREATED          SIZE
labdocker/linenotify      bot_v2           7f073d0abe25   8 minutes ago    86.4MB
labdocker/alpineweb       latest           75f4bf61fa65   54 minutes ago   62.1MB
labdocker/alpine          3.13.6           12adea71a33b   2 days ago       5.61MB
labdocker/mariadb         10.5.12          b7b798989225   3 days ago       407MB
labdocker/ubuntu          latest           fb52e22af1b0   3 days ago       72.8MB
labdocker/cadvisor        0.40.0           1ed215a61956   8 weeks ago      86.9MB
labdocker/cluster         webservicelite   837c8f41c918   4 years ago      82.6MB
labdocker/alpinepython    2.7-onbuild      0f4df961b16b   4 years ago      475MB
ubuntu@ip-10-0-1-90:~$
```

- tag image ใหม่เพื่อให้ตรงตามความต้องการในการใช้งาน

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

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

Docker Image

- ตรวจสอบรายละเอียดของ image

docker images history <image id/image name>

```
ubuntu@ip-10-0-1-90:~$ docker image history labdocker/alpine:3.13.6
IMAGE          CREATED      CREATED BY                                      SIZE      COMMENT
12adea71a33b   2 days ago   /bin/sh -c #(nop)  CMD ["/bin/sh"]              0B
<missing>      2 days ago   /bin/sh -c #(nop) ADD file:ecdfb91a737d6c292…   5.61MB
ubuntu@ip-10-0-1-90:~$     docker image history labdocker/alpineweb:latest
IMAGE          CREATED          CREATED BY                                      SIZE      COMMENT
75f4bf61fa65   53 minutes ago   /bin/sh -c #(nop)  EXPOSE 3000                  0B
<missing>      53 minutes ago   /bin/sh -c #(nop)  HEALTHCHECK &{["CMD-SHELL…   0B
<missing>      53 minutes ago   /bin/sh -c #(nop)  ENTRYPOINT ["node" "hello…   0B
<missing>      53 minutes ago   /bin/sh -c #(nop) WORKDIR /nodejs               0B
<missing>      53 minutes ago   /bin/sh -c #(nop) COPY file:7a1f27eb22f7a835…   263B
<missing>      53 minutes ago   /bin/sh -c mkdir /nodejs                        0B
<missing>      53 minutes ago   /bin/sh -c apk update &&     apk add nodejs …   56.5MB
<missing>      2 hours ago      /bin/sh -c #(nop)  ENV NODE_VERSION=v14.15.4…   0B
<missing>      2 hours ago      /bin/sh -c #(nop)  LABEL Description=NodeJS/…   0B
<missing>      2 hours ago      /bin/sh -c #(nop)  LABEL maintainer=Praparn …   0B
<missing>      2 days ago       /bin/sh -c #(nop)  CMD ["/bin/sh"]              0B
<missing>      2 days ago       /bin/sh -c #(nop) ADD file:ecdfb91a737d6c292…   5.61MB
ubuntu@ip-10-0-1-90:~$
```

Repository (Registry)

- component ในการจัดเก็บ docker (image) ต่างๆรวมไปที่ศูนย์กลางเพื่อให้ docker engine บนเครื่องต่างๆมาเรียก image ไปใช้ทำงาน
- รองรับการเก็บ version ของ image อย่างเป็นระบบ
- สามารถให้ข้อมูลหรือคู่มือแนะนำการใช้งานกับผู้ download image ได้
- มี official image ที่สร้างจากผู้พัฒนาโปรแกรมเอง
- Docker มีให้บริการ repository บน cloud แก่ผู้ใช้งาน
- Url: https://hub.docker.com/
- Free registry without cost
- Private repository (registry, trust registry)

Repository (Registry)

All legacy individual and organizational repository plans expire on their January 2021 billing cycle date. Read the FAQ

docker hub | Search for great content (e.g., mysql) | Explore | Pricing | Sign In | Sign Up

Build and Ship any Application Anywhere

Docker Hub is the world's easiest way to create, manage, and deliver your teams' container applications.

Get Started Today for Free

Already have an account? Sign In

Docker ID

Email

Password

Send me occasional product updates and announcements.

ฉันไม่ใช่โปรแกรมอัตโนมัติ

Sign Up

By creating an account, you agree to the Terms of Service, Privacy Policy, and Data Processing Terms.

Docker Hub is the world's largest library and community for container images

Browse over 100,000 container images from software vendors, open-source projects, and the community.

Official Images

NGINX | mongoDB | alpine | node | redis

Repository (Registry)

docker

Why Docker? Products Developers Pricing Company Sign In Get Started

Understanding Docker Hub Rate Limiting

Learn More

On November 20, 2020, rate limits anonymous and free authenticated use of Docker Hub went into effect. Anonymous and Free Docker Hub users are limited to 100 and 200 container image pull requests per six hours.

If you are affected by these changes you will receive this error message:

```
ERROR: toomanyrequests: Too Many Requests.
```

OR

```
You have reached your pull rate limit. You may increase the limit by authenticating and upgrading: https://www.docker.com/increase-rate-limits.
```

To increase your pull rate limits you can upgrade your account to a Docker Pro or Team subscription.

The rate limits of 100 container image requests per six hours for anonymous usage, and 200 container image requests per six hours for free Docker accounts are now in effect. Image requests exceeding these limits will be denied until the six hour window elapses.

NOTE: Docker Pro and Docker Team accounts continue to have unlimited access to pull container images from Docker Hub.

Related Resources

- Rate Limiting Documentation
- Resource Consumption Update FAQ
- Blog post on Resource Consumption and Image Retention Policy Updates
- Docker Subscription Pricing Page
- Pricing FAQ
- Guidance for Setting Up a Mirror
- Understanding Inner Loop Development and Pull Rates
- Docker Partner Information Request for Rate Limiting Exclusions

docker

Repository (Registry)

google/cadvisor

By google • Updated 12 hours ago

Analyzes resource usage and performance characteristics of running containers.

Container

Pulls 10M+

Overview Tags

cAdvisor

cAdvisor (Container Advisor) 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.

Dockefile: 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 Pull Command

```
docker pull google/cadvisor
```

Owner

google

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 xxxxxxx –e xxxx@xxx.xxx

Ex: docker login 10.38.7.248:8080 –u labdocker

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 สามเท่าจากเดิม*

Repository (Registry)

- Save image ออกมาเป็น tar file สำหรับ offline machine

```
docker image save -o xxx.tar image name
```

Ex: docker image save -o test.tar labdocker/alpinelab:1.0

- Load image จาก tar file สำหรับ offline machine

```
docker image load -i xxx.tar
```

Ex: docker image load -i test.tar

Workshop: 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 ตามตัวอย่างด้านล่าง

Workshop: 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 ไว้ขึ้น repository

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

Workshop: Build & Push Image

- ทดสอบทำการ save image ออกมาเป็นไฟล์ .tar

```
docker image save -o ~/alpineweb.tar
<account>/alpineweb:latest
```

- ดำเนินการลบและ load image จากไฟล์ .tar

```
docker image rm <account>/alpineweb:latest
docker image load -i ~/alpineweb.tar
```

Other service in hub.docker.com

Other service in hub.docker.com

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

Status.docker.com

docker system status

Docker system status real-time view

SUBSCRIBE

All Systems Operational Updated a few seconds ago

Docker Package Repositories	Operational
Docker Hub Web	Operational
Docker Hub Registry	Operational
Docker Hub Automated Builds	Operational
Docker.com Web	Operational
Docker Docs	Operational
Docker Community Forums	Operational
Docker Support Site	Operational

Metrics

Today Week Month

DOCKER HUB API RESPONSE TIME 505.16MS

Metrics

Today Week Month

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%

https://status.docker.com/

Docker Container

- Container คือชุดของ software layer ที่รันอิสระโดยอ้างอิงจาก image (run)
- ใช้สำหรับสร้างสภาพแวดล้อมที่จำเป็นต้องใช้ในการรันหรือพัฒนาโปรแกรม
- เมื่อพัฒนาโปรแกรมเสร็จเรียบร้อยแล้ว หรือต้องการ backup container สามารถสั่งเก็บ container ชุดปัจจุบันไปเป็น image เพื่อรอการเรียกใช้งานต่อไป (commit)

Docker: The Next-Gen of Virtualization

Docker Container

- Container State:

Docker Container

- Container Different (Compare with Image):

docker container diff <container name>

```
ubuntu@ip-10-0-1-90:~$ docker container diff nodejs
C /nodejs
A /nodejs/test.js
C /root
A /root/.ash_history
ubuntu@ip-10-0-1-90:~$ docker container ls
CONTAINER ID   IMAGE                        COMMAND           CREATED          STATUS                    PORTS                                       NAMES
4b04b8ba10e0   labdocker/alpineweb:latest   "node hello.js"   48 seconds ago   Up 47 seconds (healthy)   0.0.0.0:3000->3000/tcp, :::3000->3000/tcp   nodejs
ubuntu@ip-10-0-1-90:~$
```

Image → difference → Container

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                C                   STIME               TTY                 TIME                CMD
root                3069                3043                0                   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 Container

- Container change name on-the-fly:

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

```
ubuntu@ip-10-0-1-90:~$ docker container ls
CONTAINER ID   IMAGE                        COMMAND           CREATED         STATUS                     PORTS                                       NAMES
4b04b8ba10e0   labdocker/alpineweb:latest   "node hello.js"   3 minutes ago   Up 49 seconds (healthy)    0.0.0.0:3000->3000/tcp, :::3000->3000/tcp   nodejs
ubuntu@ip-10-0-1-90:~$ docker container rename nodejs nodejsnew
ubuntu@ip-10-0-1-90:~$ docker container ls
CONTAINER ID   IMAGE                        COMMAND           CREATED         STATUS                        PORTS                                       NAMES
4b04b8ba10e0   labdocker/alpineweb:latest   "node hello.js"   3 minutes ago   Up About a minute (healthy)   0.0.0.0:3000->3000/tcp, :::3000->3000/tcp   nodejsnew
ubuntu@ip-10-0-1-90:~$
```

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>
```

Workshop: Run Container

- <u>Interactive NODEJS</u>
- สั่งรัน 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
```

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

```
docker container ps -a
```

- Exist with Ctrl+C

Workshop: Run Container

- <u>DAEMON LINE BOT</u>

ธนาคารแห่งประเทศไทย

ข้อมูลทั่วไปของ API

ประเภทของ API :	REST
ประเภทของข้อมูล :	JSON
ความปลอดภัย :	PUBLIC

ตำแหน่งที่อยู่ของ API

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

พารามิเตอร์ใน Header

api-key : xk3h6ash-ahw5l3mch6hl3-2hhg6l4mng2346l34l6

ข้อมูลพารามิเตอร์สำหรับการใช้งาน API

ชื่อตัวแปร	ประเภท	คำอธิบาย
start_period	Datetime	วันที่เริ่มต้นงวดข้อมูล (YYYY-MM-DD) เช่น 2017-06-30
end_period	Datetime	วันที่สิ้นสุดงวดข้อมูล (YYYY-MM-DD) เช่น 2017-06-30

ข้อมูลพารามิเตอร์ที่จะได้รับ

ชื่อ	ประเภท	คำอธิบาย
report_name_eng	String	ชื่อรายงาน ภาษาอังกฤษ
rereport_name_th	String	ชื่อรายงาน ภาษาไทย
report_uoq_name_eng	String	หน่วย ภาษาอังกฤษ
report_uoq_name_th	String	หน่วย ภาษาไทย
report_source_of_data	String	แหล่งที่มาของข้อมูล
report_remark_eng	String	หมายเหตุ รายงานภาษาอังกฤษ
report_remark_th	String	หมายเหตุ รายงานภาษาไทย
last_updated	String	วันที่แก้ไขข้อมูลล่าสุด
period	String	วันที่งวดข้อมูล
rate	String	อัตรา

Workshop: Run Container

- Ex 2: DAEMON LINE BOT

Workshop: Run Container

- DAEMON LINE BOT

```
docker@labdocker:~$ docker container run --rm --name linebot -e "TITLE=Line BOT (Bank of Thailand)" -e "TOKEN=EIeqyillzkzpaCo1ROrebZTcGbIyzDZHpOjcdd0t6CX" labdocker/linenotify:bot_v1
null
{"statusCode":200,"body":"{\"status\":200,\"message\":\"ok\"}","headers":{"server":"nginx","date":"Sat, 20 Jan 2018 03:12:22 GMT","content-type":"application/json;charset=UTF-8","transfer-encoding":"chunked","connection":"keep-alive","keep-alive":"timeout=3","x-ratelimit-limit":"1000","x-ratelimit-imagelimit":"50","x-ratelimit-remaining":"998","x-ratelimit-imageremaining":"50","x-ratelimit-reset":"1516421345"},"request":{"uri":{"protocol":"https:","slashes":true,"auth":null,"host":"notify-api.line.me","port":443,"hostname":"notify-api.line.me","hash":null,"search":null,"query":null,"pathname":"/api/notify","path":"/api/notify","href":"https://notify-api.line.me/api/notify"},"method":"POST","headers":{"Content-Type":"application/x-www-form-urlencoded","authorization":"Bearer EIeqyillzkzpaCo1ROrebZTcGbIyzDZHpOjcdd0t6CX","content-length":466}}}
"{\"status\":200,\"message\":\"ok\"}"
docker@labdocker:~$
```

```javascript
const request = require('request');
const qs = require('querystring');
const TITLE=process.env.TITLE;
const KEY =process.env.KEY; //Input API-KEY to Header
//const KEY="U9G1L457H6DCugT7VmBaEacbHV9RX0PySO05cYaGsm";
const URL =process.env.URL; //Input URL for Request
//const URL="https://iapi.bot.or.th/Stat/Stat-ReferenceRate/DAILY_REF_RATE_V1/?";
//const TOKEN = 'EIeqyillzkzpaCo1ROrebZTcGbIyzDZHpOjcdd0t6CX';
const TOKEN=process.env.TOKEN; //Input TOKEN LINE
//Set Date
var dateFormat = require('dateformat');
var daynow = new Date(); // Today!
daynow.setDate(daynow.getDate() - 1); // Yesterday!
var daynow=dateFormat(daynow, "yyyy-mm-dd");
var DAYSTARTPERIOD=daynow;
var DAYENDPERIOD=daynow;
//Set Date
var stickerPkg=2; //stickerPackageId
var stickerId=161; //stickerId
var Message="";
//Setup QueryString
var querystring = qs.stringify({
    start_period: DAYSTARTPERIOD,
    end_period: DAYENDPERIOD
});
//Setup QueryString
function callback(error, response, body) {
  //console.log(URL+querystring);
  //console.log('Response Code: '+ response.statusCode);
  //console.log(body)
  if (!error && response.statusCode == 200) {
    var info = JSON.parse(body);
```

Workshop: Run Container

- <u>DAEMON LINE BOT</u>
- สั่งรัน container แบบ Daemon โดยตั้งชื่อว่า linebot เพื่อทำการดึงข้อมูลจาก API ของธนาคารแห่งประเทศไทยออกมาแล้วส่ง LINE Notify ไปหา Token Key ที่กำหนด

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

Workshop: Run Container

- <u>Detach Mode NODEJS</u>
- 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
```

Workshop: Run Container

- <u>Detach Mode NODEJS</u>
- 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
```

Workshop: Run Container

- Detach Mode NODEJS
- ทำการตรวจสอบ process ภายใน container

```
docker container top nodejs
```

```
docker container stats nodejs
```

- ทำการตรวจสอบความแตกต่างระหว่าง container กับ image ต้นฉบับ

```
docker container diff nodejs
```

- ตรวจสอบการ map port container

```
docker container port nodejs
```

- เปลี่ยนแปลงชื่อ container แบบ online

```
docker container rename nodejs nodejsnew
```

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/');
```

Workshop: Run Container

- <u>Detach Mode Python</u>
- สั่งสร้าง container แบบ detach (daemon) โดยตั้งชื่อว่า pythonและ map network 5000:5000

```
docker create run -t --name python -p 5000:5000 \
labdocker/cluster:webservicelite
```

- ทำการรัน

```
docker container ls –a
docker container start python
```

- ทดสอบเปิดหน้าเว็บบน url: http://<ip address>:5000
- ทดสอบ Shell เข้าสู่ container ด้วยคำสั่ง

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

```
docker container attach python
```

Workshop: Run Container

- Detach Mode Python

```python
from flask import Flask
import os
import time
app = Flask(__name__)

@app.route('/')
def hello():
    return '<H1> Welcome Page from Container Python Lab </H1>Checkpoint Date/Time: ' + time.strftime("%c") +'\n'

if __name__ == "__main__":
    app.run(host="0.0.0.0", port=5000, debug=True)
```

CPU, Memory & I/O

...

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)(CFS scheduler)

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

CPU

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

- NUMA (non-uniform memory access) architecture

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

- กำหนด cpu real-time runtime (ms) (nice system call)

```
--cap-add=sys_nice --cpu-rt-runtime=<value>
```

- กำหนด cpu run-time priority (0-99) (nice system call)

```
--cap-add=sys_nice --ulimit rtprio=xx
```

NVIDIA GPU

- Docker สามารถใช้งาน GPU ของ NVIDIA ในการนำมาประมวลผลได้โดยต้องทำการติดตั้ง nvidia-container-runtime เพิ่มเติมดังนี้ (Need restart)

```
apt-get install nvidia-container-runtime
which nvidia-container-runtime-hook
```

- Check gpu available for use in system by command

```
docker run -it --rm --gpus all ubuntu nvidia-smi
```

- Run docker container with gpu:

```
docker run -it --rm --gpus device=xxxx ubuntu nvidia-smi
```

Ref:https://github.com/NVIDIA/nvidia-docker

Workshop: CPU Configure

- Download cAdvisor

```
docker image pull labdocker/cadvisor:0.40.0
```

- สั่งรัน 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:0.40.0
```

Workshop: CPU Configure

- cAdvisor

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
```

Workshop: CPU Configure

- Result

```
CONTAINER           CPU %               MEM USAGE / LIMIT     MEM %               NET I/O             BLOCK I/O           PIDS
01109e1c7b0f        70.65%              76KiB / 995.4MiB      0.01%               648B / 0B           0B / 0B             1
909882f27a59        30.26%              40KiB / 995.4MiB      0.00%               648B / 0B           0B / 0B             1
e247e2e8240a        0.05%               16.86MiB / 995.4MiB   1.69%               718kB / 39.4MB      17.6MB / 0B         11
^C
docker@labdocker:~$ docker ps
CONTAINER ID        IMAGE                       COMMAND                  CREATED             STATUS              PORTS
           NAMES
01109e1c7b0f        labdocker/busybox:latest    "md5sum /dev/urandom"    2 minutes ago       Up 2 minutes
           Share_70
909882f27a59        labdocker/busybox:latest    "md5sum /dev/urandom"    3 minutes ago       Up 3 minutes
           Share_30
e247e2e8240a        labdocker/cadvisor:latest   "/usr/bin/cadvisor..."   4 minutes ago       Up 4 minutes        0.0.0.0:8080->
8080/tcp   cadvisor
docker@labdocker:~$
```

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
```

Quiz ?

- Result

```
CONTAINER           CPU %               MEM USAGE / LIMIT     MEM %               NET I/O             BLOCK I/O           PIDS
02d0b2973cff        99.93%              88KiB / 995.4MiB      0.01%               648B / 0B           0B / 0B             1
b1aebf2a3d71        99.88%              84KiB / 995.4MiB      0.01%               648B / 0B           0B / 0B             1
e247e2e8240a        2.02%               19.34MiB / 995.4MiB   1.94%               1.07MB / 66.7MB     17.6MB / 0B         11
^C
docker@labdocker:~$ docker ps
CONTAINER ID        IMAGE                       COMMAND                  CREATED             STATUS              PORTS
            NAMES
02d0b2973cff        labdocker/busybox:latest    "md5sum /dev/urandom"    41 seconds ago      Up 40 seconds
            Share_70
b1aebf2a3d71        labdocker/busybox:latest    "md5sum /dev/urandom"    42 seconds ago      Up 42 seconds
            Share_30
e247e2e8240a        labdocker/cadvisor:latest   "/usr/bin/cadvisor..."   6 minutes ago       Up 6 minutes        0.0.0.0:8080->
8080/tcp   cadvisor
docker@labdocker:~$
```

Memory

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

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
```

- disable oom kill

```
--oom-kill-disable =0
```

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 แล้วกำหนดเป็น “--memory-reservation”
- ทำ capacity testing เพื่อกำหนด maximum memory ที่ต้องการในการใช้งาน (+30%) แล้วกำหนด “--memory“ (Limit)
- สำหรับ critical system ให้กำหนด “--memory-reservation” เท่ากับ “--memory“ (Limit)
- Memory Priority ? (Still waiting)

I/O

- โดย default ทุก container จะมีโอกาสใช้ I/O เท่าๆกัน (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
```

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                          tty          cpu
device       r/s    w/s   kr/s   kw/s wait actv  svc_t  %w  %b  tin tout  us sy wt id
fd0          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   49.5   0   0
sd6          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
nfs49        0.0    0.0    0.0    0.0  0.0  0.0   15.1   0   0
nfs53        0.0    0.0    0.4    0.0  0.0  0.0   24.5   0   0
nfs54        0.0    0.0    0.0    0.0  0.0  0.0    6.3   0   0
nfs55        0.0    0.0    0.0    0.0  0.0  0.0    4.9   0   0
```

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

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 x 6 = 1060 iops

- Raid disk performance:
  - ((raw iops x %write)/penalty)+(raw x %read)
  - Normal situation (write 30%, read 70%)
  - **Ex:** Raid 5 ((1060 x .3)/4)+(1060x.7) □ 821.6 iops
  - **Ex**: Raid 1 ((1060 x .3)/2)+(1060x.7) □ 901 iops

Network

...

Network

- Software define network by design (virtual switch) (Support IPv4/6)
- default docker จะจัดเตรียม network มาให้สามรูปแบบ

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

- **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)

Network

- Additional network type
- **MACVLAN/IPVLAN (L2)**

Network

- Additional network type
- **IPVLAN (L3 Mode)**

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

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.

Network

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

Network

- docker network inspect bridge

192.168.99.100 - PuTTY

```
docker@labdocker:~$ docker network inspect bridge
[
    {
        "Name": "bridge",
        "Id": "5c03ed5297b82ab55b592d36324be256190b48267780a12bb603da9fd0c13887",
        "Scope": "local",
        "Driver": "bridge",
        "IPAM": {
            "Driver": "default",
            "Options": null,
            "Config": [
                {
                    "Subnet": "172.17.0.0/16"
                }
            ]
        },
        "Containers": {
            "303c0f349f5ae46873f0dfb1440cae34a30d7e58e187c448f95842c475fb770f": {
                "Name": "cadvisor",
                "EndpointID": "5aa767acbcaa4d64095778b1daca4823b92dd497fa6f4aaa436b
9032e475fcc2",
                "MacAddress": "02:42:ac:11:00:02",
                "IPv4Address": "172.17.0.2/16",
                "IPv6Address": ""
            }
        },
        "Options": {
            "com.docker.network.bridge.default_bridge": "true",
            "com.docker.network.bridge.enable_icc": "true",
            "com.docker.network.bridge.enable_ip_masquerade": "true",
            "com.docker.network.bridge.host_binding_ipv4": "0.0.0.0",
            "com.docker.network.bridge.name": "docker0",
            "com.docker.network.driver.mtu": "1500"
        }
    }
]
docker@labdocker:~$
```

Network

- Option เกี่ยวกับ network สำหรับสั่ง run container

--dns= x.x.x.x (Default --name,--net-alias,--link also dns internal docker)
--network="<bridge/none/host/custom>"
--network-alias = "xxxx"
--add-host="xxxx"
--mac-address="xx:xx:xx:xx"
--ip="x.x.x.x"
--ipv6="xx:xx:xx:xx"
–link-local-ip="x.x.x.x"
-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
```

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
  - --attachable = กำหนดให้สามารถต่อ container เข้ามาในเน็ตเวิร์คได้แบบ manual
  - --aux-address = กำหนด ip address (v4/v6) สำหรับ network driver
  - --config-from= copy configuration จาก virtual switch อื่นๆ
  - --driver, -d = driver ในการใช้งาน virtual switch (bridge, none, macvlan etc)
  - --ingress = สร้างเพื่อใช้บน switch เพื่อทำ service routeing-mesh
  - --internal = สร้างใช้ภายใน docker เท่านั้น
  - --scope = กำหนด scope การใช้งาน (host,cluster)

Network

- สามารถเพิ่มเติม option ในการสร้าง virtual switch เพิ่มเติม
  - --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

Network

host
c1
eth0: 172.17.0.2
c2
eth0: 10.0.0.2
c3
eth0: 10.0.0.254
veth
veth
veth
docker0
my_bridge
eth0: 192.168.1.2

Network

- การเพิ่ม network ใน container

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

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

- การลด network ใน container

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

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

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)

Workshop: Network Configure

- <u>Part 1: Reverse Proxy Network (DNS)</u>
- Purpose: ทำการ Deploy nodejs webserver ด้วย solution proxy ของ nginx (load balance / reverse proxy)

- **public network** เพื่อให้บุคคลภายนอกเข้าใช้งานผ่าน nginx server (reverse proxy)
  - IP Address: 192.168.100.0/24
  - Range Address: 192.168.100.128 – 192.168.100.256
  - MTU: 1500
- **private network** เพื่อให้ nginx server เข้าเรียกใช้งาน nodejs web server
  - IP Address: 192.168.101.0/24
  - Range Address: 192.168.101.128 – 192.168.101.256
  - MTU: 9000

Workshop: Network Configure

Workshop: Network Configure

Nginx Server A
Plain Deployment

Nginx Reverse Proxy

Nginx SSL Offloading

Workshop: Network Configure

Workshop: Network Configure

Workshop: Network Configure

Workshop: Network Configure

- Normal HTTP/1.1

Workshop: Network Configure

- HTTP/2 HTTPS (Old word: SPDY)

Workshop: Network Configure

Workshop: Network Configure

- ทำการสร้าง 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

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

Workshop: Network Configure

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

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 #
```

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         labdocker.com.crt;
        ssl_certificate_key     labdocker.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;
        }
}
```

Workshop: Network Configure

- Bonus: Kong API Gateway the basic

Network

- 802.1q Tagging (MACVLAN)

Demo: MACVLAN

- Reverse Proxy Network (MACVLAN) (This need docker-machine)
- Purpose: ทำการ Deploy nodejs webserver ด้วย solution proxy ของ nginx ผ่าน Macvlan
- **Macvlan:**
  - Interface: eth1
  - IP Address: 192.168.57.0/24
  - Gateway: 192.168.57.1 ⇒ Your Machine
  - Labdocker: 192.168.57.100 ⇒ Default
  - Range Address: 192.168.57.192/28
    - 192.168.57.193 – 192.168.57.206

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

Medium | jerkins | vagrant | Mesos | Vue | docker | NGINX | Taiwan | Kubernetes | PWA_Progressive_W... | MYSQL_Cluster | GeneralKB | Nod

- **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.

Demo: MACVLAN

Demo: MACVLAN

- ทำการสร้าง Switch สำหรับเชื่อมต่อ Macvlan
  - docker network create -d macvlan \
    --subnet=192.168.57.0/24 \
    --ip-range=192.168.57.192/28 \
    --gateway=192.168.57.1 \
    -o parent=eth1 macvlanlab

- ในกรณีต้องการทำ Tag VLAN (802.1q) จาก Switch/Router เข้าสู่ Docker Host และทำการสร้าง Sub Interface สามารถทำได้ดังนี้
  - docker network create -d macvlan \
    --subnet=192.168.57.0/24 \
    --ip-range=192.168.57.192/28 \
    --gateway=192.168.57.1 \
    -o parent=xxx.<vlan id> macvlanlab

Demo: MACVLAN

- สร้างเครื่อง Webserver (nodejs) และ Nginx
  - **Web1**
  - docker run -dt --name web1 --net macvlanlab \
    --net-alias web1 --ip 192.168.57.193 labdocker/alpineweb:web1 node hello.js
  - **Web2**
  - docker run -dt --name web2 --net macvlanlab \
    --net-alias web2 --ip 192.168.57.194 labdocker/alpineweb:web2 node hello.js
  - **NGINX**
  - docker run -dt --name nginx --net macvlanlab \
    --ip 192.168.57.195 labdocker/nginx:labnetworkhttp2

Demo: MACVLAN

- Internal test:

```
...10-0-1-55: ~ — -bash ...   ...emultinodejs — -bash   ...ute.amazonaws.com ...   ...chine ssh labdocker

docker@labdocker:~$ docker container exec -it nginx curl http://web1:3000
Hello World Container Docker for Nodejs Number 1
docker@labdocker:~$ 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@labdocker:~$ docker container exec -it nginx curl http://web2:3000
Hello World Container Docker for Nodejs Number 2
docker@labdocker:~$ 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@labdocker:~$ 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@labdocker:~$ 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@labdocker:~$
```

Demo: MACVLAN

- External Test:

```
...10-0-1-55: ~ — -bash ...   ...emultinodejs — -bash   ...ute.amazonaws.com ...
praparns-MacBook-Pro:~ praparn$ ping 192.168.99.193
PING 192.168.99.193 (192.168.99.193): 56 data bytes
64 bytes from 192.168.99.193: icmp_seq=0 ttl=64 time=0.346 ms
64 bytes from 192.168.99.193: icmp_seq=1 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:~ praparn$
praparns-MacBook-Pro:~ praparn$ ping 192.168.99.194
PING 192.168.99.194 (192.168.99.194): 56 data bytes
64 bytes from 192.168.99.194: icmp_seq=0 ttl=64 time=0.365 ms
64 bytes from 192.168.99.194: icmp_seq=1 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:~ praparn$
praparns-MacBook-Pro:~ praparn$ ping 192.168.99.195
PING 192.168.99.195 (192.168.99.195): 56 data bytes
64 bytes from 192.168.99.195: icmp_seq=0 ttl=64 time=0.357 ms
64 bytes from 192.168.99.195: icmp_seq=1 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:~ praparn$
praparns-MacBook-Pro:~ praparn$ curl http://192.168.99.195:8080/nodejs
Hello World Container Docker for Nodejs Number 1
praparns-MacBook-Pro:~ praparn$ curl http://192.168.99.195:8080/nodejs
Hello World Container Docker for Nodejs Number 2
praparns-MacBook-Pro:~ praparn$ curl http://192.168.99.193:3000
Hello World Container Docker for Nodejs Number 1
praparns-MacBook-Pro:~ praparn$ curl http://192.168.99.194:3000
Hello World Container Docker for Nodejs Number 2
praparns-MacBook-Pro:~ praparn$ curl https://192.168.99.195:8443/nodejs -k
Hello World Container Docker for Nodejs Number 1
praparns-MacBook-Pro:~ praparn$ curl https://192.168.99.195:8443/nodejs -k
Hello World Container Docker for Nodejs Number 2
praparns-MacBook-Pro:~ praparn$
```

Network

- Network across host (Overlay Network)

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.

Volume

...

Volume

- ตามปกติแล้ว docker จะเก็บข้อมูลทุกอย่างเอาไว้ภายใน container
  - /var/log
  - /sys/data
  - /etc/nagios/
  - /etc/mysql
  - Etc.

- สำหรับการใช้งาน container ใน production environment docker ได้สร้าง tool สำหรับอำนวยความสะดวกในการใช้งานข้อมูลร่วมกันในหลายๆกรณี อาทิเช่น
  - การ share ข้อมูลร่วมกันระหว่าง container
  - การ share ข้อมูลร่วมกันระหว่าง container / host
  - จัดเก็บ configuration / log / data ของทุกๆ container ไว้ที่ศูนย์กลาง
  - data migration
  - backup / restore

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))

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

- ตรวจสอบการคอนฟิกบน container (inspect)

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

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
```

Volume

- การใช้ 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

Volume

- การลบ volume ทิ้งจาก container ให้ใช้คำสั่ง docker volume rm เพื่อลบ volume

```
docker volume rm <volume name>
```

- Third Party Volume Plugin

```
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
```

Volume

- NFS3:

```
docker service create -d \
  --name nfs-service \
  --mount
'type=volume,source=nfsvolume,target=/app,volume-driver=local,volume-opt=type=nfs,volume-opt=device=:/var/docker-nfs,volume-opt=o=addr=10.0.0.10' nginx:latest
```

- NFS4:

```
docker service create -d \
    --name nfs-service \
    --mount
'type=volume,source=nfsvolume,target=/app,volume-driver=local,volume-opt=type=nfs,volume-opt=device=:/var/docker-nfs,"volume-opt=o=addr=10.0.0.10,rw,nfsvers=4,async"' nginx:latest
```

Volume

- CIFS/SAMBA:

```
docker volume create \
--driver local \
--opt type=cifs \
--opt device=//uxxxxx.your-server.de/backup --opt
o=addr=uxxxxx.your-server.de,username=uxxxxxxx,password=*****,file_m
ode=0777,dir_mode=0777 --name cif-volume
```

Volume

lugin	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.
BeeGFS Volume Plugin	An open source volume plugin to create persistent volumes in a BeeGFS parallel file system.
Blockbridge plugin	A volume plugin that provides access to an extensible set of container-based persistent storage options. It supports single and multi-host Docker environments with features that include tenant isolation, automated provisioning, encryption, secure deletion, snapshots and QoS.
Contiv Volume Plugin	An open source volume plugin that provides multi-tenant, persistent, distributed storage with intent based consumption. It has support for Ceph and NFS.
Convoy plugin	A volume plugin for a variety of storage back-ends 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.
Flocker plugin	A volume plugin that provides multi-host portable volumes for Docker, enabling you to run databases and other stateful containers and move them around across a cluster of machines.
Fuxi 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.
gce-docker plugin	A volume plugin able to attach, format and mount Google Compute persistent-disks.
GlusterFS plugin	A volume plugin that provides multi-host volumes management for Docker using GlusterFS.
Horcrux 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.
HPE 3Par Volume Plugin	A volume plugin that supports HPE 3Par and StoreVirtual iSCSI storage arrays.
IPFS Volume Plugin	An open source volume plugin that allows using an ipfs filesystem as a volume.
Keywhiz plugin	A plugin that provides credentials and secret management using Keywhiz as a central repository.
Local Persist Plugin	A volume plugin that extends the default local driver's functionality by allowing you specify a mountpoint anywhere on the host, which enables the files to *always persist*, even if the volume is removed via docker volume rm.
NetApp Plugin(nDVP)	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.
Netshare plugin	A volume plugin that provides volume management for NFS 3/4, AWS EFS and CIFS file systems.
Nimble Storage Volume Plugin	A volume plug-in that integrates with Nimble Storage Unified Flash Fabric arrays. The plug-in 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 EBS to name a few.
Portworx 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 shared-nothing storage backend that works with any docker scheduler.
Quobyte Volume Plugin	A volume plugin that connects Docker to Quobyte's data center file system, a general-purpose scalable and fault-tolerant storage platform.
REX-Ray 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 EMC.
Virtuozzo Storage and Ploop plugin	A volume plugin with support for Virtuozzo Storage distributed cloud file system as well as ploop devices.
VMware vSphere Storage Plugin	Docker Volume Driver for vSphere enables customers to address persistent storage requirements for Docker containers in vSphere environmen

Ref: https://docs.docker.com/engine/extend/legacy_plugins/

Volume

- การ backup volume จาก 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
```

- การ restore volume จาก 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"
```

Volume

- 3. tmpfs mount ใช้เพื่อจัดเก็บข้อมูลใน memory ไว้บน non-persistency 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

Workshop: Share Volume

- <u>Part 1: Host Path debug program</u>
- Purpose: ทำการ Map volume เพื่อนำ source code เข้าไปทำการ debug/run บน container
- Example on WorkShop: ”Monitor and LINE notify container”

LINE

Workshop: Share Volume

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

Workshop: Share Volume

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

Workshop: Share Volume

```javascript
const Monitor = require('monitor');
const request = require('request');
const TITLE=process.env.TITLE;
const INTERVAL=process.env.INTERVAL;
const HEAP_HIGH =process.env.HEAP_HIGH;
const MEM_HIGH =process.env.MEM_HIGH;
const SH_OS =process.env.SH_OS;
const TOKEN = process.env.TOKEN;
const critical_StickerPkg = 2;
const critical_StickerId = 39;
const information_StickerPkg = 2;
const information_StickerId = 34;
var options = {
  probeClass: 'Process',
  initParams: {
    pollInterval: INTERVAL
  }
}
var processMonitor = new Monitor(options);
var stickerPkg=0; //stickerPackageId
var stickerId=0; //stickerId
processMonitor.on('change', () => {
  var sendNotify="N" //final decision to send notify
  var heapWarning="N" //flag for heap memory warning
  var memoryWarning="N" //flag for memory warning
```

```javascript
var processMonitor = new Monitor(options);
var stickerPkg=0; //stickerPackageId
var stickerId=0; //stickerId
processMonitor.on('change', () => {
  var sendNotify="N" //final decision to send notify
  var heapWarning="N" //flag for heap memory warning
  var memoryWarning="N" //flag for memory warning
  //collect OS information
  var hostName = processMonitor.get('hostname');
  var pID = processMonitor.get('pid');
  var platForm = processMonitor.get('platform');
  var upTime = processMonitor.get('uptime');
  var type = processMonitor.get('type');
  var totalMem = processMonitor.get('totalmem');
  totalMem=Number(((totalMem/1024)/1024).toFixed(1)); //Convert to MB
  var heapTotal = processMonitor.get('heapTotal');
  heapTotal=Number(((heapTotal/1024)/1024).toFixed(1)); //Convert to MB
  var titleMSG="\n"+TITLE+"("+pID+")"
  var osMSG="";
  var warnMSG="";
  //collect OS information
  //collect memory information
  var heapUsed = processMonitor.get('heapUsed');
  heapUsed=Number(((heapUsed/1024)/1024).toFixed(1)); //Convert to MB
  if(HEAP_HIGH<((heapUsed/heapTotal)*100)){heapWarning="Y";}
  var freeMem = processMonitor.get('freemem');
  freeMem=Number(((freeMem/1024)/1024).toFixed(1));
  var usedMem = totalMem-freeMem;
  usedMem=Number(usedMem.toFixed(1)); //Convert to MB
  if(MEM_HIGH<((usedMem/totalMem)*100)){memoryWarning="Y";}
  //collect memory information
```

Workshop: Share Volume

praparnlueangphoonlap — Terminal MAC Pro — ssh ◂ docker-machine ssh labdocker — 102×25

Terminal MAC Pro — ssh ◂ docker-machine ss... | ~ — Terminal MAC Pro — zsh | ~ — Terminal MAC Pro — -bash

```
mit":"50","x-ratelimit-remaining":"997","x-ratelimit-imageremaining":"50","x-ratelimit-reset":"1501996
936"},"request":{"uri":{"protocol":"https:","slashes":true,"auth":null,"host":"notify-api.line.me","po
rt":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":{"Con
tent-Type":"application/x-www-form-urlencoded","authorization":"Bearer EIeqyillzkzpaCo1ROrebZTcGbIyzDZ
HpOjcdd0t6CX","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":"chunk
ed","connection":"keep-alive","keep-alive":"timeout=3","x-ratelimit-limit":"1000","x-ratelimit-imageli
mit":"50","x-ratelimit-remaining":"996","x-ratelimit-imageremaining":"50","x-ratelimit-reset":"1501996
936"},"request":{"uri":{"protocol":"https:","slashes":true,"auth":null,"host":"notify-api.line.me","po
rt":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":{"Con
tent-Type":"application/x-www-form-urlencoded","authorization":"Bearer EIeqyillzkzpaCo1ROrebZTcGbIyzDZ
HpOjcdd0t6CX","content-length":290}}}
"{\"status\":200,\"message\":\"ok\"}"
^C
/nodejs #
```

Workshop: Share Volume

- Part 2: Container Volume

Log and Inspect

...

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 กำหนดให้ใส่วันและเวลาในล็อกไฟล์
  - --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

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)

```
{
  "exec-opts": ["native.cgroupdriver=systemd"],
  "log-driver": "json-file",
  "log-opts": {
    "max-size": "100m",
    "max-file": "10"
  },
  "storage-driver": "overlay2"
}
```

Log

Driver	Description
`none`	No logs are available for the container and `docker logs` 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 `syslog` facility. The `syslog` daemon must be running on the host machine.
`journald`	Writes log messages to `journald`. The `journald` 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 `fluentd` (forward input). The `fluentd` daemon must be running on the host machine.
`awslogs`	Writes log messages to Amazon CloudWatch Logs.
`splunk`	Writes log messages to `splunk` using the HTTP Event Collector.
`etwlogs`	Writes log messages as Event Tracing for Windows (ETW) events. Only available on Windows platforms.
`gcplogs`	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/

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
07af88d6368bdfd8a74608882963a50052e5c887fc32da812369d78068643e51
ubuntu@ip-10-0-1-55:~$
```

Inspect

- การตรวจสอบค่าคอนฟิกของ container / network / volume etc

```
docker inspect <container >
```

```
docker network inspect
```

```
docker volume inspect
```

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
```

Commit

...

Commit

- เมื่อต้องการเก็บ container ที่ทำการรันเรียบร้อยเป็น snap image เป็น generation สามารถทำได้ผ่านคำสั่ง “commit”

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

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

Docker Desktop

...

Docker Desktop

- GUI tool for developer/devops on your machine

Docker Desktop

- GUI tool for developer/devops on your machine

Docker Desktop

- GUI tool for developer/devops on your machine

Docker Desktop

Docker is Updating and Extending Our Product Subscriptions

SCOTT JOHNSTON
Aug 31 2021

Docker is used by millions of developers to build, share, and run any app, anywhere, and 55% of professional developers use Docker every day at work. In these work environments, the increase in outside attacks on software supply chains is accelerating developer demand for Docker's trusted content, including Docker Official Images and Docker Verified Publisher images. Finally, the rapid global growth in developers – to an estimated 45 million by 2030 – pushes us to scale sustainably so we may continue to provide an innovative, free Docker experience that developers love.

To meet these challenges, today we're announcing updates and extensions to our product subscriptions: **Personal, Pro, Team, and Business.** These updated product subscriptions provide the productivity and collaboration developers rely on with the scale, security, and trusted content businesses require, and do so in a manner sustainable for Docker.

What you need to know:

- We're introducing a **new** product subscription, Docker Business, for organizations using Docker at scale for application development and require features like secure software supply chain management, single sign-on (SSO), container registry access controls, and more.
- Our Docker Subscription Service Agreement includes a change to the terms for **Docker Desktop**:
  - Docker Desktop **remains free** for small businesses (fewer than 250 employees AND less than $10 million in annual revenue), personal use, education, and non-commercial open source projects.
  - It requires a paid subscription (**Pro**, **Team** or **Business**), starting at $5 per user per month, for professional use in larger businesses. You may directly purchase here, or share this post and our solution brief with your manager.
  - While the effective date of these terms is August 31, 2021, there is a grace period until January 31, 2022 for those that require a paid subscription to use Docker Desktop.
- Docker Pro, Docker Team, and Docker Business subscriptions **include** commercial use of Docker Desktop.
- The **existing** Docker Free subscription has been renamed Docker Personal.
- **No changes** to Docker Engine or any upstream open source Docker or Moby project.
- *Check out our FAQ for more information.*

Personal	Pro	Team	Business
1	1	5+	50+
Ideal for individual developers, education, open source communities, and small businesses.	Ideal for individual developers	Ideal for development teams	Ideal for medium and large businesses
$0	$5 /month	$7 /user/month minimum 5 seats	$21 /user/month
Includes Docker Desktop	Includes Docker Desktop	Includes Docker Desktop	Includes Docker Desktop

Recapture for Day 1

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

Q&A for Day 1

...

Docker:Zero to Hero (Day 2)

By Praparn Luengphoonlap
Email: eva10409@gmail.com

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

Dockerfile and Build

• • •

Dockerfile

- docker file คือการคำสั่งที่ใช้สร้าง image ขึ้นเพื่อใช้งานโดยเฉพาะ และสามารถกำหนดเป็นมาตรฐานสำหรับการทำงานในองค์กรโดยสร้าง text file ตาม syntax ที่ทาง docker กำหนดไว้
  - FROM
  - MAINTAINER
  - RUN (shell), [“exec”, “parameter1”,”parameter2”,”etc”]
  - CMD (shell),[“exec”,”parameter1”,”parameter2”,”etc”]
  - EXPOSE
  - ARGS
  - ENV
  - COPY /ADD (source) (Destination), [(source) (destination)]
  - ENTRYPOINT
  - WORKDIR
  - HEALTHCHECK
  - Etc(https://docs.docker.com/engine/reference/builder)
- เมื่อสร้าง dockerfile เสร็จแล้วสามารถสั่ง image ผ่านคำสั่ง build

docker build <option> <path of dockerfile>

Dockerfile

- .dockerignore file exclude some file/folder that not sent to build in image

```
# comment
*/temp*
*/*/temp*
temp?
*.md
!README*.md
```

Dockerfile

- Option สำหรับคำสั่ง docker image build (Partial)
  - --add-host — Add customer /etc/hosts
  - --build-arg — Set build-time variables
  - --cache-from — Image to consider
  - --cgroup-parent — Specific parent control group
  - --compress — Compress context with gzip
  - --cpu-shares — CPU shares
  - --cpuset-cpus — CPU no. for use
  - --memory, -m — Memory limit
  - --disable-content -trust — image verification
  - --file, -f — Specify docker file
  - --rm — Remove intermediate image after build finished
  - --force-rm — Force remove intermediate image
  - --network — Set network in build operate (RUN)
  - --no-cache — Never use cache
  - --pull — Always attempt pull new version image
  - --tag, -t — Tag image name

Ref: https://docs.docker.com/engine/reference/commandline/build/

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 รวบยอดใน 1 ชุดคำสั่ง
- ควรมีการลบ temp ที่เกิดขึ้นจากการติดตั้ง software เพื่อลดขนาดของ image
- จัดเรียง dockerfile ให้อ่านง่าย
  - apt-update && apt-install –y \
  - curl \
  - automake \
- ควรระบุ EXPOSE port ที่ออกแบบให้ใช้งานไว้ในทุกๆครั้ง

Ref: https://docs.docker.com/develop/develop-images/dockerfile_best-practices/

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/*
```

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 pgsql 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 .
```

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 ที่จำเป็นต้องใช้งาน

- พิจารณาการใช้ cache ของ image layer โดยเราสามารถ ignore cache ด้วย option “--no-cache=true”
  - ตามปกติ docker จะเปรียบเทียบ instruction ที่กำหนดไว้กับ image layer ที่
  - Consider: instruction, base image
- ควรระบุ HealthCheck เพื่อตรวจสอบการทำงานของ container

Docker Image Version

- Docker start to introduce standard of image manifest and schema on version 1.13.0 with version1 (deprecate)
- So standard image version is base on docker version that install on machine to build image
- In case that we run the deprecate version of docker image (schema v1). Docker will show warning during start container
- Converting image manifest is possible just "docker image pull" and "docker image push" from docker machine with up-to-date version. Docker will convert image format automatically
- Current docker image have 3 release of manifest
  - Manifest v1, Schema v1 (Deprecate)
  - Manifest v2, Schema v1 (Many vulnerability)
  - Manifest v2, Schema v2 (Recommend)

Docker Image Version

- Main feature of "Manifest v2, Scheme v2
  - Support multi-architecture (fat manifest) image for contain single image
  - Keep image configuration to hash as id and embed on image
- Verify manifest and schema version by command:

```
docker manifest inspect <image version>
```

  - application/vnd.docker.distribution.manifest.v1+json
  - application/vnd.docker.distribution.manifest.v2+json
  - application/vnd.docker.distribution.manifest.list.v2+json
  - application/vnd.docker.container.image.v1+json
  - application/vnd.docker.image.rootfs.diff.tar.gzip
  - application/vnd.docker.image.rootfs.foreign.diff.tar.gzip
  - application/vnd.docker.plugin.v1+json

Docker Image Version

- Verify manifest and schema version by command:

```
ubuntu@ip-10-0-1-65:~$ docker manifest inspect labdocker/alpine:latest
{
        "schemaVersion": 2,
        "mediaType": "application/vnd.docker.distribution.manifest.v2+json",
        "config": {
                "mediaType": "application/vnd.docker.container.image.v1+json",
                "size": 1472,
                "digest": "sha256:7731472c3f2a25edbb9c085c78f42ec71259f2b83485aa60648276d408865839"
        },
        "layers": [
                {
                        "mediaType": "application/vnd.docker.image.rootfs.diff.tar.gzip",
                        "size": 2810825,
                        "digest": "sha256:596ba82af5aaa3e2fd9d6f955b8b94f0744a2b60710e3c243ba3e4a467f051d1"
                }
        ]
}
ubuntu@ip-10-0-1-65:~$
```

How can check dockerfile ?

Praparn

Secure https://www.fromlatest.io/#/

Apps NMac Ked - Mac O... Medium Infra NOVA jerkins vagrant Mesos Vue docker NGINX Taiwan Kubernetes » Other Bookmarks

FROM:latest

Analyze About Contribute

```
1  FROM labdocker/alpine:latest
2  MAINTAINER Praparn Lueangphoonlap (eva10409@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

Instruction_Kubernet....txt Instruction_Kubernet....txt Show All

Workshop: DockerFile

- Part1: SCRATCH

docker-library / hello-world

Watch 29 | Star 214 | Fork 260

Code | Issues 0 | Pull requests 1 | Projects 0 | Wiki | Insights

No description, website, or topics provided.

68 commits | 1 branch | 0 releases | 9 contributors | MIT

Branch: master | New pull request | Create new file | Upload files | Find file | Clone or download

yosifkit Merge pull request #54 from infosiftr/1809 — Latest commit b9918f5 on Jan 1, 2019

amd64	Add Windows 1809 support	2 months ago
arm32v5	Update to the "get started" guide instead and update all binaries/tex...	6 months ago
arm32v7	Update to the "get started" guide instead and update all binaries/tex...	6 months ago
arm64v8	Update to the "get started" guide instead and update all binaries/tex...	6 months ago
greetings	Add initial multiarchitecture support via Debian cross-compiled binaries	2 years ago
i386	Update to the "get started" guide instead and update all binaries/tex...	6 months ago
ppc64le	Update to the "get started" guide instead and update all binaries/tex...	6 months ago
s390x	Update to the "get started" guide instead and update all binaries/tex...	6 months ago
.dockerignore	Add Nano Server 1803 variants	8 months ago
.travis.yml	Use new microsoft/nanoserver tags (sac2016)	a year ago
Dockerfile-linux.template	Add Nano Server 1803 variants	8 months ago
Dockerfile-windows.template	Add Windows 1809 support	2 months ago
Dockerfile.build	Add i386!	2 years ago
LICENSE	Remove ending year from LICENSE	7 months ago
Makefile	Add Windows 1809 support	2 months ago
README.md	Update generated README	11 months ago
generate-stackbrew-library.sh	Add Windows 1809 support	2 months ago
hello.c	Update to the "get started" guide instead and update all binaries/tex...	6 months ago
update.sh	Add Windows 1809 support	2 months ago

README.md

https://github.com/docker-library/hello-world

Maintained by: the Docker Community

This is the Git repo of the Docker "Official Image" for hola-mundo (not to be confused with any official hola-mundo image provided by hola-mundo upstream). See the Docker Hub page for the full readme on how to use this Docker image and for information regarding contributing and issues.

The full description from Docker Hub is generated over in docker-library/docs, specifically in docker-library/docs/hola-mundo.

```
instruction.txt  nginx  nodejs  python_restfulset  scratch
ubuntu@ip-10-0-1-104:~$ docker image build -t labdocker/hello-world:1.0 ~/docker-workshop-042019/Workshop-2-1-DockerFile/scratch/
Sending build context to Docker daemon   16.9kB
Step 1/3 : FROM scratch
 --->
Step 2/3 : COPY hello /
 ---> Using cache
 ---> 937200b29047
Step 3/3 : CMD ["/hello"]
 ---> Using cache
 ---> e759df7ef1b1
Successfully built e759df7ef1b1
Successfully tagged labdocker/hello-world:1.0
ubuntu@ip-10-0-1-104:~$ docker container run labdocker/hello-world:1.0

Hello from Docker!
This message shows that your installation appears to be working correctly.

To generate this message, Docker took the following steps:
 1. The Docker client contacted the Docker daemon.
 2. The Docker daemon pulled the "hello-world" image from the Docker Hub.
    (amd64)
 3. The Docker daemon created a new container from that image which runs the
    executable that produces the output you are currently reading.
 4. The Docker daemon streamed that output to the Docker client, which sent it
    to your terminal.

To try something more ambitious, you can run an Ubuntu container with:
 $ docker run -it ubuntu bash

Share images, automate workflows, and more with a free Docker ID:
 https://hub.docker.com/

For more examples and ideas, visit:
 https://docs.docker.com/get-started/

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

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

dockerfile

/nginx

dockerfile

Workshop: DockerFile

- ตรวจสอบ dockerfile ตามตัวอย่างด้านล่าง

```
FROM labdocker/alpine:3.13.6
LABEL maintainer="Praparn Lueangphoonlap (eva10409@gmail.com)"
LABEL Description="NodeJS/NGINX Build Container" Version="1.0"
ENV NODE_VERSION=v14.15.4 NPM_VERSION=v14.15.4
RUN apk update && \
    apk add nodejs nodejs-npm curl --no-cache && \
    rm -rf /var/cache/apk/*
RUN mkdir /nodejs
COPY hello.js /nodejs/
WORKDIR /nodejs
ENTRYPOINT ["node","hello.js"]
EXPOSE 3000
HEALTHCHECK --interval=15s --timeout=5s --start-period=10s --retries=3 \
    CMD curl -f http://localhost:3000/ || exit 1
```

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

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-gc7d905f [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 MiB in 15 packages
 ---> 8f2bf208ab86
Removing intermediate container b12c752ef6a0
Step 6 : RUN mkdir /nodejs
 ---> Running in 2ee51d1af642
 ---> d3edf2c2f511
Removing intermediate container 2ee51d1af642
Step 7 : COPY hello.js /nodejs/
 ---> f703ccbc7dd4
Removing intermediate container 952935d69cce
Step 8 : CMD nginx -c /etc/nginx/nginx.conf
 ---> Running in 67940385f5ac
 ---> edfba609e20a
```

Workshop: DockerFile

- Test run

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

Workshop: DockerFile

- ตรวจสอบ dockerfile ตามตัวอย่างด้านล่าง

```
FROM labdocker/alpine:3.13.6
LABEL maintainer="Praparn Lueangphoonlap (eva10409@gmail.com)"
LABEL Description="NodeJS/NGINX Build Container" Version="1.0"
ENV NODE_VERSION=v14.15.4 NPM_VERSION=v14.15.4
RUN mkdir -p /run/nginx
RUN apk update && \
    apk add nginx curl --no-cache && \
    rm /etc/nginx/conf.d/default.conf && \
    rm -rf /var/cache/apk/*
COPY nginx.conf /etc/nginx/nginx.conf
WORKDIR /usr/sbin
ENTRYPOINT ["nginx","-c","/etc/nginx/nginx.conf"]
EXPOSE 8080
HEALTHCHECK --interval=15s --timeout=5s --start-period=10s --retries=3 \
    CMD curl -f http://localhost:8080/ || exit 1
```

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

Workshop: DockerFile

- Test run

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

Workshop: DockerFile

- Part3: Python RESTFUL
- Basic Component

Workshop: DockerFile

- Optimize for WorkLoad

Workshop: DockerFile

- Restful WebService
  - /init

```
@app.route('/init')
def init():
    MAIN_DB.execute("DROP DATABASE IF EXISTS ACCTABLE")
    MAIN_DB.execute("CREATE DATABASE ACCTABLE")
    MAIN_DB.execute("USE ACCTABLE")
    sql = """CREATE TABLE users (
         ID int,
         USER char(30),
         DESCRIPE char(250)
     )"""
    MAIN_DB.execute(sql)
    db.commit()
    return "###########   Database Create New Account Table Done  ###########"
```

  - /insertuser

```
@app.route("/users/insertuser", methods=['POST'])
def add_users():
    req_json = request.get_json()
    MAIN_DB.execute("INSERT INTO ACCTABLE.users (ID, USER, DESCRIPE) VALUES (%s,%s,%s)", (req_json['uid'], req_json['user'], req_json['descripe']))
    #curl -i -H "Content-Type: application/json" -X POST -d '{"uid": "1", "user":"Praparn Luangphoonlap", "descripe":"System Engineer"}' http://<IP
    db.commit()
    return Response("########### Record was added ###########", status=200, mimetype='application/json')
```

Workshop: DockerFile

- Restful WebService
  - /removeuser/<uid>

```
@app.route('/users/removeuser/<uid>')
def remove_users(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')
```

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 ###########"
```

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;
        }
        }
```

Workshop: DockerFile

- Database / Database Cache

```yaml
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
```

Maria Database

Redis Key-Value Database

Workshop: DockerFile

- Dockerfile: WebService

```
FROM labdocker/alpinepython:2.7-onbuild
LABEL maintainer="Praparn Lueangphoonlap (eva10409@gmail.com)"
LABEL Description="NodeJS/NGINX Build Container" Version="1.0"
RUN apk update && \
    apk add curl --no-cache && rm -rf /var/cache/apk/*
EXPOSE 5000
CMD [ "python", "main.py" ]
HEALTHCHECK --interval=30s --timeout=5s --start-period=30s --retries=3 \
    CMD curl -f http://localhost:5000/ || exit 1
```

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

Workshop: DockerFile

- Dockerfile: Webcache

```
FROM labdocker/alpine:3.13.6
LABEL 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 nginx nginx curl --no-cache && rm -rf /var/cache/apk/*
COPY nginx.conf /etc/nginx/nginx.conf
WORKDIR /usr/sbin
ENTRYPOINT ["nginx","-c","/etc/nginx/nginx.conf"]
EXPOSE 8080
HEALTHCHECK --interval=30s --timeout=5s --start-period=30s --retries=3 \
    CMD curl -f http://localhost:8080/ || exit 1
```

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

Workshop: DockerFile

- Test run container

```
ubuntu@ip-10-0-1-90:~/docker-workshop-092021/Workshop-2-1-DockerFile/python_restfulset$ docker container ls
CONTAINER ID   IMAGE                    COMMAND                  CREATED          STATUS                    PORTS                                      NAMES
ab037be198e2   labdocker/webcache:1.0   "nginx -c /etc/nginx…"   43 seconds ago   Up 42 seconds (healthy)   0.0.0.0:80->8080/tcp, :::80->8080/tcp      webcache
e7b40d9b3683   labdocker/webservice:1.0 "python main.py"         13 minutes ago   Up 13 minutes (healthy)   0.0.0.0:5000->5000/tcp, :::5000->5000/tcp  webservice
00b582c4da1d   labdocker/redis:latest   "docker-entrypoint.s…"   14 minutes ago   Up 14 minutes             6379/tcp                                   cachedb
82b1e278d02f   labdocker/mariadb:10.5.12 "docker-entrypoint.s…"  14 minutes ago   Up 14 minutes             3306/tcp                                   maindb
ubuntu@ip-10-0-1-90:~/docker-workshop-092021/Workshop-2-1-DockerFile/python_restfulset$
```

```
.ssh — ubuntu@ip-10-0-1-90: ~/docker-workshop-092021 — ssh -i docker_lab.pem ubuntu@ec2-54-254-231-32.ap-southeast-1.compute.amazonaws.com — 187×31
...092021 — Terminal MAC Pro — -bash   ...cesetup — Terminal MAC Pro — -bash   ...eacher — Terminal MAC Pro — -bash   ...utheast-1.compute.amazonaws.com   ...utheast-1.compute.amazonaws.com
ubuntu@ip-10-0-1-90:~/docker-workshop-092021$ curl ifconfig.co
54.254.231.32
ubuntu@ip-10-0-1-90:~/docker-workshop-092021$ export Server_IP=54.254.231.32
ubuntu@ip-10-0-1-90:~/docker-workshop-092021$ export Server_Port=80
ubuntu@ip-10-0-1-90:~/docker-workshop-092021$ curl http://$Server_IP:$Server_Port/
<H1> Welcome Page from Container Python Lab </H1>Checkpoint Date/Time: Fri Sep  3 18:33:20 2021
ubuntu@ip-10-0-1-90:~/docker-workshop-092021$  curl http://$Server_IP:$Server_Port/init
###########   Database Create New Account Table Done  ###########
ubuntu@ip-10-0-1-90:~/docker-workshop-092021$
```

Not Secure | 54.254.231.32

Apps | NMac Ked - Mac... | Medium | Infra | NOVA | jerkins | vagrant | Mesos | Vue | docker | Taiwan | NGINX | Other Bookmarks | Reading List

Welcome Page from Container Python Lab

Checkpoint Date/Time: Fri Sep 3 18:33:56 2021

Workshop: DockerFile

- Init / Insert / Delete Data

```
########### Record was added ###########
ubuntu@ip-10-0-1-90:~/docker-workshop-092021$       curl -i -H "Content-Type: application/json" -X POST -d '{"uid": "8", "user":"Pansa Bunsong", "descripe":"Secuirty Guard"}' http://$Server_IP:$Server_Port/users/insertuser
HTTP/1.1 200 OK
Server: nginx/1.18.0
Date: Fri, 03 Sep 2021 18:34:55 GMT
Content-Type: application/json
Content-Length: 41
Connection: keep-alive

########### Record was added ###########
ubuntu@ip-10-0-1-90:~/docker-workshop-092021$       curl -i -H "Content-Type: application/json" -X POST -d '{"uid": "9", "user":"Wiphanee Wongsaisawan", "descripe":"Administrator"}' http://$Server_IP:$Server_Port/users/insertuser
HTTP/1.1 200 OK
Server: nginx/1.18.0
Date: Fri, 03 Sep 2021 18:34:57 GMT
Content-Type: application/json
Content-Length: 41
Connection: keep-alive

########### Record was added ###########
ubuntu@ip-10-0-1-90:~/docker-workshop-092021$ curl http://$Server_IP:$Server_Port/users/1
Praparn Luangphoonlap(Database Direct)
ubuntu@ip-10-0-1-90:~/docker-workshop-092021$ curl http://$Server_IP:$Server_Port/users/1
Praparn Luangphoonlap(Database Cache)
ubuntu@ip-10-0-1-90:~/docker-workshop-092021$ curl http://$Server_IP:$Server_Port/users/2
Somchai Sunsukwan(Database Direct)
ubuntu@ip-10-0-1-90:~/docker-workshop-092021$ curl http://$Server_IP:$Server_Port/users/2
Somchai Sunsukwan(Database Cache)
ubuntu@ip-10-0-1-90:~/docker-workshop-092021$ curl http://$Server_IP:$Server_Port/users/3
Sanyachan Panrudee(Database Direct)
ubuntu@ip-10-0-1-90:~/docker-workshop-092021$ curl http://$Server_IP:$Server_Port/users/3
Sanyachan Panrudee(Database Cache)
ubuntu@ip-10-0-1-90:~/docker-workshop-092021$ curl http://$Server_IP:$Server_Port/users/removeuser/1
########### Record was deleted (Both Database Cache) ###########
ubuntu@ip-10-0-1-90:~/docker-workshop-092021$       curl http://$Server_IP:$Server_Port/users/removeuser/2
########### Record was deleted (Both Database Cache) ###########
ubuntu@ip-10-0-1-90:~/docker-workshop-092021$       curl http://$Server_IP:$Server_Port/users/removeuser/3
########### Record was deleted (Both Database Cache) ###########
ubuntu@ip-10-0-1-90:~/docker-workshop-092021$       curl http://$Server_IP:$Server_Port/users/removeuser/4
########### Record was deleted ###########
ubuntu@ip-10-0-1-90:~/docker-workshop-092021$
```

Workshop: DockerFile

- Get Data (Direct/Cache) and Delete Data

```
ubuntu@ip-10-0-1-90:~/docker-workshop-092021$ curl http://$Server_IP:$Server_Port/users/1
Praparn Luangphoonlap(Database Direct)
ubuntu@ip-10-0-1-90:~/docker-workshop-092021$ curl http://$Server_IP:$Server_Port/users/1
Praparn Luangphoonlap(Database Cache)
ubuntu@ip-10-0-1-90:~/docker-workshop-092021$ curl http://$Server_IP:$Server_Port/users/2
Somchai Sunsukwan(Database Direct)
ubuntu@ip-10-0-1-90:~/docker-workshop-092021$ curl http://$Server_IP:$Server_Port/users/2
Somchai Sunsukwan(Database Cache)
ubuntu@ip-10-0-1-90:~/docker-workshop-092021$ curl http://$Server_IP:$Server_Port/users/3
Sanyachan Panrudee(Database Direct)
ubuntu@ip-10-0-1-90:~/docker-workshop-092021$ curl http://$Server_IP:$Server_Port/users/3
Sanyachan Panrudee(Database Cache)
ubuntu@ip-10-0-1-90:~/docker-workshop-092021$ curl http://$Server_IP:$Server_Port/users/removeuser/1
########### Record was deleted (Both Database Cache) ###########
ubuntu@ip-10-0-1-90:~/docker-workshop-092021$       curl http://$Server_IP:$Server_Port/users/removeuser/2
########### Record was deleted (Both Database Cache) ###########
ubuntu@ip-10-0-1-90:~/docker-workshop-092021$       curl http://$Server_IP:$Server_Port/users/removeuser/3
########### Record was deleted (Both Database Cache) ###########
ubuntu@ip-10-0-1-90:~/docker-workshop-092021$       curl http://$Server_IP:$Server_Port/users/removeuser/4
########### Record was deleted ###########
ubuntu@ip-10-0-1-90:~/docker-workshop-092021$
```

Workshop: DockerFile

- Case for POSTMAN

Workshop: DockerFile

- Case for POSTMAN

http://192.168.99.103 | http://192.168.99.103 | http://192.168.99.103 | No Environment

POST http://192.168.99.103/users/insertuser Params Send Save

Authorization Headers (1) Body Pre-request Script Tests Code

form-data x-www-form-urlencoded raw binary JSON (application/json)

```
1 {"uid": "1", "user":"Praparn Luangphoonlap", "descripe":"Slave"}
2
```

Body Cookies Headers (5) Tests Status: 200 OK Time: 25 ms

Pretty Raw Preview HTML

```
1 ########### Record was added ###########
2
```

Workshop: DockerFile

- Case for POSTMAN

Workshop: DockerFile

- Case for POSTMAN

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

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 -v
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

Compose

...

Compose

- Compose เป็นเครื่องมือที่ใช้ในการสร้าง system service ที่จะใช้ในการรันระบบงานทั้งระบบ ซึ่งตามปกติจะประกอบไปด้วยหลาย component อาทิเช่น
  - Web component service
  - Application component service
  - Database component service
  - Load balance component service
  - etc

- compose สามารถควบคุมการ start / stop / monitor การทำงานของ service ทั้งระบบเป็น single point

- เหมาะสำหรับการสร้าง development environment / test environment / automatic deploy to production (docker-machine / swam) (build one ship to everywhere)
- *No Support --ip now

Compose

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

docker file (Webserver)

docker file (Database)

docker file (…..)

Compose .yml file

```
build:
  context: /web
  dockerfile: xxxx

build:
  context: /db
  dockerfile: xxxx
```

"docker compose build"
"docker compose up"
"docker compose down"
"docker stack deploy"

Compose

docker docs | Search the docs | Home | Guides | Manuals | Reference | Samples | Contribute

/ Manuals / Docker Compose / Install Docker Compose / Overview

- Docker Engine
- Docker Build
- Docker Compose
  - Overview
  - Key features and use cases
  - Install Docker Compose
    - Overview
    - Install the Compose plugin
    - Install the Compose standalone
    - Uninstall Compose
  - Try Docker Compose
  - Compose V2
  - Environment variables
  - Environment file
  - Using service profiles
  - GPU support in Compose
  - Extend services in Compose
  - Networking in Compose
  - Using Compose in production
  - Control startup order

Installation scenarios

Scenario one: Install Docker Desktop

The easiest and recommended way to get Docker Compose is to install Docker Desktop. Docker Desktop includes Docker Compose along with Docker Engine and Docker CLI which are Compose prerequisites.

Docker Desktop is available on:

- Linux
- Mac
- Windows

If you have already installed Docker Desktop, you can check which version of Compose you have by selecting **About Docker Desktop** from the Docker menu

Scenario two: Install the Compose plugin

If you already have Docker Engine and Docker CLI installed, you can install the Compose plugin from the command line, by either:

- Using Docker's repository
- Downloading and installing manually

Note

This is only available on Linux

Scenario three: Install the Compose standalone

You can install the Compose standalone on Linux or on Windows Server.

Note

This install scenario is no longer supported.

Ref:https://docs.docker.com/compose/install/

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:
        - <xxxx>:<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)

Compose

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

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

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/

Compose

eficode / wait-for
forked from mrako/wait-for

Watch 20 | Star 580 | Fork 161

Code | Pull requests 12 | Projects 0 | Wiki | Security | Insights

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

15 commits | 1 branch | 0 releases | 5 contributors | MIT

Branch: master | New pull request | Create new file | Upload files | Find File | Clone or download

This branch is 8 commits ahead of mrako:master. #16 Compare

suonto Merge pull request #7 from szTheory/patch-1 ... 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 vishnubob/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 passing

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

Compose

vishnubob / wait-for-it

Watch 70 | Star 3,600 | Fork 910

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

36 commits | 1 branch | 0 releases | 8 contributors | MIT

Branch: master | New pull request | Create new file | Upload files | Find File | Clone or download

douglas-gibbons Merge branch 'fwoelffel-master' — Latest commit 54d1f0b 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/fwoelffel/wait-for-it int...	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
                            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

Compose

Workshop-2-1-DockerFile ▸ python_restfulset ▸ main.py

```python
from flask import Flask
from flask import Response
from flask import request
from redis import Redis
from datetime import datetime
import MySQLdb
import sys
import redis
import time
import hashlib
import os
import json
flagloop= 0
app = Flask(__name__)
startTime = datetime.now()
while flagloop == 0:
    try:
        db = MySQLdb.connect("maindb","root","password")
    except:
        time.sleep(2)
        continue
    else:
        flagloop= 1

CACHE_DB = redis.Redis(host=os.environ.get('REDIS_HOST', 'cachedb'), port=6379)
MAIN_DB = db.cursor()
@app.route('/')
def hello():
    return '<H1> Welcome Page from Container Python Lab </H1>Checkpoint Date/Time: ' + time.strftime("%c") +'\n'
@app.route('/init')
def init():
    MAIN_DB.execute("DROP DATABASE IF EXISTS ACCTABLE")
    MAIN_DB.execute("CREATE DATABASE ACCTABLE")
    MAIN_DB.execute("USE ACCTABLE")
    sql = """CREATE TABLE users (
        ID int,
        USER char(30),
        DESCRIPE char(250)
     )"""
    MAIN_DB.execute(sql)
    db.commit()
    return "###########   Database Create New Account Table Done  ###########\n"
```

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>

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

Compose

- Compose State:

Workshop: Compose

- Part1: composenodejs

Workshop: Compose

- dockerfile

```
FROM labdocker/alpine:3.13.6
LABEL 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 curl --no-cache && rm -rf /var/cache/apk/*
RUN mkdir /nodejs
COPY hello.js /nodejs/
CMD ["nginx","-c","/etc/nginx/nginx.conf"]
WORKDIR /nodejs
ENTRYPOINT ["node","hello.js"]
EXPOSE 3000
HEALTHCHECK --interval=15s --timeout=5s --start-period=10s --retries=3 \
    CMD curl -f http://localhost:3000/ || exit 1
```

Workshop: Compose

- docker-compose.yml

```yaml
version: "3.4"
services:
  web:
    build:
      context: ./web
      dockerfile: dockerfile          # optional for specific dockerfile
    container_name: nodejs            # optional for specific container name
    command: ["node","hello.js"]      # optional for override command on dockerfile
    healthcheck:
      test: ["CMD", "curl", "-f", "http://localhost:3000"]  # option for health check and send curl for check http://localhost:3000
      interval: 30s                                         # interval health check
      timeout: 10s                                          # timeout for check
      retries: 3                                            # maximum retries
      start_period: 30s                                     # start period <news on 3.4>
    ports:
      - 3000:3000
```

Workshop: Compose

- Part 2: composemultinodejs

Workshop: Compose

- Use case: multinodejs with nginx

Path: /dockerworkshop

docker-compose.yml

/nginx

dockerfile
(webinternal)
(webpublic)

nginx.conf
load balance:
web1
web2

/web1

dockerfile (webinternal)

/web2

dockerfile (webinternal)

Workshop: Compose

- nginx: dockerfile

```
FROM labdocker/alpine:3.13.6
LABEL 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 --no-cache && \
    rm /etc/nginx/conf.d/default.conf \
    rm -rf /var/cache/apk/*
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
HEALTHCHECK --interval=15s --timeout=5s --start-period=10s --retries=3 \
    CMD curl -f http://localhost:8080/ || exit 1
```

Workshop: Compose

- Web1 & Web 2: dockerfile

```
FROM labdocker/alpine:3.13.6
LABEL 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 curl --no-cache && rm -rf /var/cache/apk/*
RUN mkdir /nodejs
COPY hello.js /nodejs/
WORKDIR /nodejs
ENTRYPOINT ["node","hello.js"]
EXPOSE 3000
HEALTHCHECK --interval=15s --timeout=5s --start-period=10s --retries=3 \
    CMD curl -f http://localhost:3000/ || exit 1
```

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/');
```

Workshop: Compose

- docker-compose.yml

```yaml
version: '3.4'
services:
  nginx:
    build:
      context: ./nginx
      dockerfile: dockerfile          # optional for specific dockerfile
      cache_from:
        - alpine:latest
    container_name: nginx
    depends_on:
      - web1
      - web2
    healthcheck:
      test: ["CMD", "curl", "-f", "http://localhost:80"]  # option for health check and send curl for check http://localhost:3000
      interval: 30s                                        # interval health check
      timeout: 10s                                         # timeout for check
      retries: 3                                           # maximum retries
      start_period: 30s                                    # start period <news on 3.4>
    networks:
      webpublic:
        aliases:
        - nginx
      webinternal:
        aliases:
        - nginx
    ports:
      - "80:8080"
```

Workshop: Compose

- docker-compose.yml

```yaml
  web1:
    build:
      context: ./web1
      dockerfile: dockerfile          # optional for specific dockerfile
      cache_from:
        - labdocker/alpineweb:latest
    container_name: web1
    healthcheck:
      test: ["CMD", "curl", "-f", "http://localhost:3000"]  # option for health check and send curl for check http://localhost:3000
      interval: 30s                                         # interval health check
      timeout: 10s                                          # timeout for check
      retries: 3                                            # maximum retries
      start_period: 30s                                     # start period <news on 3.4>
    networks:
      webinternal:
        aliases:
        - web1

  web2:
    build:
      context: ./web2
      dockerfile: dockerfile          # optional for specific dockerfile
      cache_from:
        - labdocker/alpineweb:latest
    container_name: web2
    healthcheck:
      test: ["CMD", "curl", "-f", "http://localhost:3000"]  # option for health check and send curl for check http://localhost:3000
      interval: 30s                                         # interval health check
      timeout: 10s                                          # timeout for check
      retries: 3                                            # maximum retries
      start_period: 30s                                     # start period <news on 3.4>
    networks:
      webinternal:
        aliases:
        - web2
```

Workshop: Compose

- docker-compose.yml

```yaml
networks:
    webpublic:
      driver: bridge
      ipam:
        driver: default
        config:
        - subnet: 192.168.100.0/24
    webinternal:
      driver: bridge
      ipam:
        driver: default
        config:
        - subnet: 192.168.101.0/24
```

Docker Security

...

Docker Security

- เพื่อให้การรันงานบน docker container มีความปลอดภัยสูง docker มีการพิจารณาเรื่อง security ในการทำงานแบ่งออกได้ดังนี้

- 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)
  - Beaware 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 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 Security

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

Docker Security

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

README.md

Docker Bench for 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 v1.2.0.
# ------------------------------------------------------------------------------

Initializing Tue Nov  5 10:27:42 UTC 2019


[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]        * Using 19.03.4, verify is it up to date as deemed necessary
[INFO]        * 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
[INFO]        * docker:x:998:
[WARN] 1.2.3  - Ensure auditing is configured for the Docker daemon
[WARN] 1.2.4  - Ensure auditing is configured for Docker files and directories - /var/lib/docker
[WARN] 1.2.5  - Ensure auditing is configured for Docker files and directories - /etc/docker
[WARN] 1.2.6  - Ensure auditing is configured for Docker files and directories - docker.service
[WARN] 1.2.7  - Ensure auditing is configured for Docker files and directories - docker.socket
[WARN] 1.2.8  - Ensure auditing is configured for Docker files and directories - /etc/default/docker
[INFO] 1.2.9  - Ensure auditing is configured for Docker files and directories - /etc/sysconfig/docker
[INFO]        * File not found
[INFO] 1.2.10  - Ensure auditing is configured for Docker files and directories - /etc/docker/daemon.json
[INFO]         * File not found
[WARN] 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]         * File not found


[INFO] 2 - Docker daemon configuration
[WARN] 2.1  - Ensure network traffic is restricted between containers on the default bridge
[PASS] 2.2  - Ensure the logging level is set to 'info'
[PASS] 2.3  - Ensure Docker is allowed to make changes to iptables
```

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 Benchmark v1.2.0.

```
ubuntu@ip-10-0-1-234:~/docker-bench-security$ docker run -it --net host --pid host --cap-add audit_control \
>     -e DOCKER_CONTENT_TRUST=$DOCKER_CONTENT_TRUST \
>     -v /var/lib:/var/lib:ro \
>     -v /var/run/docker.sock:/var/run/docker.sock:ro \
>     -v /usr/lib/systemd:/usr/lib/systemd:ro \
>     -v /etc:/etc:ro --label docker_bench_security \
>     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 v1.2.0.
# ------------------------------------------------------------------------------

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 Docker is up to date
[INFO]        * Using 19.03.6, verify is it up to date as deemed necessary
[INFO]        * 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
[INFO]        * docker:x:1001:ubuntu,1001
[WARN] 1.2.3  - Ensure auditing is configured for the Docker daemon
[WARN] 1.2.4  - Ensure auditing is configured for Docker files and directories - /var/lib/docker
[WARN] 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]        * File not found
[INFO] 1.2.7  - Ensure auditing is configured for Docker files and directories - docker.socket
[INFO]        * File not found
[WARN] 1.2.8  - Ensure auditing is configured for Docker files and directories - /etc/default/docker
[INFO] 1.2.9  - Ensure auditing is configured for Docker files and directories - /etc/sysconfig/docker
[INFO]        * File not found
[WARN] 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]         * File not found
[INFO] 1.2.12  - Ensure auditing is configured for Docker files and directories - /usr/sbin/runc
[INFO]         * File not found
```

Ref: https://github.com/docker/docker-bench-security

Docker Security

- Synk for docker image scanning (built-in)
  - Scan in each layer of docker image and compare with know vulnerability

```
ubuntu@ip-10-0-1-185:~$ docker scan labdockerthailand/alpineweb:latest

Testing labdockerthailand/alpineweb:latest...

✗ Low severity vulnerability found in nodejs/nodejs
  Description: CVE-2021-22960
  Info: https://snyk.io/vuln/SNYK-ALPINE313-NODEJS-1731455
  Introduced through: nodejs/nodejs@14.17.6-r0, nodejs/npm@14.17.6-r0
  From: nodejs/nodejs@14.17.6-r0
  From: nodejs/npm@14.17.6-r0 > nodejs/nodejs@14.17.6-r0
  From: nodejs/npm@14.17.6-r0
  Image layer: '/bin/sh -c apk update &&     apk add nodejs nodejs-npm curl --no-cache &&     rm -rf /var/cache/apk/*'
  Fixed in: 14.18.1-r0

✗ Low severity vulnerability found in nodejs/nodejs
  Description: CVE-2021-22959
  Info: https://snyk.io/vuln/SNYK-ALPINE313-NODEJS-1731456
  Introduced through: nodejs/nodejs@14.17.6-r0, nodejs/npm@14.17.6-r0
  From: nodejs/nodejs@14.17.6-r0
  From: nodejs/npm@14.17.6-r0 > nodejs/nodejs@14.17.6-r0
  From: nodejs/npm@14.17.6-r0
  Image layer: '/bin/sh -c apk update &&     apk add nodejs nodejs-npm curl --no-cache &&     rm -rf /var/cache/apk/*'
  Fixed in: 14.18.1-r0

✗ Medium severity vulnerability found in curl/libcurl
  Description: Insufficient Verification of Data Authenticity
  Info: https://snyk.io/vuln/SNYK-ALPINE313-CURL-1585257
  Introduced through: curl/libcurl@7.78.0-r0, curl/curl@7.78.0-r0
  From: curl/libcurl@7.78.0-r0
  From: curl/curl@7.78.0-r0 > curl/libcurl@7.78.0-r0
  From: curl/curl@7.78.0-r0
  Image layer: '/bin/sh -c apk update &&     apk add nodejs nodejs-npm curl --no-cache &&     rm -rf /var/cache/apk/*'
  Fixed in: 7.79.0-r0

✗ High severity vulnerability found in curl/libcurl
  Description: Cleartext Transmission of Sensitive Information
  Info: https://snyk.io/vuln/SNYK-ALPINE313-CURL-1585248
  Introduced through: curl/libcurl@7.78.0-r0, curl/curl@7.78.0-r0
  From: curl/libcurl@7.78.0-r0
  From: curl/curl@7.78.0-r0 > curl/libcurl@7.78.0-r0
  From: curl/curl@7.78.0-r0
  Image layer: '/bin/sh -c apk update &&     apk add nodejs nodejs-npm curl --no-cache &&     rm -rf /var/cache/apk/*'
  Fixed in: 7.79.0-r0

✗ Critical severity vulnerability found in curl/libcurl
  Description: Double Free
  Info: https://snyk.io/vuln/SNYK-ALPINE313-CURL-1585246
  Introduced through: curl/libcurl@7.78.0-r0, curl/curl@7.78.0-r0
  From: curl/libcurl@7.78.0-r0
  From: curl/curl@7.78.0-r0 > curl/libcurl@7.78.0-r0
  From: curl/curl@7.78.0-r0
  Image layer: '/bin/sh -c apk update &&     apk add nodejs nodejs-npm curl --no-cache &&     rm -rf /var/cache/apk/*'
  Fixed in: 7.79.0-r0


Package manager:   apk
Project name:      docker-image|labdockerthailand/alpineweb
Docker image:      labdockerthailand/alpineweb:latest
Platform:          linux/amd64
Base image:        alpine:3.13.6

Tested 24 dependencies for known vulnerabilities, found 5 vulnerabilities.

According to our scan, you are currently using the most secure version of the selected base image

For more free scans that keep your images secure, sign up to Snyk at https://dockr.ly/3ePqVcp

ubuntu@ip-10-0-1-185:~$
```

Docker Security

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

```
[ubuntu@ip-10-0-1-104:~$ docker container exec -it sectestdefault sh
[/ # touch /tmp/testcreatefile
[/ # ls -hl /tmp
 total 0
 -rw-r--r--    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
 dr-xr-xr-x    9 root     root           0 Mar  3 07:33 1
 dr-xr-xr-x    9 root     root           0 Mar  3 07:34 14
 dr-xr-xr-x    9 root     root           0 Mar  3 07:33 6
 drwxrwxrwt    2 root     root          40 Mar  3 07:33 acpi
 -r--r--r--    1 root     root           0 Mar  3 07:34 buddyinfo
 dr-xr-xr-x    4 root     root           0 Mar  3 07:33 bus
 -r--r--r--    1 root     root           0 Mar  3 07:34 cgroups
 -r--r--r--    1 root     root           0 Mar  3 07:34 cmdline
 -r--r--r--    1 root     root           0 Mar  3 07:34 consoles
 -r--r--r--    1 root     root           0 Mar  3 07:34 cpuinfo
 -r--r--r--    1 root     root           0 Mar  3 07:34 crypto
 -r--r--r--    1 root     root           0 Mar  3 07:34 devices
 -r--r--r--    1 root     root           0 Mar  3 07:34 diskstats
 -r--r--r--    1 root     root           0 Mar  3 07:34 dma
 dr-xr-xr-x    2 root     root           0 Mar  3 07:34 driver
 -r--r--r--    1 root     root           0 Mar  3 07:34 execdomains
 -r--r--r--    1 root     root           0 Mar  3 07:34 fb
 -r--r--r--    1 root     root           0 Mar  3 07:34 filesystems
 dr-xr-xr-x    6 root     root           0 Mar  3 07:33 fs
 -r--r--r--    1 root     root           0 Mar  3 07:34 interrupts
 -r--r--r--    1 root     root           0 Mar  3 07:34 iomem
 -r--r--r--    1 root     root           0 Mar  3 07:34 ioports
 dr-xr-xr-x   23 root     root           0 Mar  3 07:33 irq
 -r--r--r--    1 root     root           0 Mar  3 07:34 kallsyms
 crw-rw-rw-    1 root     root        1,   3 Mar  3 07:33 kcore
 -r--r--r--    1 root     root           0 Mar  3 07:34 key-users
 crw-rw-rw-    1 root     root        1,   3 Mar  3 07:33 keys
[/ # ls -hl /proc | grep test
[/ # cp /tmp/testcreatefile /proc/testprohibitfile
 cp: can't create '/proc/testprohibitfile': No such file or directory
[/ # ls -hl /proc | grep test
[/ # exit
[ubuntu@ip-10-0-1-104:~$ docker container stop sectestdefault
 sectestdefault
```

moby / moby

Watch 3,287 | Star 52,368 | Fork 15,155

Code | Issues 3,434 | Pull requests 183 | Projects 2 | Wiki | Insights

Branch: master | moby / profiles / apparmor / template.go | Find file | Copy path

cyphar apparmor: allow receiving of signals from 'docker kill' 4822fb1 on Sep 12, 2018

5 contributors

51 lines (42 sloc) | 1.53 KB | Raw | Blame | History

```
// +build linux

package apparmor // import "github.com/docker/docker/profiles/apparmor"

// baseTemplate defines the default apparmor profile for containers.
const baseTemplate = `
{{range $value := .Imports}}
{{$value}}
{{end}}

profile {{.Name}} flags=(attach_disconnected,mediate_deleted) {
{{range $value := .InnerImports}}
  {{$value}}
{{end}}

  network,
  capability,
  file,
  umount,
{{if ge .Version 208096}}
{{/* Allow 'docker kill' to actually send signals to container processes. */}}
  signal (receive) peer={{.DaemonProfile}},
{{/* Allow container processes to send signals amongst themselves. */}}
  signal (send,receive) peer={{.Name}},
{{end}}

  deny @{PROC}/* w,   # deny write for all files directly in /proc (not in a subdir)
  # deny write to files not in /proc/<number>/** or /proc/sys/**
  deny @{PROC}/{[^1-9],[^1-9][^0-9],[^1-9s][^0-9y][^0-9s],[^1-9][^0-9][^0-9][^0-9]*}/** w,
  deny @{PROC}/sys/[^k]** w,  # deny /proc/sys except /proc/sys/k* (effectively /proc/sys/kernel)
  deny @{PROC}/sys/kernel/{?,??,[^s][^h][^m]**} w,  # deny everything except shm* in /proc/sys/kernel/
  deny @{PROC}/sysrq-trigger rwklx,
  deny @{PROC}/kcore rwklx,

  deny mount,

  deny /sys/[^f]*/** wklx,
  deny /sys/f[^s]*/** wklx,
  deny /sys/fs/[^c]*/** wklx,
  deny /sys/fs/c[^g]*/** wklx,
  deny /sys/fs/cg[^r]*/** wklx,
  deny /sys/firmware/** rwklx,
  deny /sys/kernel/security/** rwklx,

{{if ge .Version 208095}}
  # suppress ptrace denials when using 'docker ps' or using 'ps' inside a container
  ptrace (trace,read) peer={{.Name}},
{{end}}
}
`
```

Ref: https://github.com/moby/moby/tree/master/profiles/apparmor

Docker Security

- Image Content Trust

Docker Security

- Recommendation for enhance docker security
  - ระวังการจัดการ user ที่มีหน้าที่ทำงานกับ docker daemon และ privilege คำสั่งบน docker command
  - ใช้ non-root user ในการจัดการ docker daemon operation (docker run etc)
  - จัดการ Image Content Trust (Sign Image) ในการสั่ง pull/push image
  - เปลี่ยนการจัดการ docker daemon socket เป็น HTTP socket (with TLS verify and authentication)
  - จำกัดสิทธิและขอบเขตการทำงานของ user บน container (Secure compute mode (seccomp)

Workshop: Docker Security

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

```
ubuntu@ip-10-0-1-234:~/docker-bench-security$ docker run -it --net host --pid host --cap-add audit_control \
>     -e DOCKER_CONTENT_TRUST=$DOCKER_CONTENT_TRUST \
>     -v /var/lib:/var/lib:ro \
>     -v /var/run/docker.sock:/var/run/docker.sock:ro \
>     -v /usr/lib/systemd:/usr/lib/systemd:ro \
>     -v /etc:/etc:ro --label docker_bench_security \
>     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 v1.2.0.
# ------------------------------------------------------------------------------

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 Docker is up to date
[INFO]        * Using 19.03.6, verify is it up to date as deemed necessary
[INFO]        * 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
[INFO]        * docker:x:1001:ubuntu,1001
[WARN] 1.2.3  - Ensure auditing is configured for the Docker daemon
[WARN] 1.2.4  - Ensure auditing is configured for Docker files and directories - /var/lib/docker
[WARN] 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]        * File not found
[INFO] 1.2.7  - Ensure auditing is configured for Docker files and directories - docker.socket
[INFO]        * File not found
[WARN] 1.2.8  - Ensure auditing is configured for Docker files and directories - /etc/default/docker
[INFO] 1.2.9  - Ensure auditing is configured for Docker files and directories - /etc/sysconfig/docker
[INFO]        * File not found
[WARN] 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]         * File not found
[INFO] 1.2.12  - Ensure auditing is configured for Docker files and directories - /usr/sbin/runc
[INFO]         * File not found
```

Workshop: Docker Security

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

```
ubuntu@ip-10-0-1-234:~/docker-bench-security$ docker run -it --net host --pid host --cap-add audit_control \
>     -e DOCKER_CONTENT_TRUST=$DOCKER_CONTENT_TRUST \
>     -v /var/lib:/var/lib:ro \
>     -v /var/run/docker.sock:/var/run/docker.sock:ro \
>     -v /usr/lib/systemd:/usr/lib/systemd:ro \
>     -v /etc:/etc:ro --label docker_bench_security \
>     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 v1.2.0.
# ------------------------------------------------------------------------------

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 Docker is up to date
[INFO]        * Using 19.03.6, verify is it up to date as deemed necessary
[INFO]        * 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
[INFO]        * docker:x:1001:ubuntu,1001
[WARN] 1.2.3  - Ensure auditing is configured for the Docker daemon
[WARN] 1.2.4  - Ensure auditing is configured for Docker files and directories - /var/lib/docker
[WARN] 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]        * File not found
[INFO] 1.2.7  - Ensure auditing is configured for Docker files and directories - docker.socket
[INFO]        * File not found
[WARN] 1.2.8  - Ensure auditing is configured for Docker files and directories - /etc/default/docker
[INFO] 1.2.9  - Ensure auditing is configured for Docker files and directories - /etc/sysconfig/docker
[INFO]        * File not found
[WARN] 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]         * File not found
[INFO] 1.2.12  - Ensure auditing is configured for Docker files and directories - /usr/sbin/runc
[INFO]         * File not found
```

1.1 Linux Hosts Specific Configuration

This section contains recommendations that securing Linux Hosts running Docker Containers.

1.1.1 Ensure a separate partition for containers has been created (Automated)

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` should be mounted under either the `/` or `/var` partitions dependent on how the Linux operating system in use is configured.

Rationale:

Docker depends on `/var/lib/docker` as the default directory where all Docker related files, including the images, are stored. This directory could fill up quickly causing both Docker and the host to become unusable. For this reason, you should create a separate partition (logical volume) for storing Docker files.

Impact:

None.

Audit:

At the Docker host execute one of the below commands:

```
grep '/var/lib/docker\s' /proc/mounts
```

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

```
mountpoint -- "$(docker info -f '{{ .DockerRootDir }}')"
```

This should return whether the configured root directory is a mount point.

Workshop: Docker Security

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

```
ubuntu@ip-10-0-1-234:~/docker-bench-security$ docker run -it --net host --pid host --cap-add audit_control \
>     -e DOCKER_CONTENT_TRUST=$DOCKER_CONTENT_TRUST \
>     -v /var/lib:/var/lib:ro \
>     -v /var/run/docker.sock:/var/run/docker.sock:ro \
>     -v /usr/lib/systemd:/usr/lib/systemd:ro \
>     -v /etc:/etc:ro --label docker_bench_security \
>     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 v1.2.0.
# ------------------------------------------------------------------------------

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 Docker is up to date
[INFO]        * Using 19.03.6, verify is it up to date as deemed necessary
[INFO]        * 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
[INFO]        * docker:x:1001:ubuntu,1001
[WARN] 1.2.3  - Ensure auditing is configured for the Docker daemon
[WARN] 1.2.4  - Ensure auditing is configured for Docker files and directories - /var/lib/docker
[WARN] 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]        * File not found
[INFO] 1.2.7  - Ensure auditing is configured for Docker files and directories - docker.socket
[INFO]        * File not found
[WARN] 1.2.8  - Ensure auditing is configured for Docker files and directories - /etc/default/docker
[INFO] 1.2.9  - Ensure auditing is configured for Docker files and directories - /etc/sysconfig/docker
[INFO]        * File not found
[WARN] 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]         * File not found
[INFO] 1.2.12  - Ensure auditing is configured for Docker files and directories - /usr/sbin/runc
[INFO]         * 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` file:

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

Then, restart the audit daemon. For example,

```
service auditd restart
```

Workshop: Docker Security

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

```
ubuntu@ip-10-0-1-234:~/docker-bench-security$ docker run -it --net host --pid host --cap-add audit_control \
>     -e DOCKER_CONTENT_TRUST=$DOCKER_CONTENT_TRUST \
>     -v /var/lib:/var/lib:ro \
>     -v /var/run/docker.sock:/var/run/docker.sock:ro \
>     -v /usr/lib/systemd:/usr/lib/systemd:ro \
>     -v /etc:/etc:ro --label docker_bench_security \
>     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 v1.2.0.
# ------------------------------------------------------------------------------

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 Docker is up to date
[INFO]        * Using 19.03.6, verify is it up to date as deemed necessary
[INFO]        * 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
[INFO]        * docker:x:1001:ubuntu,1001
[WARN] 1.2.3  - Ensure auditing is configured for the Docker daemon
[WARN] 1.2.4  - Ensure auditing is configured for Docker files and directories - /var/lib/docker
[WARN] 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]        * File not found
[INFO] 1.2.7  - Ensure auditing is configured for Docker files and directories - docker.socket
[INFO]        * File not found
[WARN] 1.2.8  - Ensure auditing is configured for Docker files and directories - /etc/default/docker
[INFO] 1.2.9  - Ensure auditing is configured for Docker files and directories - /etc/sysconfig/docker
[INFO]        * File not found
[WARN] 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]         * File not found
[INFO] 1.2.12  - Ensure auditing is configured for Docker files and directories - /usr/sbin/runc
[INFO]         * 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.
## 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:~$
```

Workshop: Docker Security

- <u>Docker Bench for Security</u>
- Purpose: ทำการตรวจสอบช่องโหว่บน 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:/var/lib:ro     -v /var/run/docker.sock:/var/run/docker.sock:ro     -v /usr/lib/systemd:/usr/lib/systemd:ro     -v /e
tc:/etc:ro --label docker_bench_security     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 v1.2.0.
# ------------------------------------------------------------------------------

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]        * Using 19.03.6, verify is it up to date as deemed necessary
[INFO]        * 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
[INFO]        * docker:x: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
[PASS] 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]        * File not found
[INFO] 1.2.7  - Ensure auditing is configured for Docker files and directories - docker.socket
[INFO]        * File not found
[PASS] 1.2.8  - Ensure auditing is configured for Docker files and directories - /etc/default/docker
[INFO] 1.2.9  - Ensure auditing is configured for Docker files and directories - /etc/sysconfig/docker
[INFO]        * File not found
[PASS] 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]         * File not found
[INFO] 1.2.12  - Ensure auditing is configured for Docker files and directories - /usr/sbin/runc
[INFO]         * File not found
```

Workshop: Docker Security

- <u>Apparmor</u>
- Purpose: ระงับการทำงานบางอย่างบน container ที่ไม่จำเป็นเพื่อลดความเสี่ยงในการใช้งานเกินขอบเขตของ application

```
ubuntu@ip-10-0-1-104:~$ docker container run -dt --name sectestrestrict --security-opt "apparmor=restrict-apparmor" labdocker/alpine:latest sh
6db2a1be7b9d1b3f15409e49bc1a0e1a9373cc9b8740a017ff8fb163283b8d1b
ubuntu@ip-10-0-1-104:~$ docker container exec -it sectestdefault 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 /home/testcreatefile2
touch: /home/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:~$
```

Workshop: Docker Security

- <u>Content Trust</u>
- Purpose: ทำการ 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: sha256:65242e8220a341cec40628caaea77eb4acd2fc252329aa853526fde15a4a1d85 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):
```

Registry

...

Registry

- Registry เป็น container แบบหนึ่งที่ใช้สำหรับจัดเก็บ image ที่สร้างขึ้นไว้บน 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 ได้เช่น
- Etc: s3 (aws), azure, gcs (google cloud) etc
- มี option gc (garbage collect) ด้วย option
- “bin/registry garbage-collect [--dry-run] config.yml”

Workshop: Registry

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

```
docker image pull registry:2.8.0 (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:2.8.0
```

- ดำเนินการ tag image และทดสอบ push image docker เข้า registry

```
docker image tag labdocker/alpine:3.13.6 \
localhost:5000/alpine:3.13.6
```

```
docker push localhost:5000/alpine:3.13.6
```

- *ต้องบันทึก digest image ไว้เพื่อใช้อ้างอิงภายหลัง

Workshop: Registry

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

```
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>
```

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:3.13.6
```

- ทำการดึง image ใหม่มาจาก registry

```
docker image pull localhost:5000/alpine:3.13.6
```

- ทำการลบ container registry เพื่อยกเลิกการใช้งาน

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

Registry on difference machine

- <u>Part2: Registry on multiple docker-machine</u>
- กรณีที่ต้องใช้งาน registry จาก docker server มากกว่า 1 เครื่อง เราสามารถดึงต้องสร้าง TLS certificate (next generate of SSL) เพื่อ trust registry ระหว่างกัน

Registry on difference machine

- การสร้าง certificate จาก CA / Self-Signed

Registry on difference machine

- เพิ่ม domain “labdocker.com” ลงใน /etc/hosts (registry, all machine)

192.168.99.100 - PuTTY

```
# 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
I /etc/hosts 12/12 100%
```

192.168.99.100 - PuTTY

```
docker@labdocker:~$ sudo vi /etc/hosts
docker@labdocker:~$ ping 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
^C
--- 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:~$
```

Registry on difference machine

- สำหรับกรณีเครื่อง ubuntu ทั่วไปที่ใช้งาน 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; 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$
```

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=htpasswd" \
        -e "REGISTRY_AUTH_HTPASSWD_REALM=Registry Realm" \
        -e REGISTRY_AUTH_HTPASSWD_PATH=/auth/htpasswd \
        -e REGISTRY_STORAGE_DELETE_ENABLED=true \
        -e REGISTRY_HTTP_TLS_CERTIFICATE=/certs/labdocker.com.crt \
        -e REGISTRY_HTTP_TLS_KEY=/certs/labdocker.com.key \
        registry:2.8.0
```

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

```
docker image tag labdocker/alpine:3.13.6 labdocker.com:5000/alpine:1.0
docker login -u <username> -p <password>
docker image push labdocker.com:5000/alpine:1.0
```

Registry on difference machine

- ในการตรวจสอบ registry image file สำหรับ registry ที่มีการ enable TLS เรียบบร้อยแล้วต้องเพิ่มพารามิเตอร์ “--cacert “ เพื่อระบุ certificate สำหรับการใช้งานและเปลี่ยน protocol เป็น https

- การ List รายการ image ที่จัดเก็บไว้บน

```
curl –u <username:password> --cacert /Share_DockerToolbox/labdocker.com.crt  -X
GET https://labdocker.com:5000/v2/_catalog
```

- ตรวจสอบ revision image tag ที่จัดเก็บไว้ในแต่ละ image

```
curl –u <username:password> --cacert /home/docker/labdockercert/labdocker.com.crt
-X GET https://labdocker.com:5000/v2/alpine/tags/list
```

- สามารถทำการลบ image ที่จัดเก็บไว้โดยอ้างอิงจาก digest

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

Registry on difference machine

- ดึง image ผ่าน labdocker 2

```
rtt min/avg/max/mdev = 0.122/0.158/0.194/0.036 ms
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: sha256:02892826401a9d18f0ea01f8a2f35d328ef039db4e1edcc45c630314a0457d5b
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 Trust Registry (DTR)

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

Artifact Registry

- Sonatype

Sonatype | Documentation My Sonatype Community Guides Learn Support Who is Sonatype?

Search the docs...

Switch to another product

NEXUS REPOSITORY MANAGER 3

- Download
- Release Notes
- System Requirements
- Upgrade Compatibility - Repository Manager 2 to 3
- Repository Manager 2 to 3 Feature Equivalency
- Repository Manager Feature Matrix
- Repository Manager Concepts
- Supported Formats
- Installation
- Upgrading
- User Interface
- Configuration
- Backup and Restore
- Cleanup Policies
- High Availability
- Quick Start Guide - Proxying Maven and NPM
- Staging
- Tagging

Nexus Repository Manager 3 > Private Registry for Docker

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 Manager 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.

Artifact Registry

- Harbor CNCF Registry (Incubator)

Swarm

...

Swarm: Conceptual

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

Swarm: Conceptual

- Traditional Docker Deployment

Swarm: Conceptual

- Swarm Deployment

Swarm Mode Architecture

- Swarm Mode (build-in swarm) เป็นฟังก์ชั่นที่มาพร้อม docker-engine ตั้งแต่แรก และสามารถบริหารจัดการ swarm ได้ด้วยตัวเองรวมถึง build-in TLS certificate เพื่อใช้ทำงานระหว่าง node ทันที (Security On the Box)

- Swarm Role
  - Manager Node
  - Worker Node

Swarm Mode Architecture

Swarm Mode Architecture

Swarm Mode Architecture

docker

Swarm Mode Architecture

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

All

Engineering

Curated

Docker Weekly

The core team also wanted to give a special thanks to one of our external maintainers and Docker Captain, Chanwit Kaewkasi, 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 nodes and just under 100K containers. This was achieved through our global community, who donated access to their machines in all shapes and sizes from bare metal, 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.

K8S vs Docker what is the difference ?

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

K8S vs Docker what is the difference ?

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 ubunt

```
ubuntu@ip-10-0-1-104:~$ docker swarm init --advertise-addr 10.0.1.104:2377 --task-history-limit 2
Swarm initialized: current node (3tnvltyyohbekgygvsb1jjovy) is now a manager.

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

    docker swarm join --token SWMTKN-1-1qnpobqw0hilpc5i90sz803tlhyi1h4k84uok10y814bvhyqf2-0ag405byoqpiqgqhoy6r866tv 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-1-1qnpobqw0hilpc5i90sz803tlhyi1h4k84uok10y814bvhyqf2-3u9rqxfwhbhw52qwqeeiewybl 10.0.1.104:2377

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

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 ภายนอกเพื่อทำ trust node ภายใน swarm
  - --force-new-cluster: บังคับการสร้าง swarm cluster ใหม่
  - --advertise-addr value: ระบุ ip address ที่ใช้คุยระหว่าง swarm node (default: 0.0.0.0:2377)
  - --task-history-limit <int>: กำหนดการจัดเก็บ history task ย้อนหลังบน swarm (default: 10)
  - --availability: <“active”/”pause”/”drain”>
  - Etc.

Swarm Init/Join

- ทำการ 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 SWMTKN-1-1qnpobqw0hilpc5i90sz803tlhyi1h4k84uok10y814bvhyqf2-0ag405byoqpiqgqhoy6r866tv 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 SWMTKN-1-1qnpobqw0hilpc5i90sz803tlhyi1h4k84uok10y814bvhyqf2-0ag405byoqpiqgqhoy6r866tv 10.0.1.104:2377
This node joined a swarm as a worker.
ubuntu@ip-10-0-1-62:~$
```

Swarm Init/Join

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

Swarm Init/Join

- ตรวจสอบ Token ที่ใช้ในการ 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@1
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-1-1qnpobqw0hilpc5i90sz803tlhyi1h4k84uok10y814bvhyqf2-0ag405byoqpiqgqhoy6r866tv 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-1-1qnpobqw0hilpc5i90sz803tlhyi1h4k84uok10y814bvhyqf2-3u9rqxfwhbhw52qwqeeiewybl 10.0.1.104:2377

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

- Option:
  - -q, --q : แสดงเฉพาะ token เท่านั้น
  - --rotate: ทำการเปลี่ยน token ใหม่

Swarm Init/Join

- ตรวจสอบ node ที่ทำงานภายใต้ swarm cluster ด้วยคำสั่ง docker node ls

```
docker node ls
```

```
...rkshop_112018 — Terminal MAC Pro — -bash   ...4: ~ — ssh -i k8s_lab ubuntu@13.229.126.99   ...3: ~ — 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                            18.06.1-ce
3tnvltyyohbekgygvsb1jjovy *   ip-10-0-1-104     Ready     Active          Leader            18.06.1-ce
ua0ajfjd4d1mksqzwr80oy2ru     ip-10-0-1-163     Ready     Active                            18.06.1-ce
ubuntu@ip-10-0-1-104:~$
```

Swarm Init/Join

- Option:
  - promote <ID>: ทำการ promote node จาก worker กลายเป็น manager
  - demote <ID>: ทำการ demote node จาก manager กลายเป็น worker ปกติ
  - Inspect <ID>: ตรวจสอบค่าคอนฟิกกูเรชั่นของ node
  - ls: แสดง node ทั้งหมดภายใต้ docker swarm cluster
  - ps <ID>: แสดงงานทั้งหมดที่รันอยู่ภายใต้ node
  - rm <ID>: ลบ node ออกจาก swarm cluster
  - update <ID>: ทำการ update node information
    - --availability <status>: อัพเดตสถานะของ 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)

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
```

Workshop: Swarm Mode

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

```
docker swarm join --token
SWMTKN-1-1qnpobqw0hilpc5i90sz803tlhyi1h4k84uok10y81
4bvhyqf2-0ag405byoqpiqgqhoy6r866tv 10.0.1.104:2377
```

- ตรวจสอบสถานะของ swarm หลังจาก join node เข้าไปเรียบร้อยแล้ว

```
docker node ls
```

```
...rkshop_112018 — Terminal MAC Pro — -bash   ...4: ~ — ssh -i k8s_lab ubuntu@13.229.126.99   ...3: ~ — 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                            18.06.1-ce
3tnvltyyohbekgygvsb1jjovy *   ip-10-0-1-104     Ready     Active          Leader            18.06.1-ce
ua0ajfjd4d1mksqzwr80oy2ru     ip-10-0-1-163     Ready     Active                            18.06.1-ce
ubuntu@ip-10-0-1-104:~$
```

Swarm Service

- สำหรับการสั่งรัน container บน swarm ได้มีการเปลี่ยนแปลงรูปแบบใหม่เช่นกัน โดยกำหนดการรัน container ถูกนิยามใหม่ในรูปแบบของคำสั่ง "docker service" เพื่อนิยามแทนการรันกลุ่มของ 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
4i4gu0o15f819jgohbgn994f7
ubuntu@ip-10-0-1-104:~$
```

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  Credential spec for managed service account (Windows only)
  -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/

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 string           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-TTY
      --update-delay duration              Delay between updates (ns|us|ms|s|m|h) (default 0s)
      --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>])
      --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/

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
4i4gu0o15f819jgohbgn994f7
ubuntu@ip-10-0-1-104:~$ docker service ls
ID                  NAME                MODE                REPLICAS            IMAGE                          PORTS
4i4gu0o15f81        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
h6u6lt7o564f        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
h6u6lt7o564f        nodejs.1            labdocker/alpineweb:latest     ip-10-0-1-104       Running             Running 2 minutes ago
r5j8wpaew0tm        nodejs.2            labdocker/alpineweb:latest     ip-10-0-1-104       Running             Running 26 seconds ago
w66ihyd8kbs2        nodejs.3            labdocker/alpineweb:latest     ip-10-0-1-163       Running             Running 16 seconds ago
gjgkorz84g7g        nodejs.4            labdocker/alpineweb:latest     ip-10-0-1-62        Running             Running 16 seconds ago
jubxx7bup73b        nodejs.5            labdocker/alpineweb:latest     ip-10-0-1-62        Running             Running 16 seconds ago
ubuntu@ip-10-0-1-104:~$
```

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

```
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": "4i4gu0o15f819jgohbgn994f7",
        "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": {},
```

Swarm Service

```
        },
        "UpdateStatus": {
            "State": "updating",
            "StartedAt": "2018-10-21T16:29:50.119815453Z",
            "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:~$
```

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
lchljj6puie2        nodejs.1            labdocker/alpineweb:latest     ip-10-0-1-104       Running             Running about a minute ago
h6u6lt7o564f         \_ nodejs.1        labdocker/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
lchljj6puie2        nodejs.1            labdocker/alpineweb:latest     ip-10-0-1-104       Running             Running 2 minutes ago
h6u6lt7o564f         \_ nodejs.1        labdocker/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:~$
```

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/

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 list                     Remove a mount by its target path
      --network-add network               Add a network
      --network-rm list                   Remove a network
      --no-healthcheck                    Disable any 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/

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 the restart policy (ns|us|ms|s|m|h)
    --rollback                         Rollback to previous specification
    --rollback-delay duration          Delay between task rollbacks (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-parallelism 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               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/

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_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 -dt --name nodejs \
> labdocker/alpineweb:latest node hello.js

sm7d6yzuedh7kpv4o3zv8tyx1
ubuntu@ip-10-0-1-104:~$
ubuntu@ip-10-0-1-104:~$ docker service ls
ID                  NAME                MODE                REPLICAS            IMAGE                          PORTS
sm7d6yzuedh7        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
pq53791w4q8a        nodejs.1            labdocker/alpineweb:latest     ip-10-0-1-104       Running             Running 7 seconds ago
ubuntu@ip-10-0-1-104:~$
```

Orchestrator Assignment

- Swarm สามารถควบคุมการจ่ายงานให้จาก manager ไปยัง workerได้หลากหลายเทคนิค 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)

Orchestrator Assignment

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

```
.ssh — ubuntu@ip-10-0-1-104: ~ — ssh -i k8s_lab ubuntu@13.229.126.99 — 165×16
...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 node ls
ID                            HOSTNAME          STATUS    AVAILABILITY    MANAGER STATUS    ENGINE VERSION
4g6pqogaae5co8zxe8w6pxfd6     ip-10-0-1-62      Ready     Active                            18.06.1-ce
3tnvltyyohbekgygvsb1jjovy *   ip-10-0-1-104     Ready     Active          Leader            18.06.1-ce
ua0ajfjd4d1mksqzwr80oy2ru     ip-10-0-1-163     Ready     Active                            18.06.1-ce
ubuntu@ip-10-0-1-104:~$ docker service create -dt --constraint 'node.id==3tnvltyyohbekgygvsb1jjovy' \
> --name nodejs --replicas=5 labdocker/alpineweb:latest node hello.js
v52nswafowr62xq6ib9rpzu7r
ubuntu@ip-10-0-1-104:~$ docker service ps nodejs
ID             NAME        IMAGE                       NODE          DESIRED STATE    CURRENT STATE                ERROR    POR
TS
k1jt8r2sfahw   nodejs.1    labdocker/alpineweb:latest  ip-10-0-1-104 Running          Running less than a second ago
p6rwlezix0i8   nodejs.2    labdocker/alpineweb:latest  ip-10-0-1-104 Running          Running less than a second ago
17z39pfjyb5f   nodejs.3    labdocker/alpineweb:latest  ip-10-0-1-104 Running          Running less than a second ago
oadfa4hqb5q5   nodejs.4    labdocker/alpineweb:latest  ip-10-0-1-104 Running          Running less than a second ago
i2g4d3bkkc70   nodejs.5    labdocker/alpineweb:latest  ip-10-0-1-104 Running          Running less than a second ago
```

Orchestrator Assignment

- Remove constraint/Add constraint
  - node.hostname

```
.ssh — ubuntu@ip-10-0-1-104: ~ — ssh -i k8s_lab ubuntu@13.229.126.99 — 174×11
...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==3tnvltyyohbekgygvsb1jjovy' --constraint-add 'node.hostname!=ip-10-0-1-104' nodejs
nodejs
ubuntu@ip-10-0-1-104:~$ docker service ps nodejs
ID             NAME            IMAGE                        NODE            DESIRED STATE   CURRENT STATE           ERROR   PORTS
k1jt8r2sfahw   nodejs.1        labdocker/alpineweb:latest   ip-10-0-1-104   Running         Running 3 minutes ago
or0dudbrhex2   nodejs.2        labdocker/alpineweb:latest   ip-10-0-1-163   Running         Running 2 seconds ago
p6rwlezix0i8    \_ nodejs.2    labdocker/alpineweb:latest   ip-10-0-1-104   Shutdown        Shutdown 3 seconds ago
n9kxw9v7efno   nodejs.3        labdocker/alpineweb:latest   ip-10-0-1-62    Ready           Ready 2 seconds ago
17z39pfjyb5f    \_ nodejs.3    labdocker/alpineweb:latest   ip-10-0-1-104   Shutdown        Running 2 seconds ago
oadfa4hqb5q5   nodejs.4        labdocker/alpineweb:latest   ip-10-0-1-104   Running         Running 3 minutes ago
i2g4d3bkkc70   nodejs.5        labdocker/alpineweb:latest   ip-10-0-1-104   Running         Running 3 minutes ago
```

```
...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 inspect nodejs |grep update
            "Message": "update completed"
ubuntu@ip-10-0-1-104:~$ docker service ps nodejs
ID             NAME            IMAGE                        NODE            DESIRED STATE   CURRENT STATE                ERROR   PORTS
77m2kkx4lvs8   nodejs.1        labdocker/alpineweb:latest   ip-10-0-1-62    Running         Running 24 seconds ago
k1jt8r2sfahw    \_ nodejs.1    labdocker/alpineweb:latest   ip-10-0-1-104   Shutdown        Shutdown 24 seconds ago
or0dudbrhex2   nodejs.2        labdocker/alpineweb:latest   ip-10-0-1-163   Running         Running about a minute ago
p6rwlezix0i8    \_ nodejs.2    labdocker/alpineweb:latest   ip-10-0-1-104   Shutdown        Shutdown about a minute ago
n9kxw9v7efno   nodejs.3        labdocker/alpineweb:latest   ip-10-0-1-62    Running         Running 48 seconds ago
17z39pfjyb5f    \_ nodejs.3    labdocker/alpineweb:latest   ip-10-0-1-104   Shutdown        Shutdown 48 seconds ago
p7iek0f77pji   nodejs.4        labdocker/alpineweb:latest   ip-10-0-1-163   Running         Running 12 seconds ago
oadfa4hqb5q5    \_ nodejs.4    labdocker/alpineweb:latest   ip-10-0-1-104   Shutdown        Shutdown 13 seconds ago
0rcd8pgpe4ex   nodejs.5        labdocker/alpineweb:latest   ip-10-0-1-163   Running         Running 36 seconds ago
i2g4d3bkkc70    \_ nodejs.5    labdocker/alpineweb:latest   ip-10-0-1-104   Shutdown        Shutdown 36 seconds ago
ubuntu@ip-10-0-1-104:~$
```

Workshop: node constrain

- สร้าง service nodejs เพื่อเริ่มทำงานบน swarm

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

.ssh — ubuntu@ip-10-0-1-104: ~ — ssh -i k8s_lab ubuntu@13.229.126.99 — 165×16

...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 node ls
ID                            HOSTNAME          STATUS    AVAILABILITY   MANAGER STATUS   ENGINE VERSION
4g6pqogaae5co8zxe8w6pxfd6     ip-10-0-1-62      Ready     Active                          18.06.1-ce
3tnvltyyohbekgygvsb1jjovy *   ip-10-0-1-104     Ready     Active         Leader           18.06.1-ce
ua0ajfjd4d1mksqzwr80oy2ru     ip-10-0-1-163     Ready     Active                          18.06.1-ce
ubuntu@ip-10-0-1-104:~$ docker service create -dt --constraint 'node.id==3tnvltyyohbekgygvsb1jjovy' \
> --name nodejs --replicas=5 labdocker/alpineweb:latest node hello.js
v52nswafowr62xq6ib9rpzu7r
ubuntu@ip-10-0-1-104:~$ docker service ps nodejs
ID             NAME       IMAGE                     NODE          DESIRED STATE   CURRENT STATE                ERROR   POR
TS
k1jt8r2sfahw   nodejs.1   labdocker/alpineweb:latest ip-10-0-1-104 Running         Running less than a second ago
p6rwlezix0i8   nodejs.2   labdocker/alpineweb:latest ip-10-0-1-104 Running         Running less than a second ago
17z39pfjyb5f   nodejs.3   labdocker/alpineweb:latest ip-10-0-1-104 Running         Running less than a second ago
oadfa4hqb5q5   nodejs.4   labdocker/alpineweb:latest ip-10-0-1-104 Running         Running less than a second ago
i2g4d3bkkc70   nodejs.5   labdocker/alpineweb:latest ip-10-0-1-104 Running         Running less than a second ago
```

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
```

```
...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                            18.06.1-ce
3tnvltyyohbekgygvsb1jjovy *   ip-10-0-1-104     Ready     Active          Leader            18.06.1-ce
ua0ajfjd4d1mksqzwr80oy2ru     ip-10-0-1-163     Ready     Active                            18.06.1-ce
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=nvdim' 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": "nvdim"
ubuntu@ip-10-0-1-104:~$ docker node inspect ip-10-0-1-62|grep storage
                "storage": "sata"
ubuntu@ip-10-0-1-104:~$
```

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 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  --constraint 'node.labels.storage==sas' --name nodejs --replicas=5 labdocker/alpineweb:latest node hello.js
pl6f8550yokx23bgm3l2txo91
ubuntu@ip-10-0-1-104:~$ docker service ps nodejs
ID                  NAME                IMAGE                     NODE                DESIRED STATE       CURRENT STATE           ERROR               PORTS
vkh4tyggb2z1        nodejs.1            labdocker/alpineweb:latest ip-10-0-1-104      Running             Running 7 seconds ago
7hko49pz2o3a        nodejs.2            labdocker/alpineweb:latest ip-10-0-1-104      Running             Running 6 seconds ago
sl1cqvidnitn        nodejs.3            labdocker/alpineweb:latest ip-10-0-1-104      Running             Running 6 seconds ago
3axgwaukkmgt        nodejs.4            labdocker/alpineweb:latest ip-10-0-1-104      Running             Running 7 seconds ago
i2cwhj6qmbcu        nodejs.5            labdocker/alpineweb:latest ip-10-0-1-104      Running             Running 7 seconds ago
ubuntu@ip-10-0-1-104:~$
```

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
pl6f8550yokx23bgm312txo91
ubuntu@ip-10-0-1-104:~$ docker service ps nodejs
ID                  NAME                IMAGE                        NODE                DESIRED STATE       CURRENT STATE           ERROR               PORTS
vkh4tyggb2z1        nodejs.1            labdocker/alpineweb:latest   ip-10-0-1-104       Running             Running 7 seconds ago
7hko49pz2o3a        nodejs.2            labdocker/alpineweb:latest   ip-10-0-1-104       Running             Running 6 seconds ago
sl1cqvidnitn        nodejs.3            labdocker/alpineweb:latest   ip-10-0-1-104       Running             Running 6 seconds ago
3axgwaukkmgt        nodejs.4            labdocker/alpineweb:latest   ip-10-0-1-104       Running             Running 7 seconds ago
i2cwhj6qmbcu        nodejs.5            labdocker/alpineweb:latest   ip-10-0-1-104       Running             Running 7 seconds ago
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                                  18.06.1-ce
3tnvltyyohbekgygvsb1jjovy *   ip-10-0-1-104       Ready               Active              Leader              18.06.1-ce
ua0ajfjd4d1mksqzwr80oy2ru     ip-10-0-1-163       Ready               Active                                  18.06.1-ce
ubuntu@ip-10-0-1-104:~$
```

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          STATUS    AVAILABILITY    MANAGER STATUS    ENGINE VERSION
4g6pqogaae5co8zxe8w6pxfd6     ip-10-0-1-62      Ready     Active                            18.06.1-ce
3tnvltyyohbekgygvsb1jjovy *   ip-10-0-1-104     Ready     Active          Leader            18.06.1-ce
ua0ajfjd4d1mksqzwr80oy2ru     ip-10-0-1-163     Ready     Active                            18.06.1-ce
ubuntu@ip-10-0-1-104:~$ docker node update --label-add 'physical=DELLPE820' ip-10-0-1-104
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:~$
```

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
n36z17qwvkj2euas95qxcxwpt
ubuntu@ip-10-0-1-104:~$ docker service ps nodejs
ID                  NAME                IMAGE                       NODE                DESIRED STATE       CURRENT STATE           ERROR               PORTS
zlp6rgo1p7o8        nodejs.1            labdocker/alpineweb:latest  ip-10-0-1-62        Running             Running 2 seconds ago
pyvgye34w7b1        nodejs.2            labdocker/alpineweb:latest  ip-10-0-1-104       Running             Running 2 seconds ago
w4vwxvqflz1x        nodejs.3            labdocker/alpineweb:latest  ip-10-0-1-62        Running             Running 2 seconds ago
w5isypfl0p51        nodejs.4            labdocker/alpineweb:latest  ip-10-0-1-62        Running             Running 2 seconds ago
tf2bwqa7bj1q        nodejs.5            labdocker/alpineweb:latest  ip-10-0-1-62        Running             Running 2 seconds ago
6f8tjnlyyyrs        nodejs.6            labdocker/alpineweb:latest  ip-10-0-1-163       Running             Running 3 seconds ago
7jidk3fa3opc        nodejs.7            labdocker/alpineweb:latest  ip-10-0-1-62        Running             Running 2 seconds ago
kg2bqew9g4o9        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
ozm25d536exf        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:~$
```

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
n36z17qwvkj2euas95qxcxwpt
ubuntu@ip-10-0-1-104:~$ docker service ps nodejs
ID                  NAME                IMAGE                       NODE                DESIRED STATE       CURRENT STATE           ERROR               PORTS
zlp6rgo1p7o8        nodejs.1            labdocker/alpineweb:latest  ip-10-0-1-62        Running             Running 2 seconds ago
pyvgye34w7b1        nodejs.2            labdocker/alpineweb:latest  ip-10-0-1-104       Running             Running 2 seconds ago
w4vwxvqflz1x        nodejs.3            labdocker/alpineweb:latest  ip-10-0-1-62        Running             Running 2 seconds ago
w5isypfl0p51        nodejs.4            labdocker/alpineweb:latest  ip-10-0-1-62        Running             Running 2 seconds ago
tf2bwqa7bj1q        nodejs.5            labdocker/alpineweb:latest  ip-10-0-1-62        Running             Running 2 seconds ago
6f8tjnlyyyrs        nodejs.6            labdocker/alpineweb:latest  ip-10-0-1-163       Running             Running 3 seconds ago
7jidk3fa3opc        nodejs.7            labdocker/alpineweb:latest  ip-10-0-1-62        Running             Running 2 seconds ago
kg2bqew9g4o9        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
ozm25d536exf        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:~$
```

Config and Secret

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

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>
```

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>
```

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
bni17w6tmbks39l1ri3ryvvxz
ubuntu@ip-10-0-1-104:~$ docker config ls
ID                          NAME                CREATED             UPDATED
bni17w6tmbks39l1ri3ryvvxz   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
pt2dhhjiqroqd7gb6ig9bv62n
ubuntu@ip-10-0-1-104:~$ docker secret create labdocker.com.key ~/docker_workshop_112018/Workshop-2-4-Swarm/labdocker.com.key
nu11fusm2zbl815112s8pe6nb
ubuntu@ip-10-0-1-104:~$ docker secret ls
ID                          NAME                DRIVER              CREATED                 UPDATED
pt2dhhjiqroqd7gb6ig9bv62n   labdocker.com.crt                       Less than a second ago  Less than a second ago
nu11fusm2zbl815112s8pe6nb   labdocker.com.key                       Less than a second ago  Less than a second ago
ubuntu@ip-10-0-1-104:~$ docker service create -dt \
```

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;
}
```

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:http2
```

```
praparns-MacBook-Pro:docker_workshop_112018 praparn$ curl -k https://18.136.196.130
<!DOCTYPE html>
<html>
<head>
<title>Welcome to nginx!</title>
<style>
    body {
        width: 35em;
        margin: 0 auto;
        font-family: Tahoma, Verdana, Arial, sans-serif;
    }
</style>
</head>
<body>
<h1>Welcome to nginx!</h1>
<p>If you see this page, the nginx web server is successfully installed and
working. Further configuration is required.</p>

<p>For online documentation and support please refer to
<a href="http://nginx.org/">nginx.org</a>.<br/>
Commercial support is available at
<a href="http://nginx.com/">nginx.com</a>.</p>

<p><em>Thank you for using nginx.</em></p>
</body>
</html>
praparns-MacBook-Pro:docker_workshop_112018 praparn$
```

Overlay and Ingress Network

- เพื่อให้ทุกๆ node ภายใต้ 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” เพื่อรองรับการใช้งาน service

Overlay and Ingress Network

Overlay and Ingress Network

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
lwxhyfrildn83atwsqxdb9ior
ubuntu@ip-10-0-1-104:~$ docker network ls
NETWORK ID          NAME                DRIVER              SCOPE
cdb776996466        bridge              bridge              local
7f907ae7a4c9        docker_gwbridge     bridge              local
d44f5d624dd1        host                host                local
hd63tliqe10s        ingress             overlay             swarm
a1ef67647608        none                null                local
lwxhyfrildn8        swarmnet            overlay             swarm
ubuntu@ip-10-0-1-104:~$
```

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@13.229.126.99   ...63: ~ — ssh -i k8s_lab ubuntu@13.229.98.3   ...~ — ssh -i k8s_lab ubuntu@18.136.19
ubuntu@ip-10-0-1-104:~$ docker service create -dt --name nodejs \
> --replicas=2 --network swarmnet -p 3000:3000 \
> labdocker/alpineweb:latest node hello.js
4zw4z04vcxgu0k819q1ms1ar6
ubuntu@ip-10-0-1-104:~$ docker service ps nodejs
ID              NAME        IMAGE                       NODE          DESIRED STATE   CURRENT STATE           ERROR   PORTS
t1022pjovfjs    nodejs.1    labdocker/alpineweb:latest  ip-10-0-1-104 Running         Preparing 1 second ago
51tldbd8z00y    nodejs.2    labdocker/alpineweb:latest  ip-10-0-1-163 Running         Preparing 1 second ago
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.t1022pjovfjsrxgtfficvr9mf
ubuntu@ip-10-0-1-104:~$ docker inspect nodejs.1.t1022pjovfjsrxgtfficvr9mf|grep IPAddress
            "SecondaryIPAddresses": null,
            "IPAddress": "",
                    "IPAddress": "10.255.0.8",
                    "IPAddress": "192.168.100.6",
ubuntu@ip-10-0-1-104:~$
```

```
ubuntu@ip-10-0-1-163:~$ 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.2.51tldbd8z00yaq761ac6ozpy0
ubuntu@ip-10-0-1-163:~$ docker inspect nodejs.2.51tldbd8z00yaq761ac6ozpy0|grep IPAddress
            "SecondaryIPAddresses": null,
            "IPAddress": "",
                    "IPAddress": "10.255.0.9",
                    "IPAddress": "192.168.100.7",
ubuntu@ip-10-0-1-163:~$
```

Workshop: Overlay and Ingress

- ตรวจสอบ IP Address ของ service และทดสอบ Ping ข้าม 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
ubuntu@ip-10-0-1-104:~$ docker service inspect nodejs |more
[
    {
        "ID": "4zw4z04vcxgu0k819q1ms1ar6",
        "Version": {
            "Index": 233
        },
        "CreatedAt": "2018-10-21T17:12:30.830220563Z",
        "UpdatedAt": "2018-10-21T17:12:30.831796908Z",
        "Spec": {
            "Name": "nodejs",
            "Labels": {},
            "TaskTemplate": {
                "ContainerSpec": {
                    "Image": "labdocker/alpineweb:latest@sha256:a7aa70c78120ed126487aecbf49db1054f1ff131858516c91137c3accbb4a333",
                    "Args": [
                        "node",
                        "hello.js"
                    ],
                    "Init": false,
                    "TTY": true,
                    "StopGracePeriod": 10000000000,
                    "DNSConfig": {},
                    "Isolation": "default"
                },
```

```
...orkshop_112018 — Terminal MAC Pro — -bash    ...04: ~ — ssh -i k8s_lab ubuntu@13.229.126.99    ...63: ~ — ssh -i k8s_lab ul
                }
            ],
            "VirtualIPs": [
                {
                    "NetworkID": "hd63tliqe10s27mu9w4ilckui",
                    "Addr": "10.255.0.7/16"
                },
                {
                    "NetworkID": "lwxhyfrildn83atwsqxdb9ior",
                    "Addr": "192.168.100.5/24"
                }
            ]
        }
    }
]
ubuntu@ip-10-0-1-104:~$
```

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
bfc3f7a2e264        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
^C
--- 192.168.100.5 ping statistics ---
2 packets transmitted, 2 packets received, 0% packet loss
round-trip min/avg/max = 0.076/0.096/0.116 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=1.114 ms
64 bytes from 192.168.100.7: seq=1 ttl=64 time=0.226 ms
^C
--- 192.168.100.7 ping statistics ---
2 packets transmitted, 2 packets received, 0% packet loss
round-trip min/avg/max = 0.226/0.670/1.114 ms
ubuntu@ip-10-0-1-104:~$
```

```
ubuntu@ip-10-0-1-163:~$ docker ps
CONTAINER ID        IMAGE                       COMMAND                  CREATED             STATUS              PORTS               NAMES
cf75d3a7074b        labdocker/alpineweb:latest  "node hello.js node …"   7 minutes ago       Up 7 minutes        3000/tcp            nodejs.2.51tldbd8z00yaq761ac6ozpy0
ubuntu@ip-10-0-1-163:~$ docker exec -it nodejs.2.51tldbd8z00yaq761ac6ozpy0 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=0.100 ms
64 bytes from 192.168.100.5: seq=1 ttl=64 time=0.062 ms
^C
--- 192.168.100.5 ping statistics ---
2 packets transmitted, 2 packets received, 0% packet loss
round-trip min/avg/max = 0.062/0.081/0.100 ms
ubuntu@ip-10-0-1-163:~$ docker exec -it nodejs.2.51tldbd8z00yaq761ac6ozpy0 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
^C
--- 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:~$
```

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$
```

HA Manager Role

- Add redundancy swarm manager in swarm cluster
  - ทำการ 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                            18.06.1-ce
3tnvltyyohbekgygvsb1jjovy *   ip-10-0-1-104     Ready     Active          Leader            18.06.1-ce
ua0ajfjd4d1mksqzwr80oy2ru     ip-10-0-1-163     Ready     Active                            18.06.1-ce
ubuntu@ip-10-0-1-104:~$ docker node update --role manager ip-10-0-1-163
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                            18.06.1-ce
3tnvltyyohbekgygvsb1jjovy *   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
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
3tnvltyyohbekgygvsb1jjovy *   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:~$
```

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
3tnvltyyohbekgygvsb1jjovy *   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.
praparns-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
3tnvltyyohbekgygvsb1jjovy     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
3tnvltyyohbekgygvsb1jjovy     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:~$
```

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
4g6pqogaae5co8zxe8w6pxfd6     ip-10-0-1-62      Ready     Active          Reachable         18.06.1-ce
3tnvltyyohbekgygvsb1jjovy *   ip-10-0-1-104     Ready     Active          Reachable         18.06.1-ce
ua0ajfjd4d1mksqzwr80oy2ru     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
4g6pqogaae5co8zxe8w6pxfd6     ip-10-0-1-62      Ready     Drain                             18.06.1-ce
3tnvltyyohbekgygvsb1jjovy *   ip-10-0-1-104     Ready     Active          Reachable         18.06.1-ce
ua0ajfjd4d1mksqzwr80oy2ru     ip-10-0-1-163     Ref       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
4g6pqogaae5co8zxe8w6pxfd6     ip-10-0-1-62      Down      Drain                             18.06.1-ce
3tnvltyyohbekgygvsb1jjovy *   ip-10-0-1-104     Ready     Active          Reachable         18.06.1-ce
ua0ajfjd4d1mksqzwr80oy2ru     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
3tnvltyyohbekgygvsb1jjovy *   ip-10-0-1-104     Ready     Active          Reachable         18.06.1-ce
ua0ajfjd4d1mksqzwr80oy2ru     ip-10-0-1-163     Ready     Active          Leader            18.06.1-ce
ubuntu@ip-10-0-1-104:~$
```

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   ...4: ~ — ssh -i k8s_lab ubuntu@13.229.224.202   ...63: ~ — 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-1-1qnpobqw0hilpc5i90sz803tlhyi1h4k84uok10y814bvhyqf2-3u9rqxfwhbhw52qwqeeiewybl 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
mydkcrgwkurxyjjb1kf14odl7 *   ip-10-0-1-62      Ready     Active          Reachable         18.06.1-ce
3tnvltyyohbekgygvsb1jjovy     ip-10-0-1-104     Ready     Active          Reachable         18.06.1-ce
ua0ajfjd4d1mksqzwr80oy2ru     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-1-1qnpobqw0hilpc5i90sz803tlhyi1h4k84uok10y814bvhyqf2-3u9rqxfwhbhw52qwqeeiewybl 10.0.1.62:2377

ubuntu@ip-10-0-1-62:~$
```

Workshop: HA Manager Role

- ทำการ 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
4g6pqogaae5co8zxe8w6pxfd6     ip-10-0-1-62      Ready     Active          Reachable         18.06.1-ce
3tnvltyyohbekgygvsb1jjovy *   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:~$
```

```
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
3tnvltyyohbekgygvsb1jjovy *   ip-10-0-1-104     Ready     Active          Reachable         18.06.1-ce
ua0ajfjd4d1mksqzwr80oy2ru     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
4g6pqogaae5co8zxe8w6pxfd6     ip-10-0-1-62      Ready     Drain                             18.06.1-ce
3tnvltyyohbekgygvsb1jjovy *   ip-10-0-1-104     Ready     Active          Reachable         18.06.1-ce
ua0ajfjd4d1mksqzwr80oy2ru     ip-10-0-1-163     Ready     Active          Leader            18.06.1-ce
ubuntu@ip-10-0-1-104:~$
```

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 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 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>
```

Workshop: Swarm Compose

- Optimize for WorkLoad

Workshop: Swarm Compose

Workshop: Swarm Compose

```yaml
version: '3.3'
services:
  webcache:
    image: labdocker/cluster:webcache
    container_name: nginx
    deploy:
      mode: global
      update_config:
        parallelism: 1
        delay: 10s
      restart_policy:
        condition: on-failure
        delay: 30s
        max_attempts: 5
        window: 60s
      endpoint_mode: vip
    depends_on:
      - webservice
      - cachedb
      - maindb
    networks:
      webpublic:
        aliases:
        - webcache
      webinternal:
        aliases:
        - webcache
    ports:
      - "80:80"

  webservice:
    image: labdocker/cluster:webservice
    container_name: webservice
    deploy:
      mode: replicated
      replicas: 6
      update_config:
        parallelism: 2
        delay: 10s
      restart_policy:
        condition: on-failure
        delay: 30s
        max_attempts: 3
        window: 60s
      endpoint_mode: vip
    depends_on:
      - cachedb
      - maindb
    networks:
      webinternal:
        aliases:
        - webservice
    ports:
      - "5000:5000"
```

Workshop: Swarm Compose

```yaml
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
```

```yaml
81    networks:
82        webpublic:
83          driver: overlay
84          ipam:
85            driver: default
86            config:
87            - subnet: 192.168.100.0/24
88        webinternal:
89          driver: overlay
90          ipam:
91            driver: default
92            config:
93            - subnet: 192.168.101.0/24
```

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$
```

Workshop: Swarm Compose

```
docker@swarm-mng:/Share_DockerToolbox/python_restfulset$ docker stack ps webservice
ID                  NAME                                          IMAGE                          NODE                DESIRED STATE       CURRENT STATE
        ERROR                       PORTS
mcj0ux4x25uv        webservice_webcache.wwzx60rvl5res39v0lxir2gfh   labdocker/cluster:webcache     swarm-node2         Running             Running 46 seconds ag
o
ol223n2c2htp        webservice_webcache.0k95omm28xfef22thummpz3uf   labdocker/cluster:webcache     swarm-mng           Running             Running 49 seconds ag
o
fj0qu565u8mt        webservice_webcache.q814dhi3bzl4sze4zmjpiy3qa   labdocker/cluster:webcache     swarm-node1         Running             Running 50 seconds ag
o
kw8oprgn8b3v        webservice_webcache.wwzx60rvl5res39v0lxir2gfh   labdocker/cluster:webcache     swarm-node2         Shutdown            Failed about a minute
 ago    "task: non-zero exit (1)"
a4foun5hzks3        webservice_webcache.0k95omm28xfef22thummpz3uf   labdocker/cluster:webcache     swarm-mng           Shutdown            Failed about a minute
 ago    "task: non-zero exit (1)"
horm2noczrvt        webservice_webcache.q814dhi3bzl4sze4zmjpiy3qa   labdocker/cluster:webcache     swarm-node1         Shutdown            Failed about a minute
 ago    "task: non-zero exit (1)"
vhutisinb7np        webservice_webcache.wwzx60rvl5res39v0lxir2gfh   labdocker/cluster:webcache     swarm-node2         Shutdown            Failed about a minute
 ago    "task: non-zero exit (1)"
ktjzx8thiq6k        webservice_webcache.q814dhi3bzl4sze4zmjpiy3qa   labdocker/cluster:webcache     swarm-node1         Shutdown            Failed about a minute
 ago    "task: non-zero exit (1)"
7siaoo4hnvmh        webservice_webcache.0k95omm28xfef22thummpz3uf   labdocker/cluster:webcache     swarm-mng           Shutdown            Failed about a minute
 ago    "task: non-zero exit (1)"
axdh6t2ua971        webservice_cachedb.wwzx60rvl5res39v0lxir2gfh    labdocker/redis:latest         swarm-node2         Running             Running about a minut
e ago
t8jc8aopm6wu        webservice_cachedb.q814dhi3bzl4sze4zmjpiy3qa    labdocker/redis:latest         swarm-node1         Running             Running about a minut
e ago
32d2fao3cjcu        webservice_cachedb.0k95omm28xfef22thummpz3uf    labdocker/redis:latest         swarm-mng           Running             Running about a minut
e ago
mtbhxlkuwjg7        webservice_webservice.1                         labdocker/cluster:webservice   swarm-node1         Running             Running 39 seconds ag
o
nw2jky8drn42         \_ webservice_webservice.1                     labdocker/cluster:webservice   swarm-mng           Shutdown            Failed about a minute
 ago    "task: non-zero exit (1)"
ivhofu5u9kxe        webservice_maindb.1                             labdocker/mysql:latest         swarm-node2         Running             Running about a minut
```

Workshop: Swarm Compose

```
praparns-MacBook-Pro:~ praparn$ docker-machine ls
NAME          ACTIVE   DRIVER       STATE     URL                         SWARM   DOCKER        ERRORS
labdocker     -        virtualbox   Running   tcp://192.168.99.103:2376           Unknown       Unable to query docker version: Get https://192.168.99.103:2376/
v1.15/version: x509: certificate is valid for 192.168.99.100, not 192.168.99.103
swarm-mng     -        virtualbox   Running   tcp://192.168.99.104:2376           v17.06.0-ce
swarm-node1   -        virtualbox   Running   tcp://192.168.99.105:2376           v17.06.0-ce
swarm-node2   -        virtualbox   Running   tcp://192.168.99.106:2376           v17.06.0-ce
praparns-MacBook-Pro:~ praparn$ export Server_IP=192.168.99.104
praparns-MacBook-Pro:~ praparn$ curl http://$Server_IP:$Server_Port/
<H1> Welcome Page from Container Python Lab </H1>Checkpoint Date/Time: Sun Aug 13 17:27:58 2017
```

Workshop: Swarm Compose

http://192.168.99.104 | http://192.168.99.104 | http://192.168.99.104 | No Environment

POST http://192.168.99.104/users/insertuser Params Send Save

Authorization Headers (1) Body Pre-request Script Tests Code

form-data x-www-form-urlencoded raw binary JSON (application/json)

```
{"uid": "1", "user":"Praparn Luangphoonlap", "descripe":"Slave"}
```

Body Cookies Headers (5) Tests Status: 200 OK Time: 49 ms

Pretty Raw Preview HTML

```
########### Record was added ###########
```

http://192.168.99.104 | http://192.168.99.104 | http://192.168.99.104 | http://192.168.99.104 | No Environment

GET http://192.168.99.104/users/1 Params Send Save

Authorization Headers Body Pre-request Script Tests Code

Type No Auth

Body Cookies Headers (6) Tests Status: 200 OK Time: 49 ms

Pretty Raw Preview HTML

```
Praparn Luangphoonlap(Database Direct)
```

Workshop: Swarm Compose

http://192.168.99.1 | http://192.168.99.1 | http://192.168.99.1 | http://192.168.99.1 | http://192.168.99.1 | No Environment

GET http://192.168.99.104/users/removeuser/1 Params Send Save

Authorization Headers Body Pre-request Script Tests Code

Type No Auth

Body Cookies Headers (5) Tests Status: 200 OK Time: 33 ms

Pretty Raw Preview HTML

```
1 ########### Record was deleted (Both Database Cache) ###########
```

Portainer for Docker

...

portainer for Docker

- portainer.io เป็น open source management ที่สามารถใช้บริหารจัดการ docker-machine ผ่านหน้า GUI web ได้อย่างมีประสิทธิภาพแบบ On premise
- การใช้งาน portainer จะใช้การบริหารจัดการผ่านหน้าเว็บ (x.x.x.x:9443)

portainer for Docker

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

Home
Dashboard

Help support portainer admin
my account log out

Node info

Name	labdocker
Docker version	18.01.0-ce
CPU	4
Memory	1 GB

1 Containers — 1 running, 0 stopped

31 Images — 2.4 GB

5 Volumes

4 Networks

Container list
Containers

Help support portainer admin
my account log out

Containers — Search Settings

Start | Stop | Kill | Restart | Pause | Resume | Remove | + Add container

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

Items per page 10

docker

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

portainer.io 1.16.2

Create container
Containers > Add container

Help support portainer admin
my account log out

Name: nginx

Image configuration

Name: labdocker/nginx:latest
Registry: DockerHub

Always pull the image

Ports configuration

Publish all exposed ports

Port mapping map additional port

host 80 → container 80 TCP UDP

Access control

Enable access control

Administrators
I want to restrict the management of this resource to administrators only

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

Actions

Deploy the container

Advanced container settings

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.
Commercial support is available at nginx.com.

Thank you for using nginx.

Container list
Containers

Help support portainer | admin
my account | log out

Containers | Search | Settings

Start | Stop | Kill | Restart | Pause | Resume | Remove | + Add container

Name	State Filter	Quick actions	Stack	Image	IP Address	Published Ports	Ownership
nginx	running		-	labdocker/nginx:latest	172.17.0.3	80:80	administrators
vigilant_pare	running		-	portainer/portainer	172.17.0.2	9000:9000	public

Items per page 10

docker

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
Containers > nginx > Stats

Help support portainer admin
my account log out

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

Memory
10 MB
8 MB
6 MB
4 MB
2 MB
0.0B
22:28:31 22:28:38 22:28:43

CPU usage

CPU
1.0%
0.8%
0.6%
0.4%
0.2%
0.0%
22:28:31 22:28:38 22:28:43

Network usage

RX on eth0 TX on eth0
3 kB
2 kB
1 kB
0.0B
22:28:31 22:28:38 22:28:43

Processes Search

UID	PID	PPID	C	STIME	TTY	TIME	CMD

portainer.io

ACTIVE ENDPOINT
local

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

PORTAINER SETTINGS

Container console
Containers > nginx > Console

Help support portainer admin
my account log out

Console

Exec into container as `default user` using command `sh` Disconnect

```
/usr/sbin # ls
add-shell     arping      chroot      deluser     fakeidentd  ftpd        killall5    nanddump    nginx       rdate       readprofile  sendmail    udhcpd
addgroup      brctl       crond       dnsd        fbset       httpd       loadfont    nandwrite   ntpd        rdev        remove-shell setfont
adduser       chpasswd    delgroup    ether-wake  fdformat    inetd       lspci       nbd-client  powertop    readahead   rfkill       setlogcons
/usr/sbin #
```

Docker: The Next-Gen of Virtualization

docker

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

Q&A

By Praparn Luengphoonlap
Email: eva10409@gmail.com