

1st
Week

Agenda

- ▷ [30m] Ice-Breaking
- ▷ [10m] Ground-Rule
- ▷ [10m] Prerequisite
- ▷ [30m] Container / Docker Overview
- ▷ [20m] Docker Hands-On
- ▷ [10m] Breaktime
- ▷ [50m] Kubernetes Overview
- ▷ [20m] Quiz

Ice-Breaking

- ▷ 자기 소개
- ▷ 설문조사 결과 공유
- ▷ 인증샷

Ground-Rule

- ▷ 가급적 지각/결석 하지 않기
- ▷ 가급적 Camera 켜 놓고 수업 참여하기
- ▷ 가급적 적극적으로 참여하기
- ▷ 3시간이 넘더라도 배고프다고 화내지 않기
- ▷ Slack 잊지 않기
- ▷ 꼭 끝까지 함께하기

Prerequisite

▷ 자료 공유

- <https://github.com/whatwant-school/advanced-kubernetes>

▷ Desktop/Laptop

- OS: Windows 10 (VirtualBox 설치 가능 환경)
- Mem: 8GB 이상 (16GB 추천)
- Network: 공유기

///

Breaktime

///

Container / Docker Overview

A Brief History of Containers: From the 1970s Till Now

1979: Unix V7 – chroot 도입

2000: FreeBSD Jails – 서비스와 고객 서비스를 구분하기 위해 여러 개의 독립적이고 작은 시스템(jails)으로 분할

2001: Linux VServer - Jails와 유사하게, 리소스(파일 시스템, 네트워크 주소, 메모리)를 분할 할 수 있는 운영 체제 가상화를 Linux 커널 패치로 구현

2004: Solaris Containers – 첫 번째 공개 베타 출시

2005: Open VZ (Open Virtuzzo) - 가상화, 격리, 리소스 관리 및 체크 포인트를 위해 패치 된 Linux 커널을 사용하는 Linux 용 운영 체제 수준의 가상화 기술

2006: Process Containers - 2006년 Google 출시. 리소스 사용량(CPU, Mem, Disk I/O, NW)을 제한, 계산 및 격리하도록 설계. 1년 후 "cgroups"으로 이름 변경.

2008: LXC (LinuX Containers) – 컨테이너 관리자의 가장 완벽한 최초 구현. cgroups & namespace를 사용하여 구현.

2011: Warden – CloudFoundry에서 초기는 LXC를 사용하고 나중에 자체 구현으로 대체. cgroups, namespace 및 프로세스 수명주기 관리 서비스 포함.

2013: LMCTFY (Let Me Contain That For You) – Linux 애플리케이션 컨테이너를 제공하는 Google 컨테이너 스택의 오픈 소스 버전. 2015년 중단.

2013: Docker - 컨테이너 인기 폭발. 초기 단계 LXC 사용, 추후 자체 라이브러리 libcontainer로 대체.

2014: Kubernetes (Google)

2015: Kubernetes to CNCF

2016: The Importance of Container Security Is Revealed – DevSecOps

2017: Container Tools Become Mature – 컨테이너 도구의 성숙

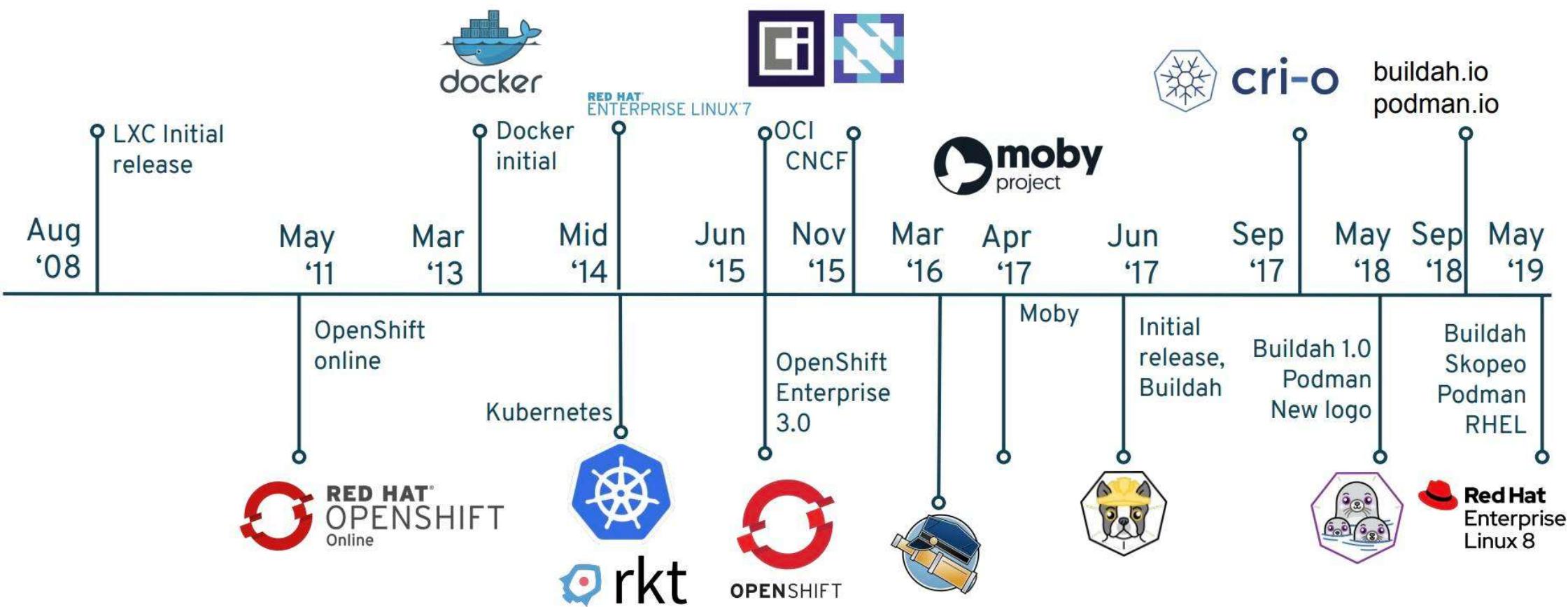
2017: containerd to CNCF (Docker)

2018: The Gold Standard – 시장 표준

2019: A Shifting Landscape - 변화

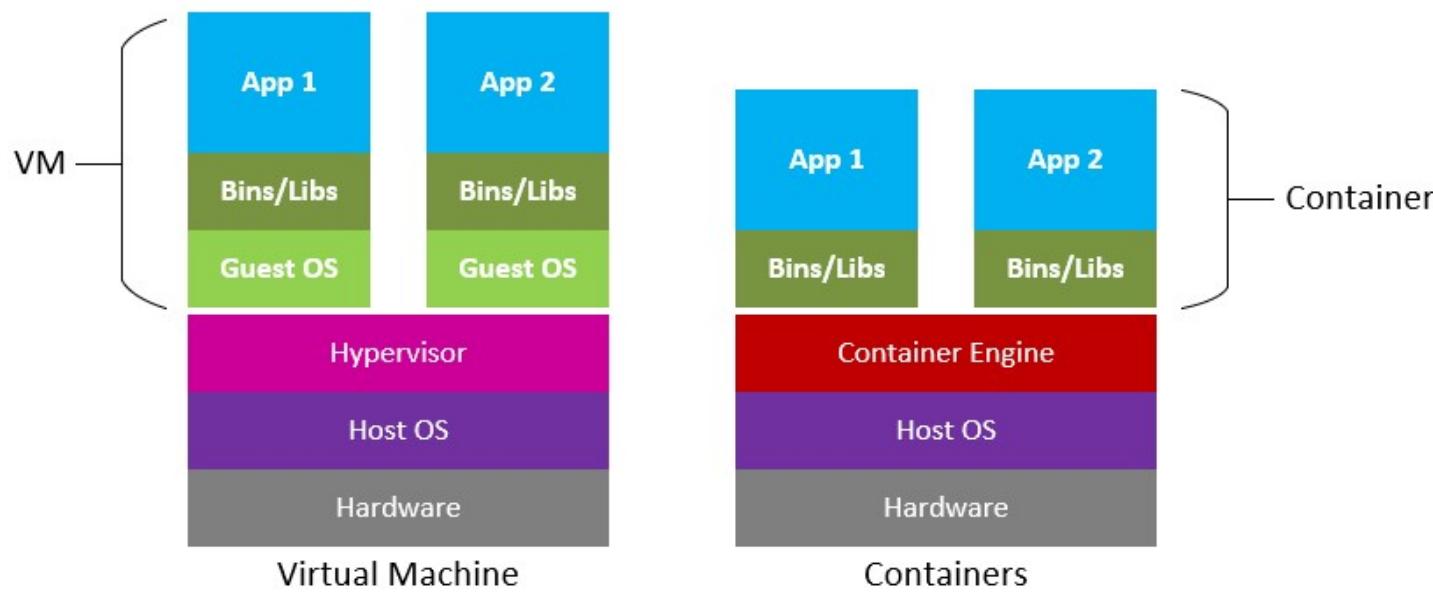
※ 참고 : <https://blog.aquasec.com/a-brief-history-of-containers-from-1970s-chroot-to-docker-2016>

Evolution of the open-source container



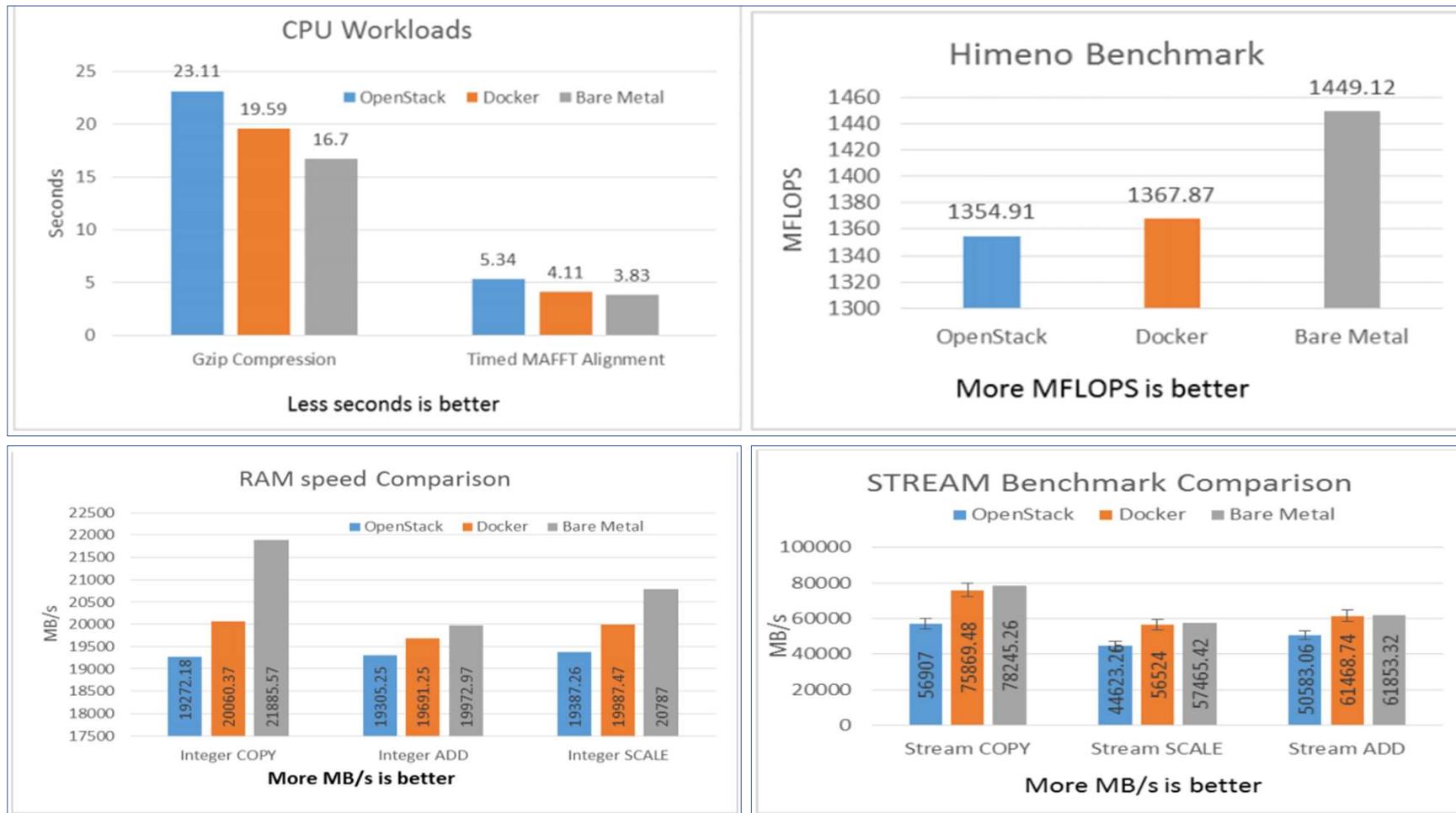
※ 참고 : <https://developer.ibm.com/tutorials/multi-architecture-cri-o-container-images-for-red-hat-openshift/>

Virtual Machine vs. Containers



※ 참고 : <https://cloudblogs.microsoft.com/opensource/2019/07/15/how-to-get-started-containers-docker-kubernetes/>

Docker Container vs. Openstack VM vs. Bare Metal Server



※ 참고 : <http://ijeeecs.iaescore.com/index.php/IJEECS/article/view/7925>

cgroups (control groups)

- ▷ Linux kernel feature that **limits**, accounts for, and isolates the resource usage (**CPU**, **memory**, **disk I/O**, **network**, etc.) of a **collection of processes**.

```
> cat /proc/cgroups
```

| #subsys_name | hierarchy | num_cgroups | enabled |
|--------------|-----------|-------------|---------|
| cpuset | 10 | 1 | 1 |
| cpu | 5 | 65 | 1 |
| cpuacct | 5 | 65 | 1 |
| blkio | 8 | 65 | 1 |
| memory | 12 | 99 | 1 |
| devices | 2 | 65 | 1 |
| freezer | 9 | 1 | 1 |
| net_cls | 7 | 1 | 1 |
| perf_event | 6 | 1 | 1 |
| net_prio | 7 | 1 | 1 |
| hugetlb | 3 | 1 | 1 |
| pids | 11 | 68 | 1 |
| rdma | 4 | 1 | 1 |

- cgroups를 관리하는 방법으로 'cgroupfs'와 'systemd' 2 종류 존재
 - 또한 cgroups 정책도 'v1', 'v2' 2 종류가 공존
 - Kubernetes v1.22에서 이와 관련 정책 변경이 있음

```
› ls -al /sys/fs/cgroup
```

```
합계 0
drwxr-xr-x 15 root root 380 12월 24 23:02 .
drwxr-xr-x  9 root root   0 12월 24 23:02 ..
dr-xr-xr-x  4 root root   0 12월 24 23:02 blkio
lrwxrwxrwx  1 root root  11 12월 24 23:02 cpu -> cpu,cpuacct
dr-xr-xr-x  4 root root   0 12월 24 23:02 cpu,cpuacct
lrwxrwxrwx  1 root root  11 12월 24 23:02 cpuacct -> cpu,cpuacct
dr-xr-xr-x  2 root root   0 12월 24 23:02 cpuset
dr-xr-xr-x  4 root root   0 12월 24 23:02 devices
dr-xr-xr-x  2 root root   0 12월 24 23:02 freezer
dr-xr-xr-x  2 root root   0 12월 24 23:02 hugetlb
dr-xr-xr-x  4 root root   0 12월 24 23:02 memory
lrwxrwxrwx  1 root root  16 12월 24 23:02 net_cls -> net_cls,net_prio
dr-xr-xr-x  2 root root   0 12월 24 23:02 net_cls,net_prio
lrwxrwxrwx  1 root root  16 12월 24 23:02 net_prio -> net_cls,net_prio
dr-xr-xr-x  2 root root   0 12월 24 23:02 perf_event
dr-xr-xr-x  4 root root   0 12월 24 23:02 pids
dr-xr-xr-x  2 root root   0 12월 24 23:02 rdma
dr-xr-xr-x  5 root root   0 12월 24 23:02 systemd
dr-xr-xr-x  5 root root   0 12월 24 23:02 unified
```

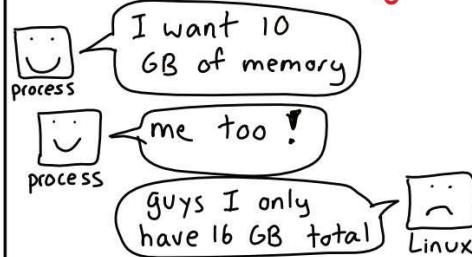
※ 참고 : <https://en.wikipedia.org/wiki/Cgroups>

cgroups (control groups)

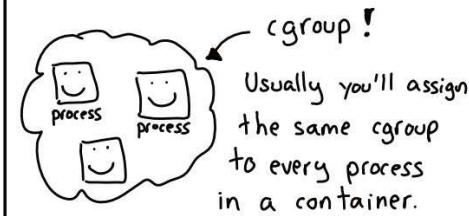
JULIA EVANS
@b0rk

cgroups

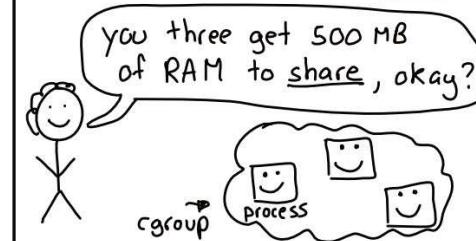
processes can use
a lot of memory



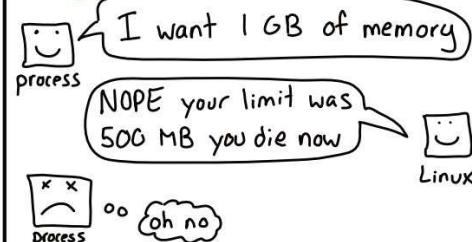
a cgroup is a
group of processes



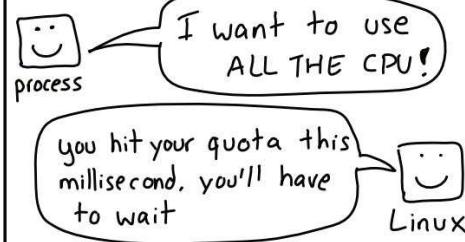
cgroups have
memory/CPU limits



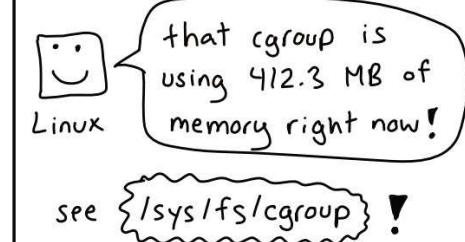
use too much memory:
get OOM killed



use too much CPU:
get slowed down



cgroups track
memory & CPU usage



namespaces

- ▷ a feature of the Linux kernel that **partitions kernel resources** such that one set of processes sees one set of resources while another set of processes sees a different set of resources.

```
> sudo ls -al /proc/1/ns
```

```
합계 0
dr-x---x--x 2 root root 0 12월 24 23:32 .
dr-xr-xr-x 9 root root 0 12월 24 23:02 ..
lrwxrwxrwx 1 root root 0 12월 24 23:32 cgroup -> 'cgroup:[4026531835]'
lrwxrwxrwx 1 root root 0 12월 24 23:32 ipc -> 'ipc:[4026531839]'
lrwxrwxrwx 1 root root 0 12월 24 23:32 mnt -> 'mnt:[4026531840]'
lrwxrwxrwx 1 root root 0 12월 24 23:32 net -> 'net:[4026531992]'
lrwxrwxrwx 1 root root 0 12월 24 23:32 pid -> 'pid:[4026531836]'
lrwxrwxrwx 1 root root 0 12월 24 23:32 pid_for_children -> 'pid:[4026531836]'
lrwxrwxrwx 1 root root 0 12월 24 23:32 user -> 'user:[4026531837]'
lrwxrwxrwx 1 root root 0 12월 24 23:32 uts -> 'uts:[4026531838]'
```

same namespaces

```
> sudo ls -al /proc/2/ns
```

```
합계 0
dr-x---x--x 2 root root 0 12월 24 23:36 .
dr-xr-xr-x 9 root root 0 12월 24 23:02 ..
lrwxrwxrwx 1 root root 0 12월 24 23:36 cgroup -> 'cgroup:[4026531835]'
lrwxrwxrwx 1 root root 0 12월 24 23:36 ipc -> 'ipc:[4026531839]'
lrwxrwxrwx 1 root root 0 12월 24 23:36 mnt -> 'mnt:[4026531840]'
lrwxrwxrwx 1 root root 0 12월 24 23:36 net -> 'net:[4026531992]'
lrwxrwxrwx 1 root root 0 12월 24 23:36 pid -> 'pid:[4026531836]'
lrwxrwxrwx 1 root root 0 12월 24 23:36 pid_for_children -> 'pid:[4026531836]'
lrwxrwxrwx 1 root root 0 12월 24 23:36 user -> 'user:[4026531837]'
lrwxrwxrwx 1 root root 0 12월 24 23:36 uts -> 'uts:[4026531838]'
```

※ 참고 : https://en.wikipedia.org/wiki/Linux_namespaces

namespaces

JULIA EVANS
@b0rk

namespaces

inside a container,
things look different



I only see 4
processes in 'ps aux',
that's weird...

Commands that
will look different

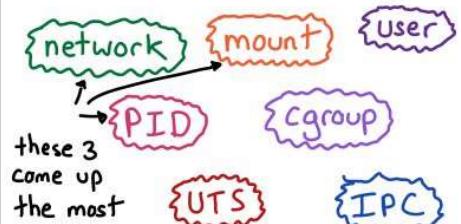
- ps aux (less processes!)
- mount & df
- netstat -tulpn
(different open ports!)
- hostname
- ... and LOTS more

Why those commands
look different:
⋮ namespaces ⋮



I'm in a different
PID namespace so
'ps aux' shows different
processes!

every process has 7
kinds of namespaces



there's a default
("host") namespace



"outside a
container" just
means "using the
default namespaces"

processes can have
any combination
of namespaces



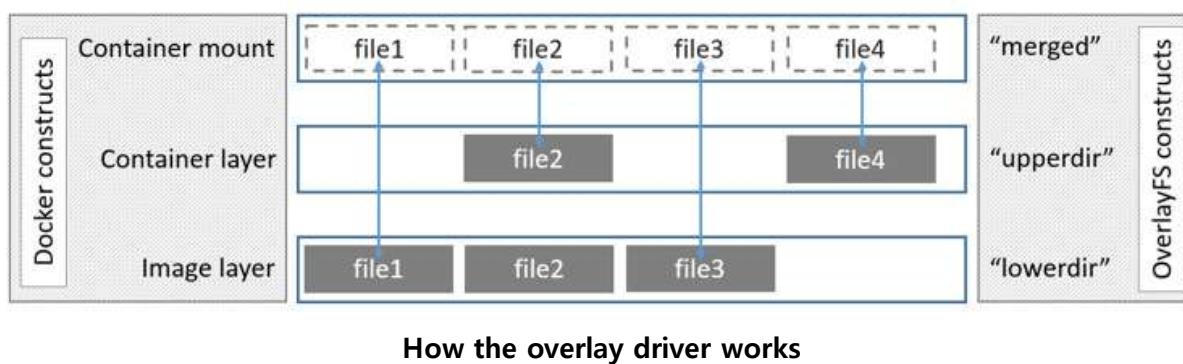
I'm using the host
network namespace
but my own mount
namespace!

♡ this? more at wizardzines.com

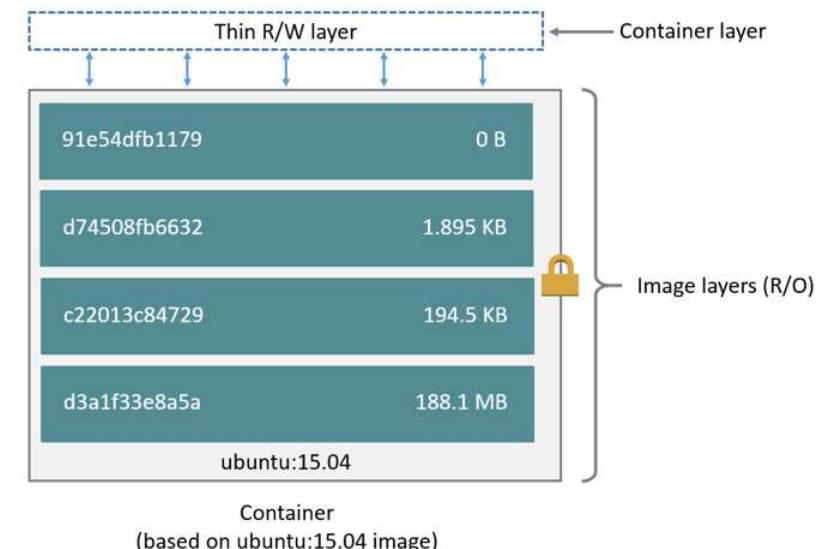
Docker Storage Driver

▷ Docker supports the following storage drivers

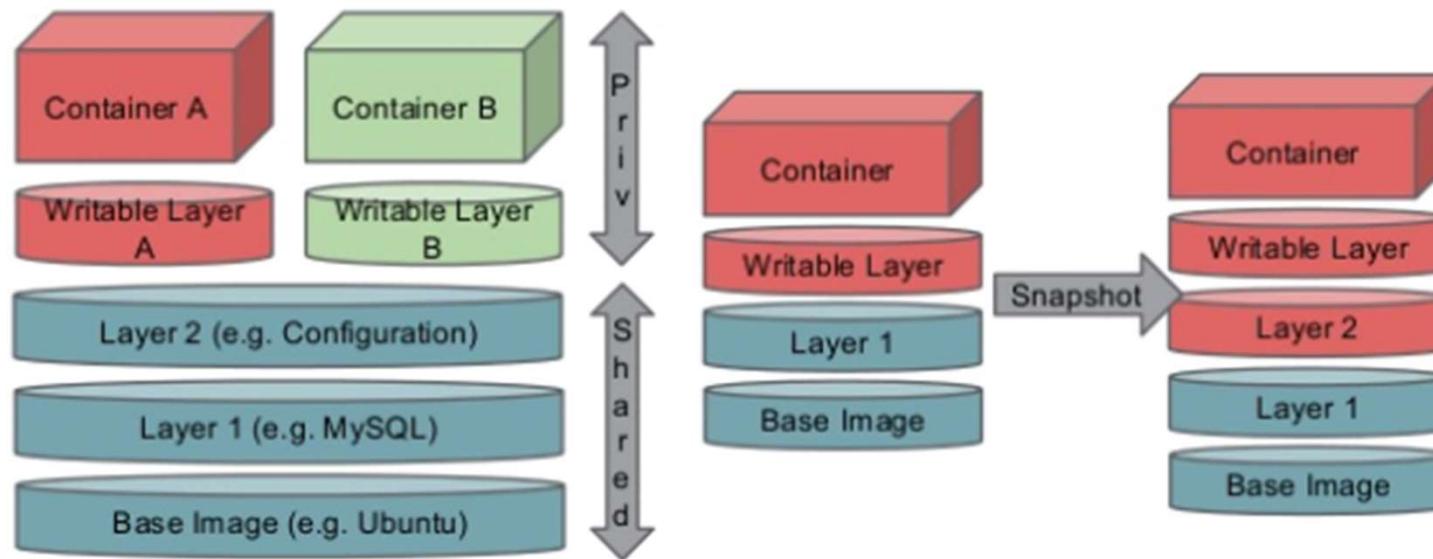
- **overlay2** : 기본 드라이버
- **aufs** : Docker 18.06 및 이전 버전에서 사용
- **fuse-overlayfs** : Rootless 지원 안되는 호스트에서 Rootless Docker를 사용할 때
- **devicemapper** : production 환경을 위해서는 direct-lvm 필요.
- **btrfs** and **zfs** : "snapshots" 같은 고급 기능을 지원하지만 설치와 유지보수가 까다로움.
- **vfs** : 테스트 목적으로만 사용하는 것을 권장



※ 참고 : <https://docs.docker.com/storage/storagedriver/overlayfs-driver/>



Docker Storage Driver



※ 참고 : <https://docs.docker.com/storage/storagedriver/overlayfs-driver/>

Container Network Model (CNM)

▷ Sandbox

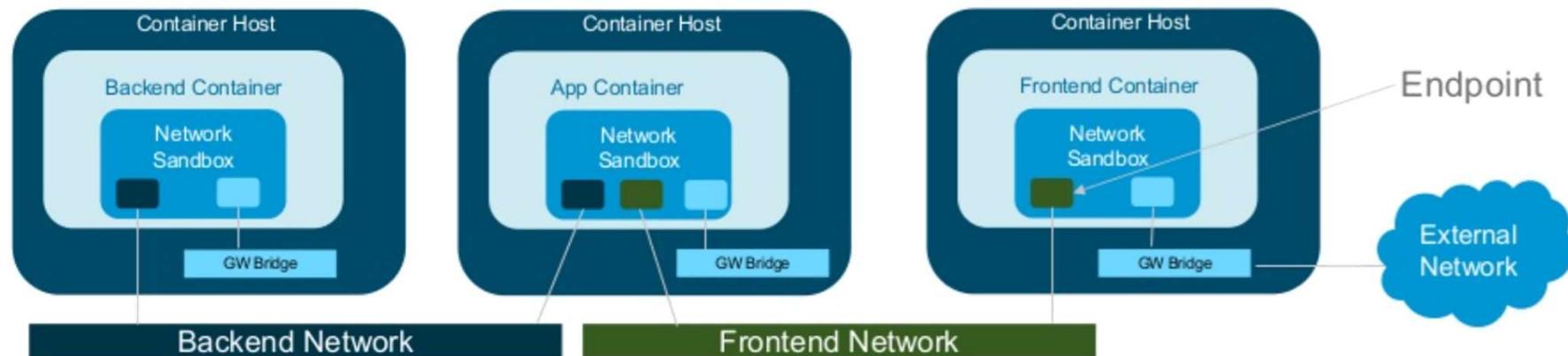
- A Sandbox contains the configuration of a container's network stack.
- This includes management of the container's interfaces, routing table and DNS settings.
- An implementation of a Sandbox could be a Linux Network Namespace, a FreeBSD Jail or other similar concept.

▷ Endpoint

- An Endpoint joins a Sandbox to a Network.
- An implementation of an Endpoint could be a veth pair, an Open vSwitch internal port or similar

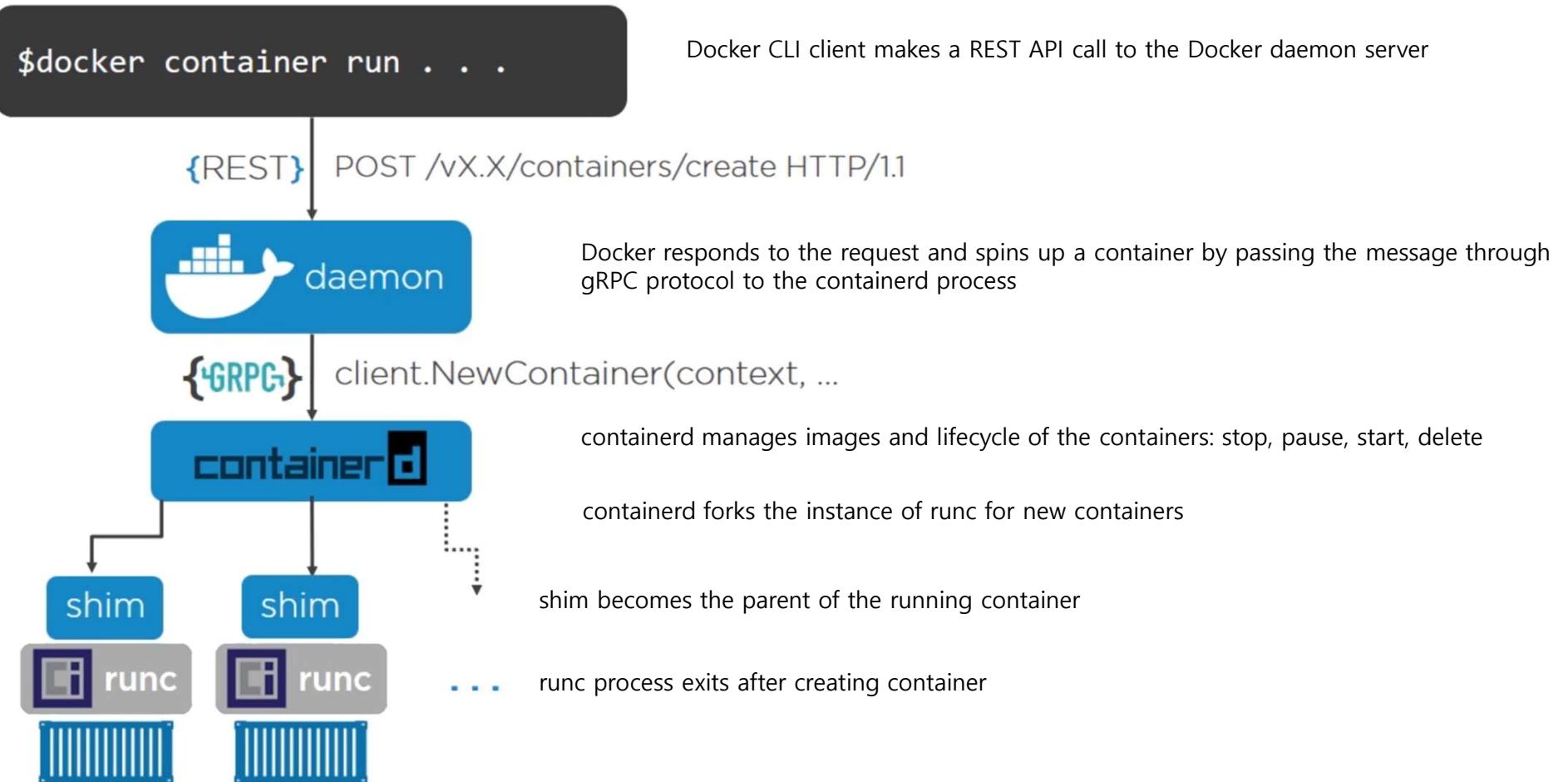
▷ Network

- A Network is a group of Endpoints that are able to communicate with each-other directly.
- An implementation of a Network could be a VXLAN Segment, a Linux bridge, a VLAN, etc.



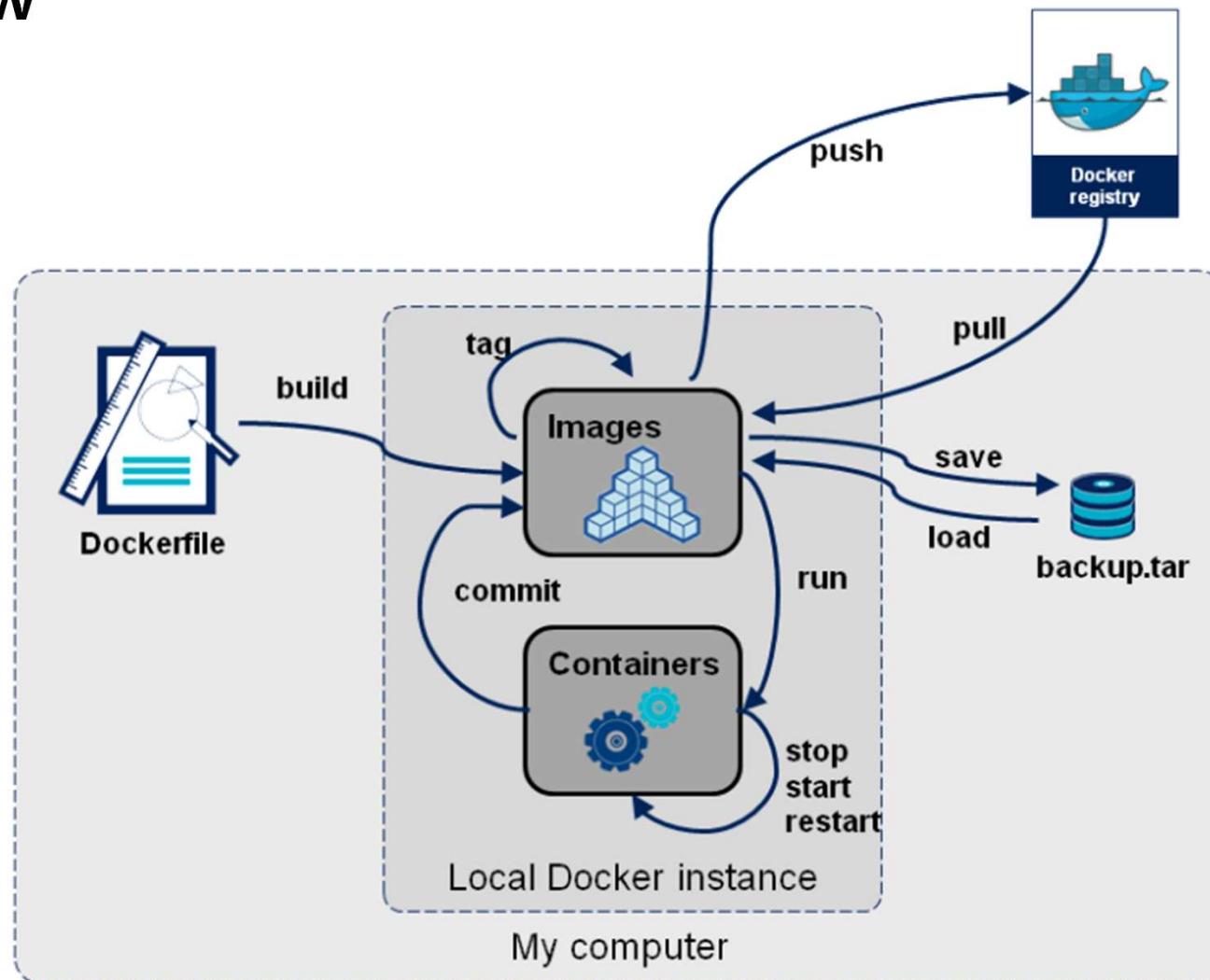
※ 참고 : <https://www.slideshare.net/OpenNetworkingSummit/container-networking-deep-dive>

Docker Engine Architecture



※ 참고 : <https://betterprogramming.pub/docker-for-front-end-developers-c758a44e622f>

Docker flow



※ 참고 : <https://blog.wonizz.tk/2019/07/31/docker-dockerfile/>

Dockerfile

```
1  # fetch node v4 LTS codename argon
2  FROM node:argon
3
4  # Request samplename build argument
5  ARG samplename
6
7  # Create app directory
8  RUN mkdir -p /usr/src/spfx-samples
9  WORKDIR /usr/src/spfx-samples
10
11 #Install app dependencies
12 RUN git clone https://github.com/SharePoint/sp-dev-fx-webparts.git .
13 WORKDIR /usr/src/spfx-samples/samples/$samplename
14
15 # install gulp on a global scope
16 RUN npm install gulp -g
17
18 # RUN ["npm", "install", "gulp"]
19 RUN npm install
20 RUN npm cache clean
21
22 # Expose required ports
23 EXPOSE 4321 35729 5432
24
25 # Run sample
26 CMD ["gulp", "serve"]
27
```

※ 참고 : <https://n8d.at/how-to-run-sharepoint-pattern-and-practices-samples-through-docker/>

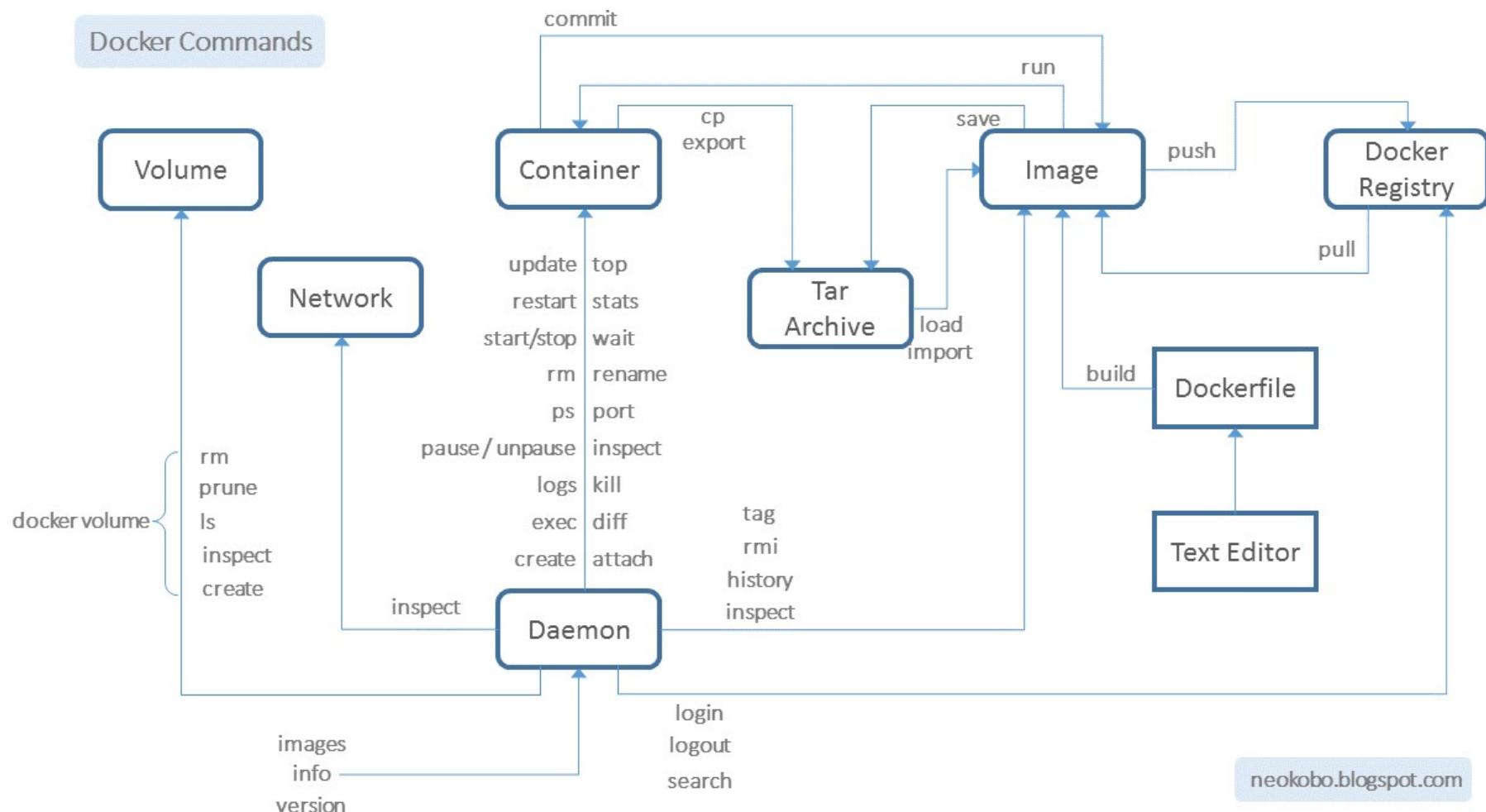
Docker Hub (Registry)

The screenshot shows a web browser displaying the Docker Hub search results for 'ubuntu' and 'alpine'. The search bar at the top contains 'hub.docker.com/search?type=image'. The Docker Hub interface includes a blue header with 'docker hub', a search bar, and navigation links for 'Explore', 'Pricing', 'Sign In', and 'Sign Up'. Below the header, there are tabs for 'Docker', 'Containers', and 'Plugins', with 'Containers' selected. On the left, there are filters for 'Images' (including 'Verified Publisher' and 'Official Images') and 'Categories' (such as 'Analytics', 'Application Frameworks', 'Application Infrastructure', etc.). The main content area shows two search results:

- ubuntu** - Official Image
Updated 20 days ago
Ubuntu is a Debian-based Linux operating system based on free software.
Tags: Container, Linux, riscv64, x86-64, 386, PowerPC 64 LE, ARM 64, ARM, IBM Z, Base Images, Operating Systems
Downloads: 1B+, Stars: 10K+
- alpine** - Official Image
Updated a month ago
A minimal Docker image based on Alpine Linux with a complete package index and only 5 MB in size!
Tags: Container, Linux, 386, riscv64, PowerPC 64 LE, ARM 64, ARM, x86-64, IBM Z, Featured Images, Base Images, Operating Systems
Downloads: 1B+, Stars: 8.3K

※ 참고 : <https://hub.docker.com/>

Docker command



※ 참고 : <http://neokobo.blogspot.com/2017/12/docker-command-flowchart.html>

///

Docker Hands-On

Agenda

- ▷ VirtualBox Install
- ▷ Ubuntu Install
- ▷ Docker Install
- ▷ Hands-On
- ▷ Tip #1 - DockerHub
- ▷ Tip #2 - Katacoda

VirtualBox Install

The screenshot shows a web browser displaying the official VirtualBox website at [virtualbox.org](https://www.virtualbox.org/). The page features a large blue header with the "VirtualBox" logo and the text "Welcome to VirtualBox.org!". Below the header, there is a sidebar with links to "About", "Screenshots", "Downloads", "Documentation", "End-user docs", "Technical docs", "Contribute", and "Community". The main content area contains text about the product's capabilities, a "Download VirtualBox 6.1" button, and a "Hot picks:" section. On the right side, there is a "News Flash" sidebar listing several recent releases with dates like May 17th, 2021, November 22nd, 2021, October 19th, 2021, July 28th, 2021, July 20th, 2021, April 29th, 2021, and April 29th, 2021.

VirtualBox is a powerful x86 and AMD64/Intel64 [virtualization](#) product for enterprise as well as home use. Not only is VirtualBox an extremely feature rich, high performance product for enterprise customers, it is also the only professional solution that is freely available as Open Source Software under the terms of the GNU General Public License (GPL) version 2. See "[About VirtualBox](#)" for an introduction.

Presently, VirtualBox runs on Windows, Linux, Macintosh, and Solaris hosts and supports a large number of [guest operating systems](#) including but not limited to Windows (NT 4.0, 2000, XP, Server 2003, Vista, Windows 7, Windows 8, Windows 10), DOS/Windows 3.x, Linux (2.4, 2.6, 3.x and 4.x), Solaris and OpenSolaris, OS/2, and OpenBSD.

VirtualBox is being actively developed with frequent releases and has an ever growing list of features, supported guest operating systems and platforms it runs on. VirtualBox is a community effort backed by a dedicated company: everyone is encouraged to contribute while Oracle ensures the product always meets professional quality criteria.

Download VirtualBox 6.1

Hot picks:

- Pre-built virtual machines for developers at [Oracle Tech Network](#)
- Hyperbox** Open-source Virtual Infrastructure Manager [project site](#)
- phpVirtualBox** AJAX web interface [project site](#)

News Flash

- Important May 17th, 2021**
We're hiring!
Looking for a new challenge? We're hiring a VirtualBox senior developer in 3D area (Europe/Russia/India).
- New November 22nd, 2021**
VirtualBox 6.1.30 released!
Oracle today released a 6.1 maintenance release which improves stability and fixes regressions. See the [Changelog](#) for details.
- New October 19th, 2021**
VirtualBox 6.1.28 released!
Oracle today released a 6.1 maintenance release which improves stability and fixes regressions. See the [Changelog](#) for details.
- New July 28th, 2021**
VirtualBox 6.1.26 released!
Oracle today released a 6.1 maintenance release which improves stability and fixes regressions. See the [Changelog](#) for details.
- New July 20th, 2021**
VirtualBox 6.1.24 released!
Oracle today released a 6.1 maintenance release which improves stability and fixes regressions. See the [Changelog](#) for details.
- New April 29th, 2021**
VirtualBox 6.1.22 released!
Oracle today released a 6.1 maintenance release which improves stability and fixes regressions. See the [Changelog](#) for details.

※ 참고 : <https://www.virtualbox.org/>

Ubuntu Install

The screenshot shows a web browser window displaying the Ubuntu download page at ubuntu.com/download/desktop. The page has a dark header with the Canonical logo, a search bar, and navigation links for Enterprise, Developer, Community, Download, and Desktop. The Desktop link is highlighted in orange. Below the header, there's a navigation bar with tabs for Downloads, Overview, Cloud, IoT, Raspberry Pi, Server, Desktop (which is active), Xilinx, Alternative downloads, and Ubuntu flavours. The main content area features a large heading "Download Ubuntu Desktop" and a section for "Ubuntu 20.04.3 LTS". It includes a description of LTS support, a link to "Ubuntu 20.04 LTS release notes", and a "Recommended system requirements" section. At the bottom, there are two checked checkboxes: "2 GHz dual core processor or better" and "Internet access is helpful". To the right of the requirements, there's a green "Download" button with a white arrow icon. Below the button, text mentions alternative download options like torrents and network installers.

※ 참고 : <https://ubuntu.com/download/desktop>

Docker Install

Ubuntu 설치 확인!

```
> lsb_release -a

No LSB modules are available.
Distributor ID: Ubuntu
Description:    Ubuntu 18.04.6 LTS
Release:      18.04
Codename:     bionic
```

파키지 및 버전 확인



<https://download.docker.com/linux/ubuntu/dists/>

```
> wget https://download.docker.com/linux/ubuntu/dists/bionic/pool/stable/amd64/containerd.io_1.4.12-1_amd64.deb
> wget https://download.docker.com/linux/ubuntu/dists/bionic/pool/stable/amd64/docker-ce-cli_20.10.12~3-0~ubuntu-bionic_amd64.deb
> wget https://download.docker.com/linux/ubuntu/dists/bionic/pool/stable/amd64/docker-ce_20.10.12~3-0~ubuntu-bionic_amd64.deb

> sudo dpkg --install ./containerd.io_1.4.12-1_amd64.deb
> sudo dpkg --install ./docker-ce-cli_20.10.12~3-0~ubuntu-bionic_amd64.deb
> sudo dpkg --install ./docker-ce_20.10.12~3-0~ubuntu-bionic_amd64.deb

> sudo usermod -aG docker $USER

> docker --version
Docker version 20.10.12, build e91ed57

> docker run hello-world
```

Just do - docker build

index.html

```
<!doctype html>
<html lang="en">
<head>
  <meta charset="utf-8">
  <title>Advanced Kubernetes</title>
</head>
<body>
  <h2>Hello from Nginx container</h2>
</body>
</html>
```

Dockerfile

```
FROM nginx:latest
COPY ./index.html /usr/share/nginx/html/index.html
```

예제 파일 내용을 복사한 후 docker image 를 만듭니다

```
> git clone https://github.com/whatwant-school/advanced-kubernetes.git
> cd advanced-kubernetes/01-week/
> docker build -t webserver .
...
> docker images
```

| REPOSITORY | TAG | IMAGE ID | CREATED | SIZE |
|-------------|--------|--------------|----------------|--------|
| webserver | latest | 6223db426adf | 42 seconds ago | 141MB |
| nginx | latest | f6987c8d6ed5 | 3 days ago | 141MB |
| hello-world | latest | feb5d9fea6a5 | 3 months ago | 13.3kB |

※ 참고 : <https://www.docker.com/blog/how-to-use-the-official-nginx-docker-image/>

Just do - docker run / ps

빌드한 이미지를 (container로) 실행하자

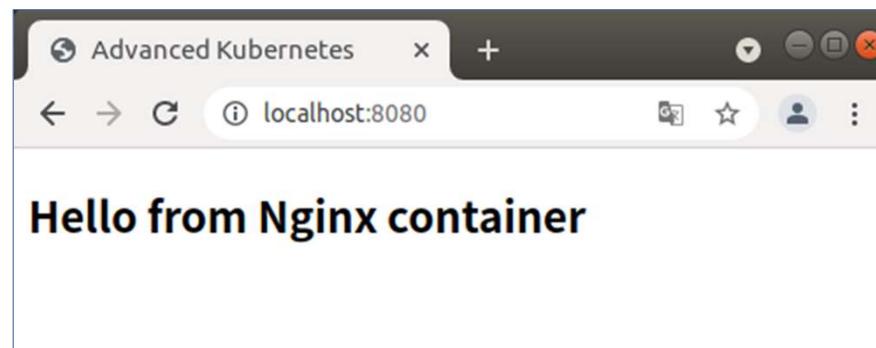
```
> docker run -it --rm -d -p 8080:80 --name web webserver
```

```
4ade5015b7f84e3f115331072a18038819e39dbf65c54cd7413d153900b93264
```

```
> docker ps -al
```

| CONTAINER ID | IMAGE | COMMAND | CREATED | STATUS | PORTS | NAMES |
|--------------|-----------|--------------------------|----------------|---------------|---------------------------------------|-------|
| 4ade5015b7f8 | webserver | "/docker-entrypoint...." | 42 seconds ago | Up 41 seconds | 0.0.0.0:8080->80/tcp, :::8080->80/tcp | web |

Chrome을 통해서 웹페이지 확인!



Just do - docker stop / images / rmi

동작하고 있는 container를 중단해보자

```
> docker ps -al
```

| CONTAINER ID | IMAGE | COMMAND | CREATED | STATUS | PORTS | NAMES |
|--------------|-----------|--------------------------|----------------|---------------|---------------------------------------|-------|
| 4ade5015b7f8 | webserver | "/docker-entrypoint...." | 42 seconds ago | Up 41 seconds | 0.0.0.0:8080->80/tcp, :::8080->80/tcp | web |

```
> docker stop 4ade5015b7f8
```

```
4ade5015b7f8
```

등록되어 있는 image를 확인하고, 삭제해보자

```
> docker images
```

| REPOSITORY | TAG | IMAGE ID | CREATED | SIZE |
|------------|--------|--------------|-------------|-------|
| webserver | latest | 6223db426adf | 7 hours ago | 141MB |
| nginx | latest | f6987c8d6ed5 | 4 days ago | 141MB |

```
> docker rmi webserver
```

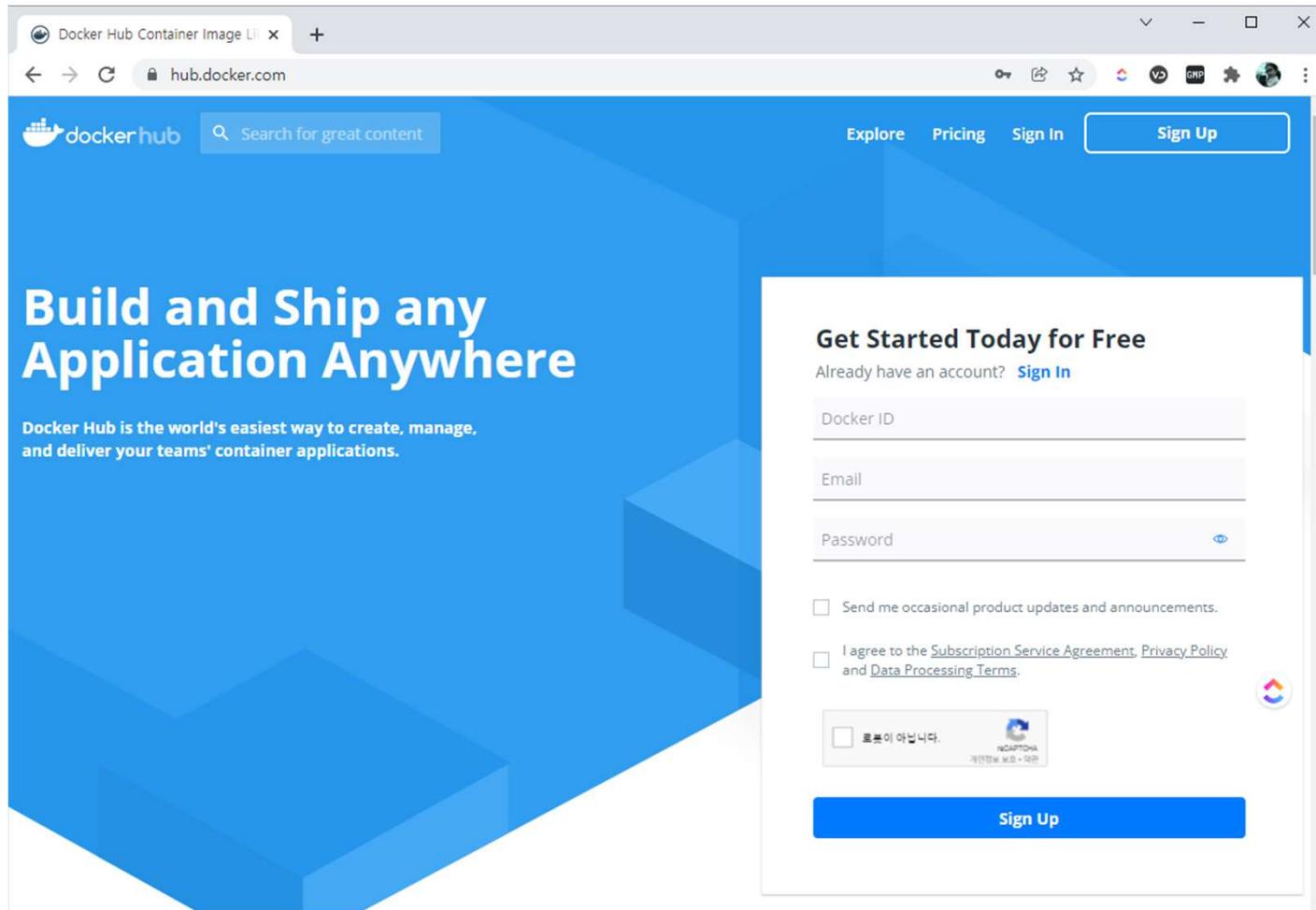
```
Untagged: webserver:latest
Deleted: sha256:6223db426adf9a43a506b324924f450b6c466ea2ee60cf8cc165d923d9806f3c
Deleted: sha256:572dc68dceacf746d58efe16951ec47936d46d68e2d27e089a05f8cd79738895
```

///

Tip #1 - DockerHub

DockerHub - Sign Up / In

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



DockerHub - Create Repository

The image displays two screenshots of the Docker Hub 'Create Repository' interface. The top screenshot shows the initial search screen where the user has entered 'whatwant' in the search bar and selected 'Create Repository'. The bottom screenshot shows the detailed 'Create Repository' form.

Screenshot 1: Initial Search Screen

- URL: hub.docker.com
- Search bar: whatwant
- Action button: Create Repository

Screenshot 2: Create Repository Form

- URL: hub.docker.com/repository/create?namespace=whatwant
- Form fields:
 - Namespace: whatwant
 - Name: sample-web
 - Description: advanced kubernetes
- Visibility settings:
 - Public (selected): Appears in Docker Hub search results
 - Private: Only visible to you
- Pro tip (CLI instructions):

```
docker tag local-image:tagname new-repo:tagname
docker push new-repo:tagname
```
- Note: Make sure to change *tagname* with your desired image repository tag.
- Buttons: Cancel, Create

DockerHub - docker login / tag / push

DockerHub 권한을 위해 로그인이 필요하다

```
> docker login
```

...

```
Login Succeeded
```

앞에서 진행해보았던 이미지를 재활용 해보자

```
> git clone https://github.com/whatwant-school/advanced-kubernetes.git
```

```
> cd advanced-kubernetes/01-week/
```

```
> docker build -t webserver .
```

...

업로드하기 전에 tagging을 하고 push

```
> docker tag webserver:latest whatwant/sample-web:v0.1
```

```
> docker push whatwant/sample-web:v0.1
```

The push refers to repository [docker.io/whatwant/sample-web]

ba032a7dca37: Pushed

51a4ac025eb4: Mounted from library/nginx

...

2edcec3590a4: Mounted from library/nginx

v0.1: digest: sha256:f47f5ecb4f828d28f930a9c262f33066c5ca59e6b3f72c2ac882c71e3e981e31 size: 1777

docker build -t webserver .

DockerHub - Repository

The screenshot shows a web browser displaying the DockerHub repository page for 'whatwant/sample-web'. The URL in the address bar is `hub.docker.com/repository/docker/whatwant/sample-web`. The page has a blue header with the DockerHub logo, a search bar, and navigation links for Explore, Repositories, Organizations, and Help. A yellow 'Upgrade' button and a user profile for 'whatwant' are also visible.

The main content area shows the repository details:

- General** tab is selected.
- whatwant / sample-web** is the repository name.
- Tags**: advanced kubernetes
- Last pushed**: 2 minutes ago
- Docker commands**: To push a new tag to this repository, use the command `docker push whatwant/sample-web:tagname`.
- Public View** button.

Tags and Scans section:

- This repository contains 1 tag(s).
- Table:

| TAG | OS | PULLED | PUSHED |
|------|----|---------------|---------------|
| v0.1 | | 2 minutes ago | 2 minutes ago |
- [See all](#)

VULNERABILITY SCANNING - DISABLED (with an 'Enable' link).

Automated Builds section:

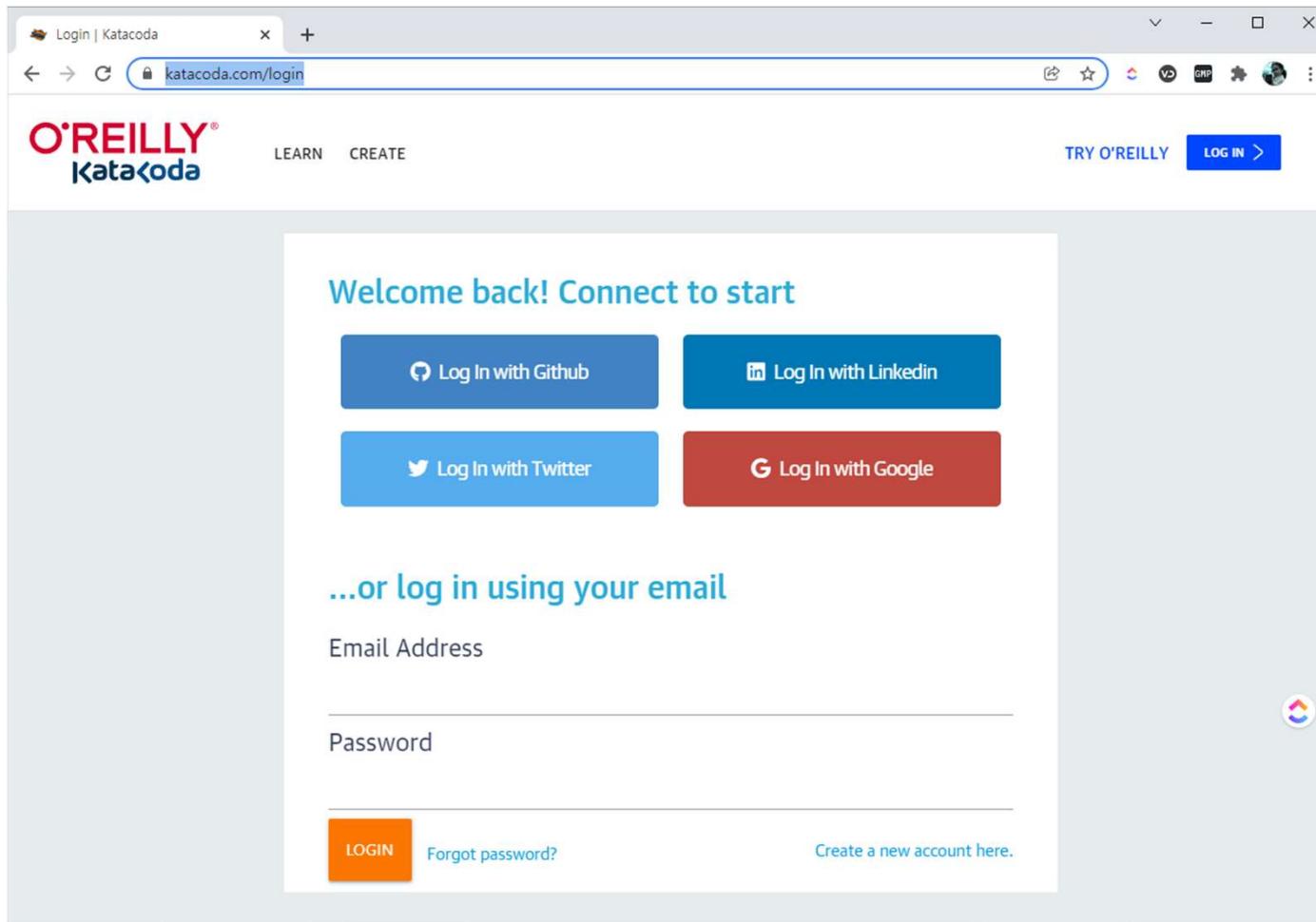
- Manually pushing images to Hub? Connect your account to GitHub or Bitbucket to automatically build and tag new images whenever your code is updated, so you can focus your time on creating.
- Available with Pro, Team and Business subscriptions.
- [Upgrade to Pro](#) and [Learn more](#) buttons.

///

Tip #2 - Katacoda

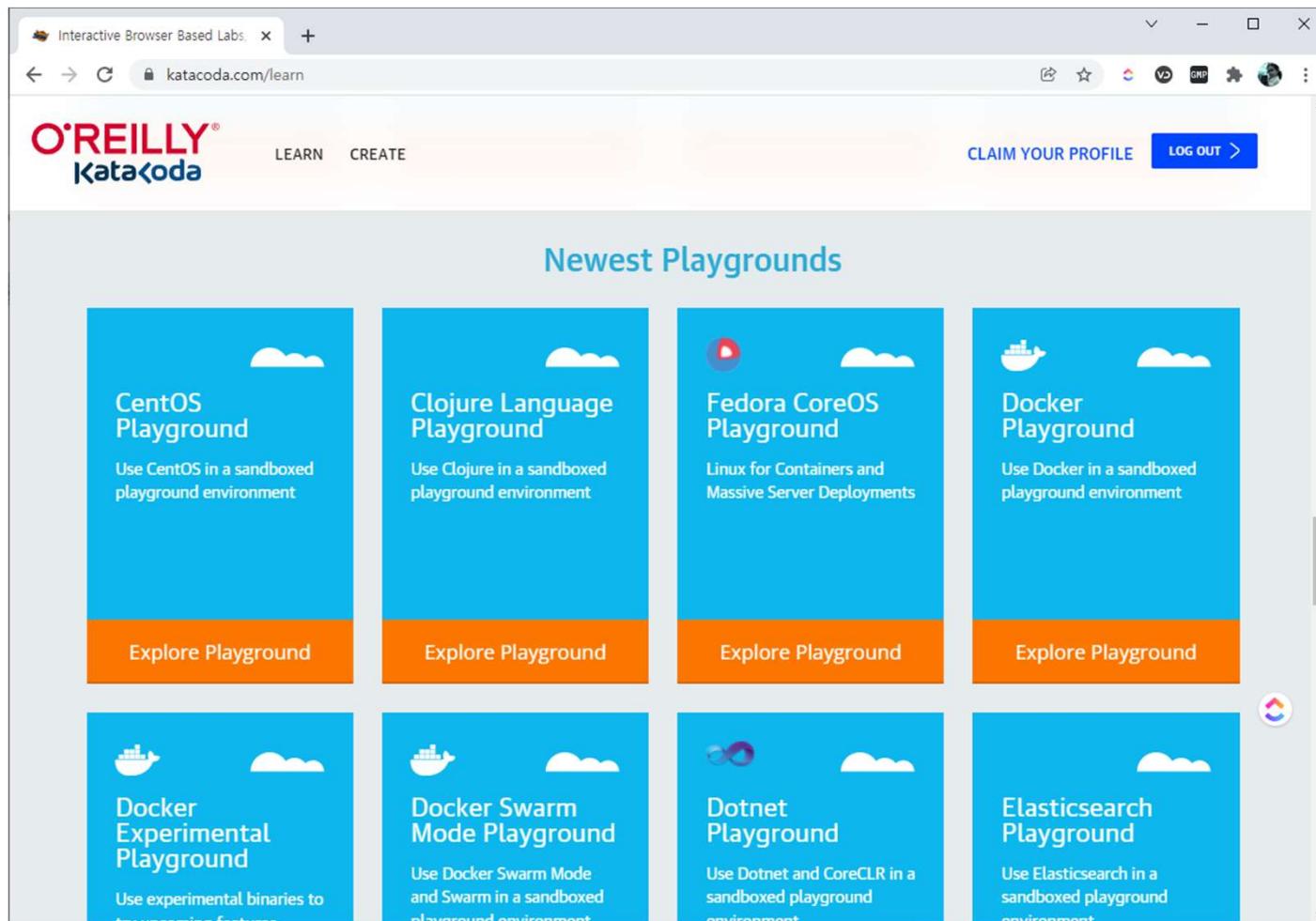
Katacoda - Sign Up / In

<https://www.katacoda.com/login>



Katacoda - Docker Playground

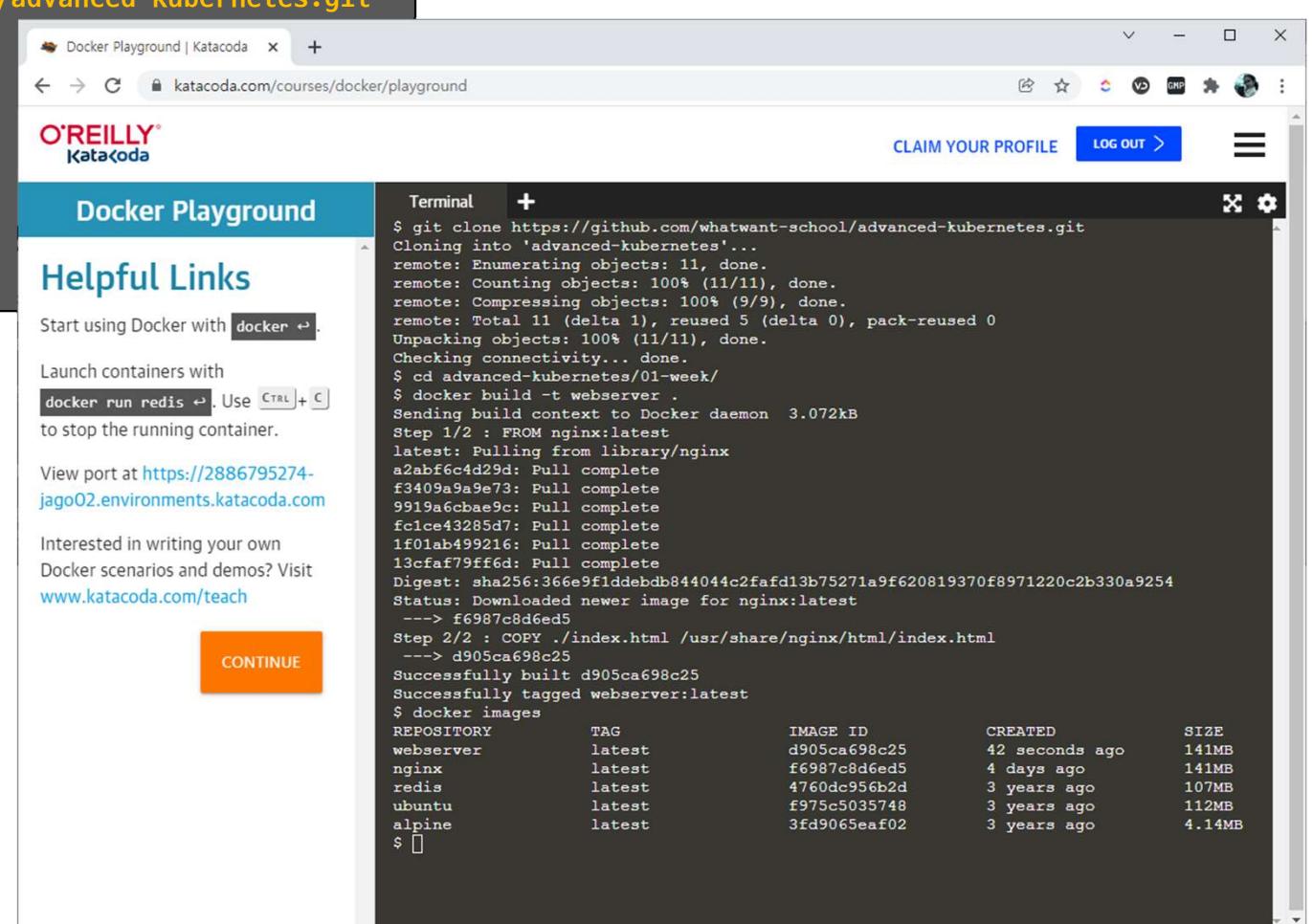
<https://www.katacoda.com/courses/docker/playground>



Katacoda - docker build / images

정말 고맙게도, git도 지원해준다.

```
> git clone https://github.com/whatwant-school/advanced-kubernetes.git  
> cd advanced-kubernetes/01-week/  
  
> docker build -t webserver .  
...  
  
> docker images
```



The screenshot shows a browser window for the Katacoda Docker Playground. The left sidebar contains helpful links for Docker usage, including how to start with Docker, launch containers with Docker Run, and view ports. A large orange 'CONTINUE' button is at the bottom of this sidebar. The main area is a terminal window displaying the command-line session:

```
$ git clone https://github.com/whatwant-school/advanced-kubernetes.git  
Cloning into 'advanced-kubernetes'...  
remote: Enumerating objects: 11, done.  
remote: Counting objects: 100% (11/11), done.  
remote: Compressing objects: 100% (9/9), done.  
remote: Total 11 (delta 1), reused 5 (delta 0), pack-reused 0  
Unpacking objects: 100% (11/11), done.  
Checking connectivity... done.  
$ cd advanced-kubernetes/01-week/  
$ docker build -t webserver .  
Sending build context to Docker daemon 3.072kB  
Step 1/2 : FROM nginx:latest  
latest: Pulling from library/nginx  
a2abf6c4d29d: Pull complete  
f3409a9a9e73: Pull complete  
9919a6cbae9c: Pull complete  
fc1ce43285d7: Pull complete  
1f01ab499216: Pull complete  
13cfaf79ff6d: Pull complete  
Digest: sha256:366e9f1ddebdb844044c2fafd13b75271a9f620819370f8971220c2b330a9254  
Status: Downloaded newer image for nginx:latest  
--> f6987c8d6ed5  
Step 2/2 : COPY ./index.html /usr/share/nginx/html/index.html  
--> d905ca698c25  
Successfully built d905ca698c25  
Successfully tagged webserver:latest  
$ docker images
```

| REPOSITORY | TAG | IMAGE ID | CREATED | SIZE |
|------------|--------|--------------|----------------|--------|
| webserver | latest | d905ca698c25 | 42 seconds ago | 141MB |
| nginx | latest | f6987c8d6ed5 | 4 days ago | 141MB |
| redis | latest | 4760dc956b2d | 3 years ago | 107MB |
| ubuntu | latest | f975c5035748 | 3 years ago | 112MB |
| alpine | latest | 3fd9065eaf02 | 3 years ago | 4.14MB |

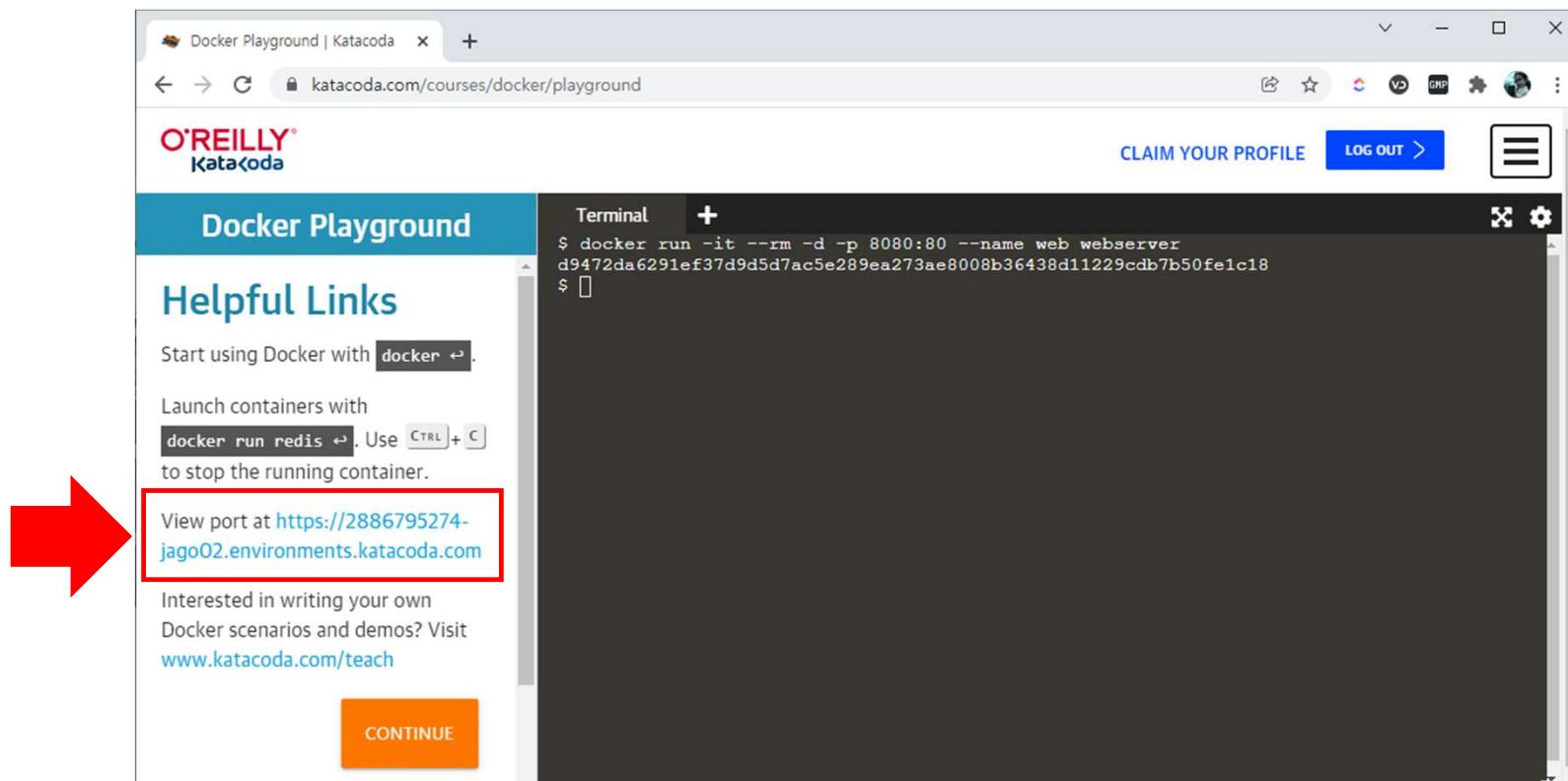
Katacoda - docker run

실행 ???

```
> docker run -it --rm -d -p 8080:80 --name web webserver
```

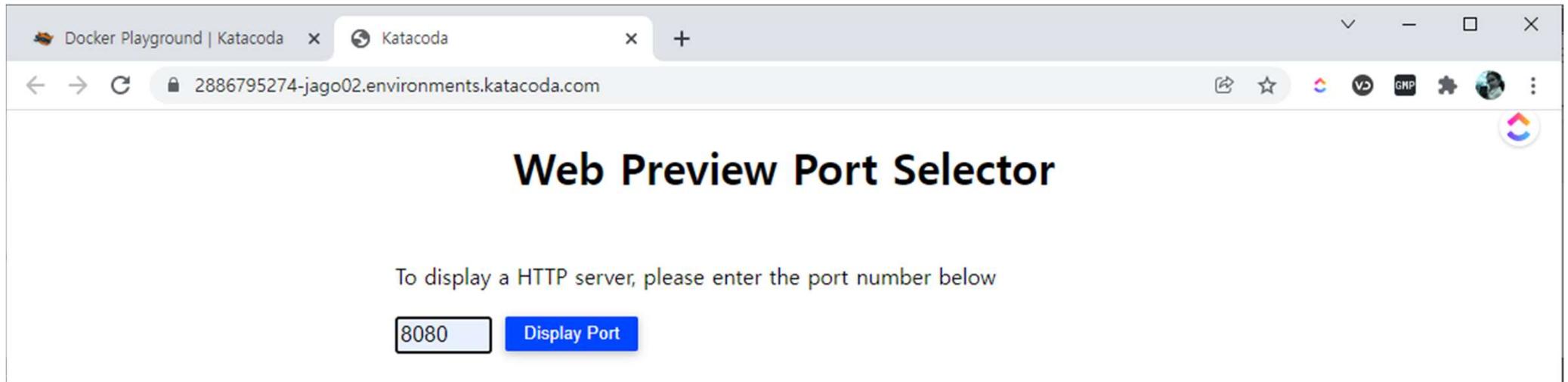
...

그런데, 이렇게 띄운 웹서버를 어떻게 확인해야 할까?



Katacoda - Browsing

8080 port로 실행을 했으니, 아래와 같이 적어주고 "Display Port" 클릭 !!!



///

Breaktime

///

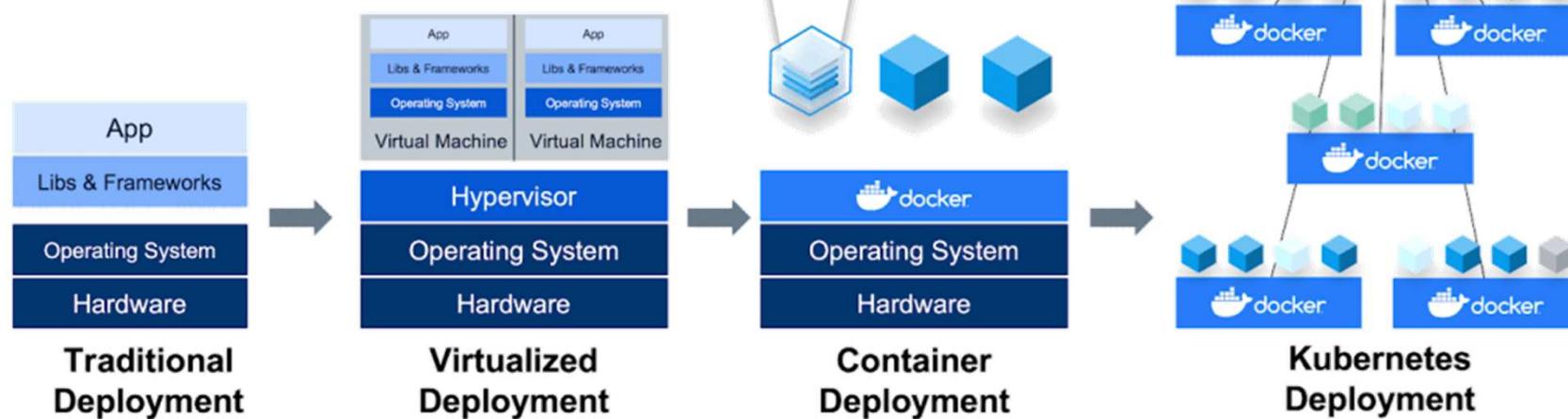
Kubernetes Overview

Kubernetes is ...

**Kubernetes, also known as K8s,
is an open-source system for
automating deployment, scaling, and management
of containerized applications.**

Docker & Kubernetes

Kubernetes & Docker work together to build & run containerized applications



※ 참고 : <https://www.docker.com/blog/top-questions-docker-kubernetes-competitors-or-together/>

Features of Kubernetes



You Must Know About

1 Automatic Binpacking

2 Service Discovery & Load Balancing

3 Storage Orchestration

4 Self Healing

6 Batch Execution

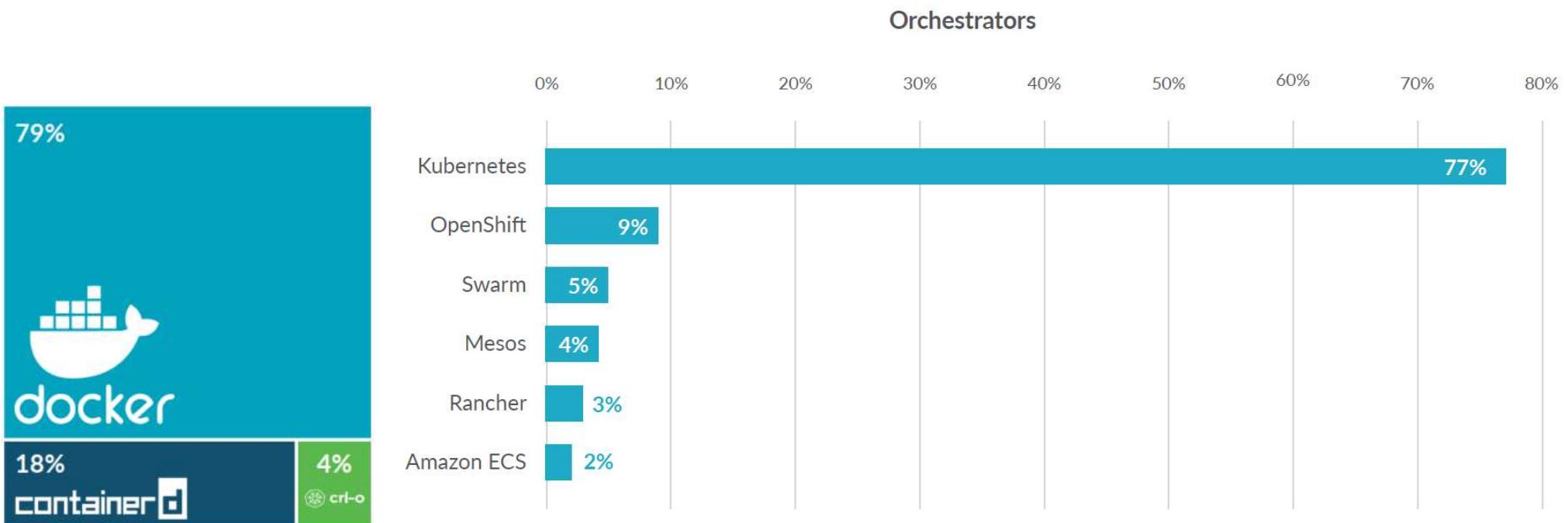
7 Horizontal Scaling

5 Secret & Configuration Management

8 Automatic Rollbacks & Rollouts

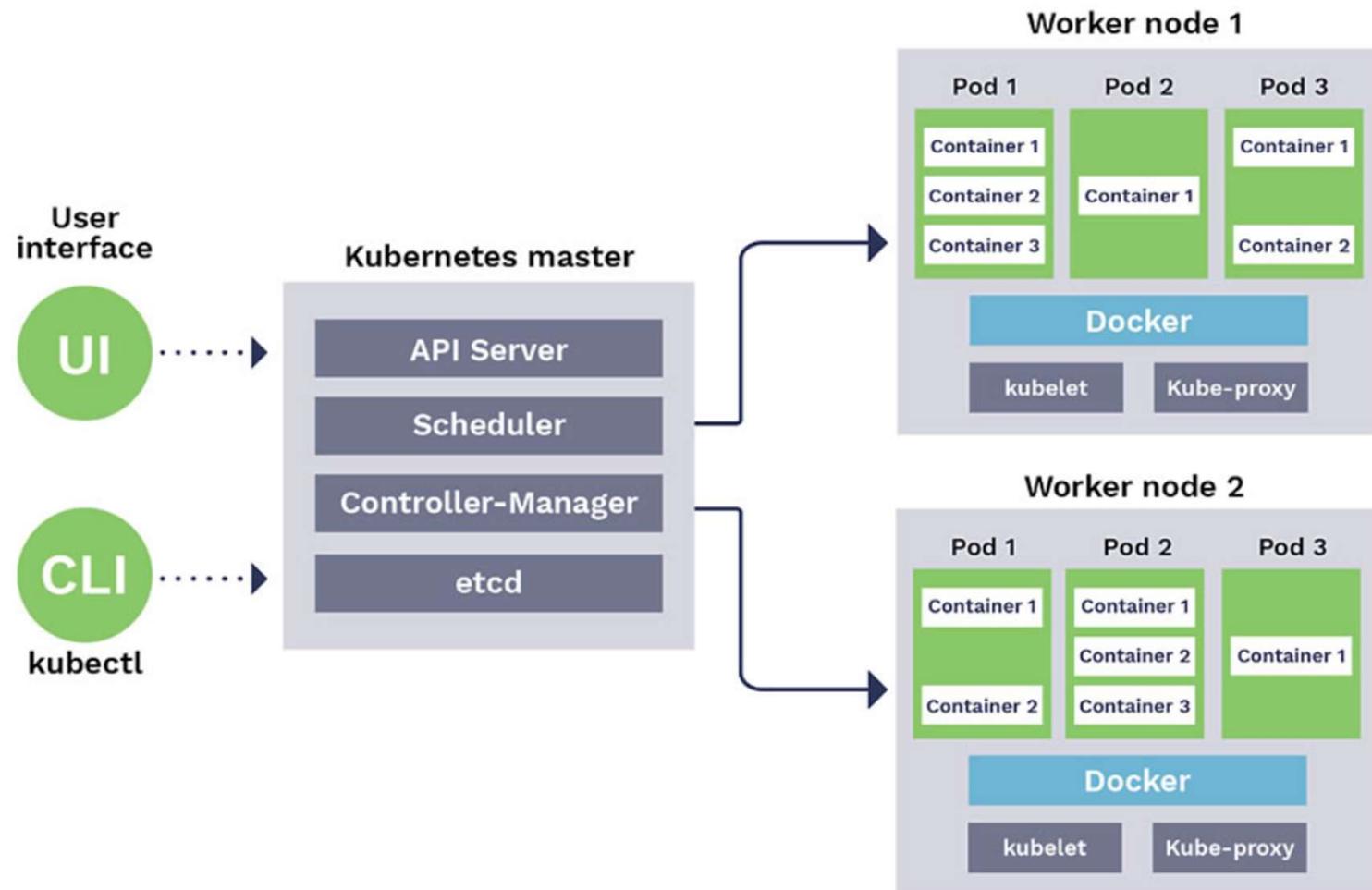
※ 참고 : <https://www.xenonstack.com/blog/debug-application-running-in-kubernetes/>

Market Share



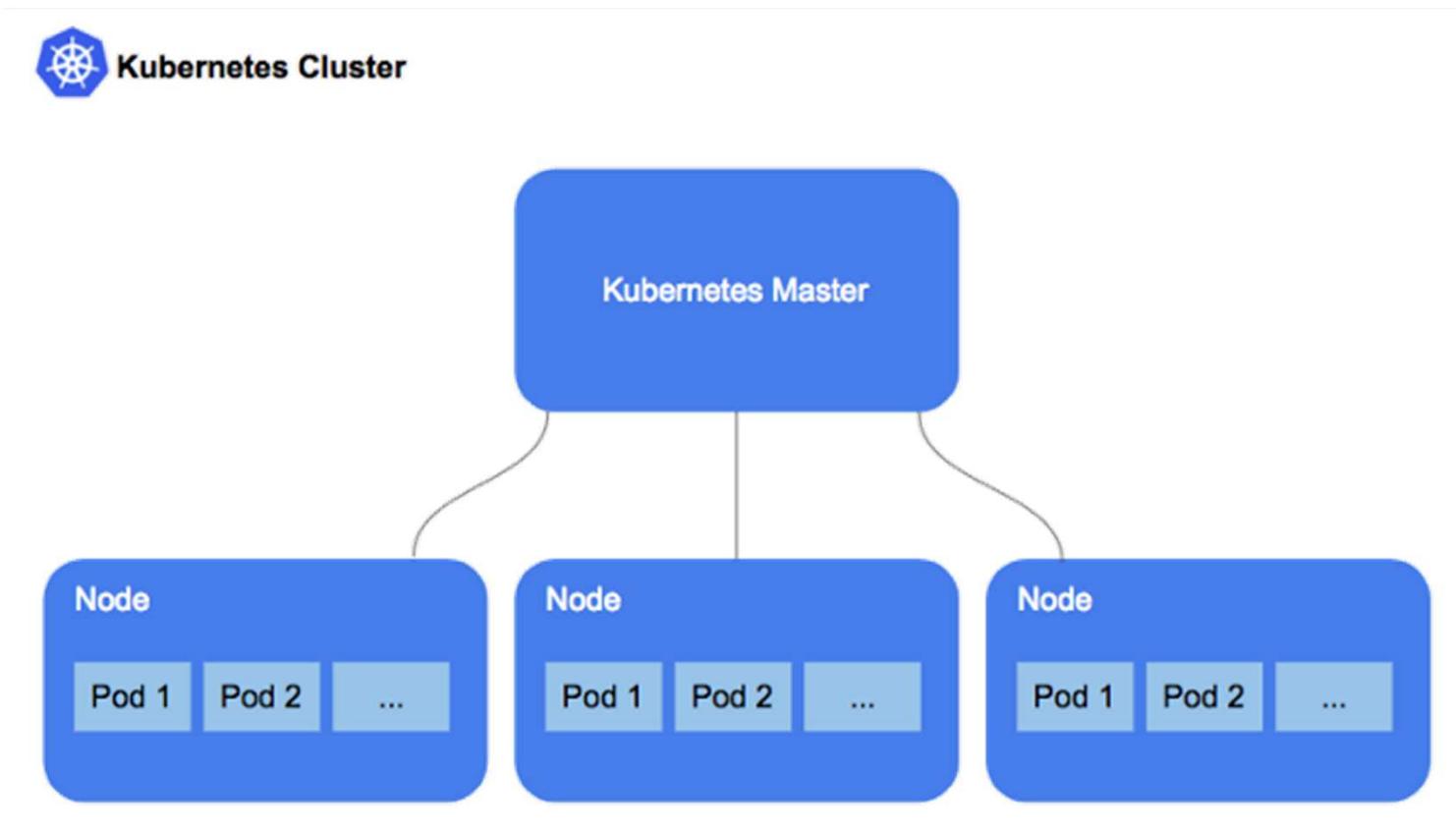
※ 참고 : <https://sysdig.com/blog/sysdig-2019-container-usage-report/>

Kubernetes Architecture



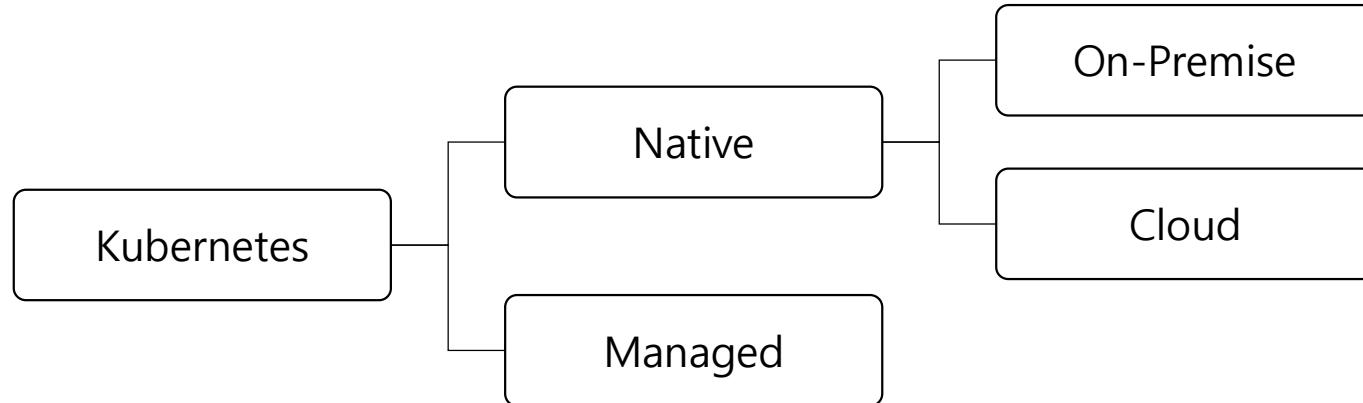
※ 참고 : <https://keetmalin.wixsite.com/keetmalin/post/understanding-container-orchestration-with-kubernetes>

Cluster



※ 참고 : <https://medium.com/@tomerf/so-you-want-to-configure-the-perfect-db-cluster-inside-a-kubernetes-cluster-a4d2c26aca7a>

How ...



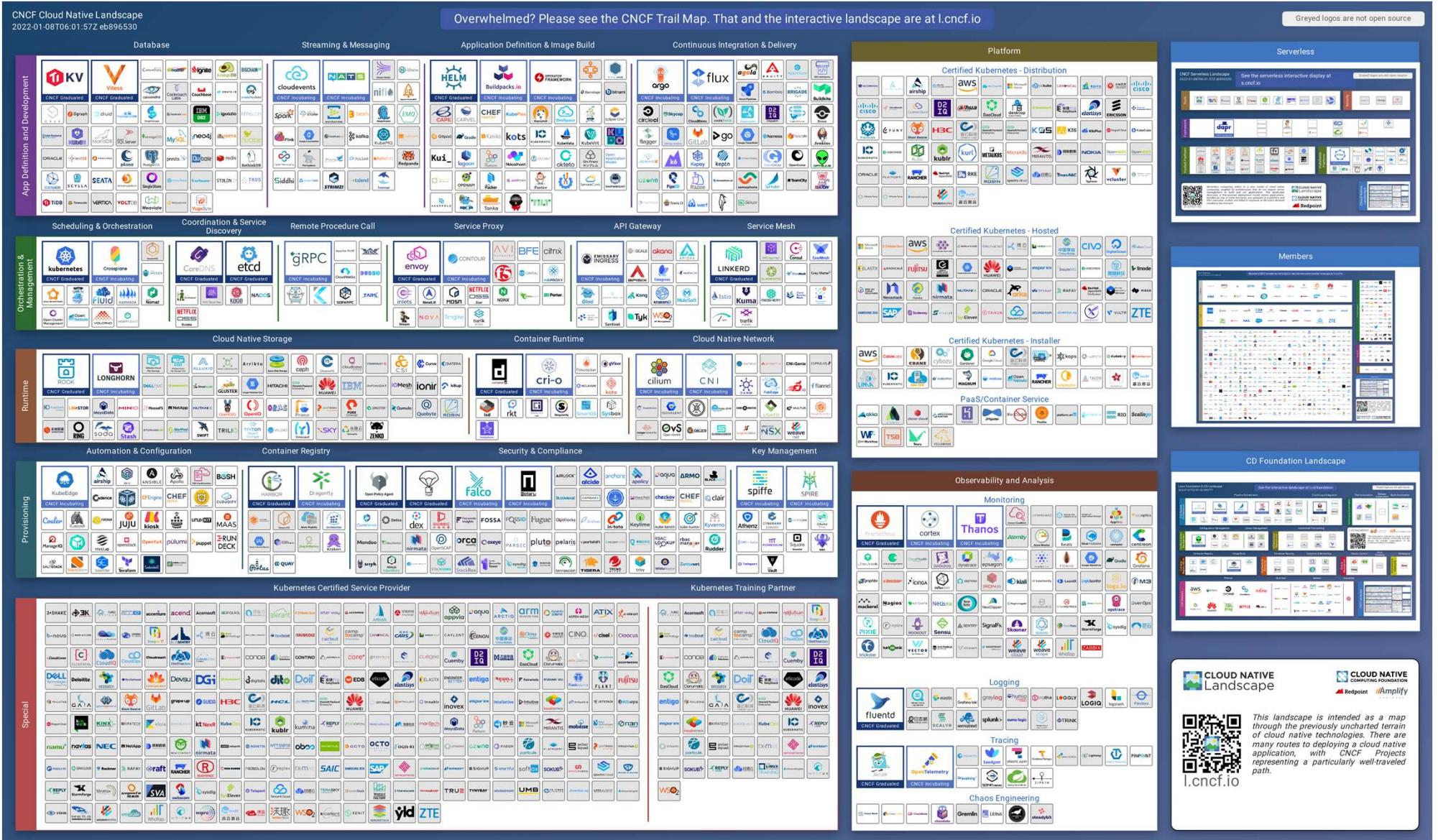
| 항목 | 자체 서버환경 | Cloud 환경 | Managed K8s |
|----------------|---------|----------|-------------|
| 인프라 관리 | 사용자 | 벤더사 | 벤더사 |
| K8s 설치, 관리 | 사용자 | 사용자 | 벤더사 |
| K8s(클러스터) 백업 | 사용자 | 사용자 | 벤더사 |
| K8s(클러스터) 스케일링 | 사용자 | 사용자 | 벤더사 |
| 워커노드 프로비저닝 | 사용자 | 사용자 | 벤더사 |
| 애플리케이션 스케일링 | 사용자 | 사용자 | 사용자 |
| 애플리케이션 배포 | 사용자 | 사용자 | 사용자 |

| | |
|-------|----------------------------------|
| AWS | EKS (Elastic Kubernetes Service) |
| Azure | AKS (Azure Kubernetes Service) |
| GCP | GKE (Google Kubernetes Service) |
| NCP | NKS (Naver Kubernetes Service) |

※ 참고 : https://www.cloocus.com/insight-kubernetes_2/

///

Tip #3 - CNCF



※ 참고 : <https://landscape.cncf.io/images/landscape.png>

///

<https://kahoot.it/>

자습 (복습)

- ▷ Docker
- ▷ VirtualBox