

Managing Kubernetes

2021-04-23

written by whatwant

Agenda

Chapter1. Kubernetes Overview

1주차: **Docker and Kubernetes**

Chapter2. Kubernetes Core

2주차: **Environment & POD**

3주차: **Replication and other controllers**

4주차: **Services & Volumes**

5주차: **ConfigMaps and Secrets & Kubernetes REST API**

6주차: **Deployment & StatefulSet**

7주차: **Summary & HandsOn**

Chapter3. Kubernetes Managing

8주차: **Authentication and User Management & Authorization & Admission Control**

9주차: **Networking**

10주차: **Monitoring**

11주차: **Disaster Recovery**

※ 참고 : <https://home.modulabs.co.kr/product/managing-kubernetes/>

1 week

Docker and Kubernetes

Docker

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 Virtuozzo) - 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

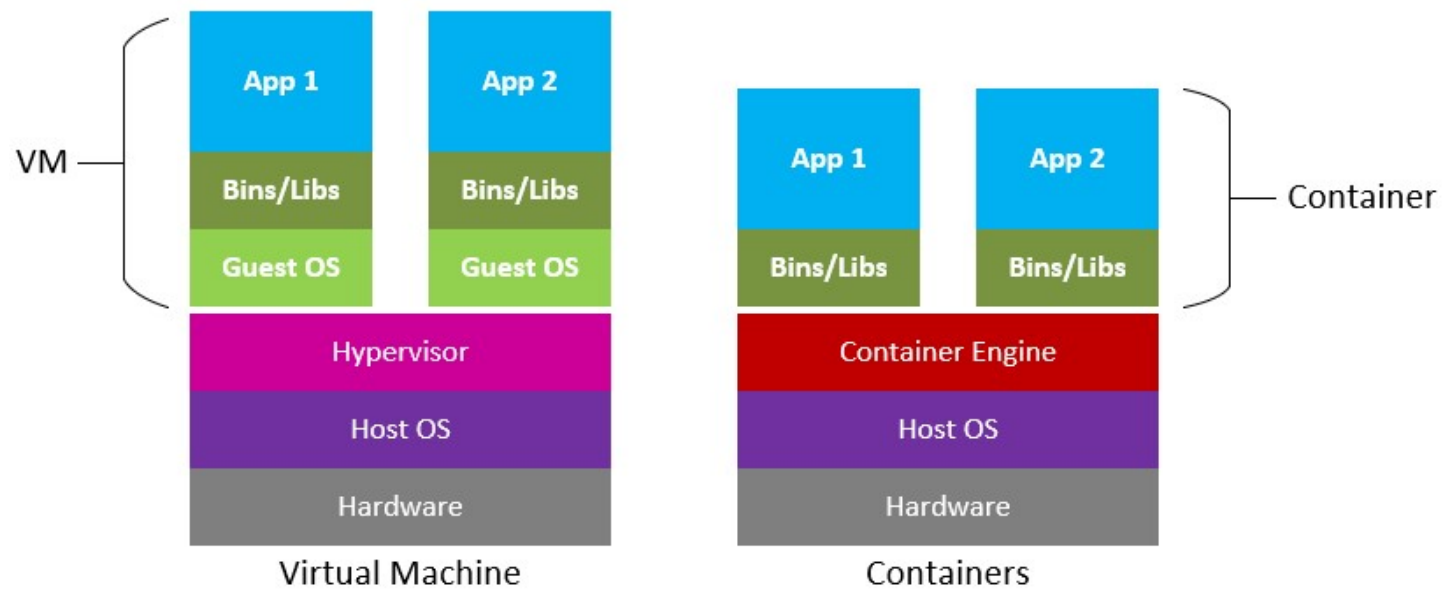
2017: containerd to CNCF (Docker)

2018: The Gold Standard – 시장 표준

2019: A Shifting Landscape - 변화

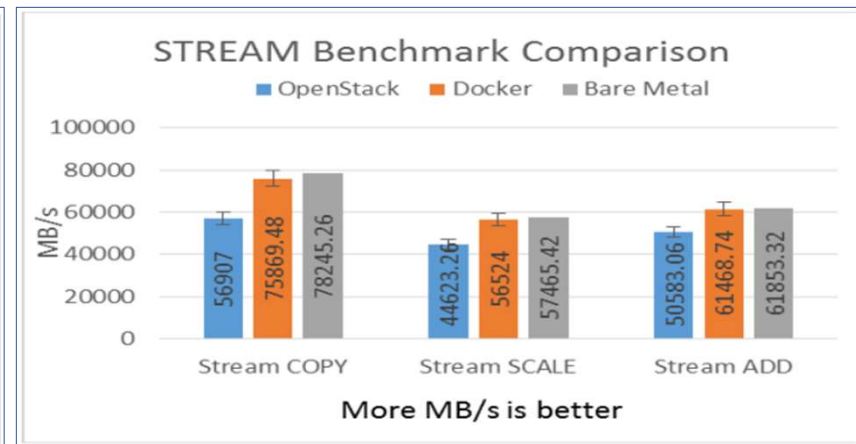
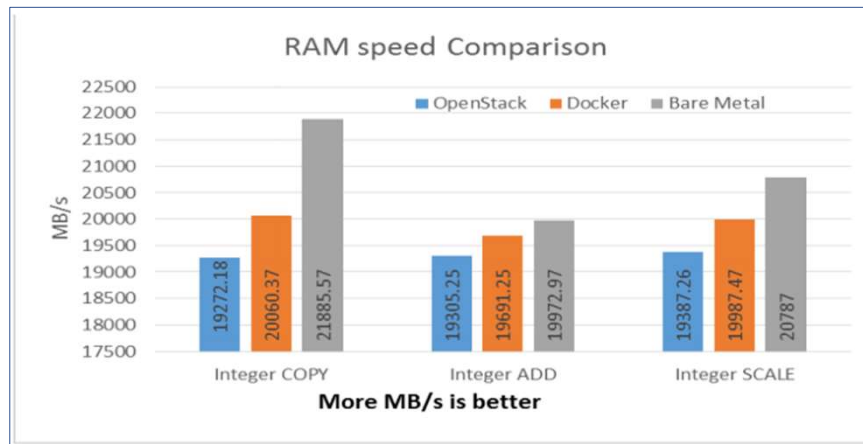
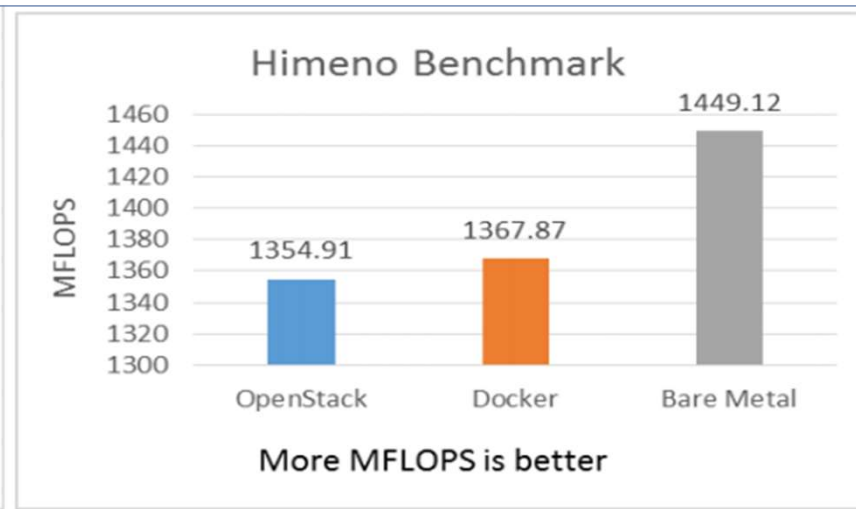
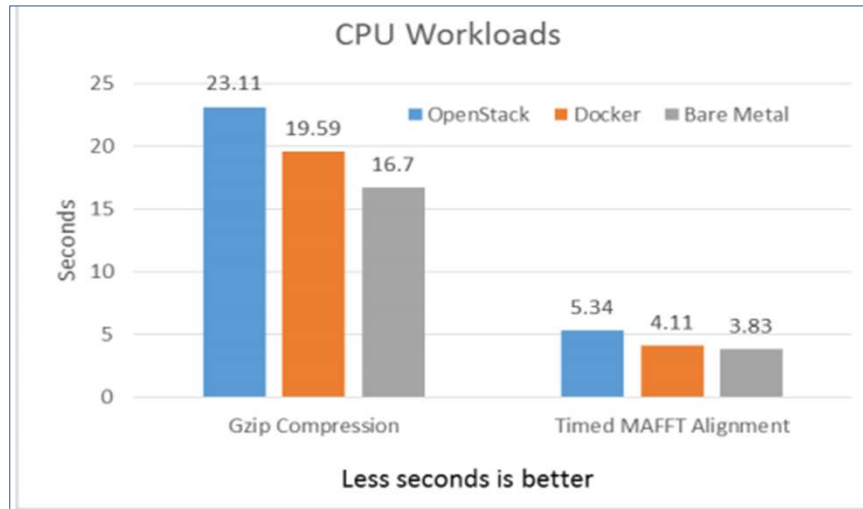
※ 참고 : <https://blog.aquasec.com/a-brief-history-of-containers-from-1970s-chroot-to-docker-2016>

Virtual Machine vs. Containers



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

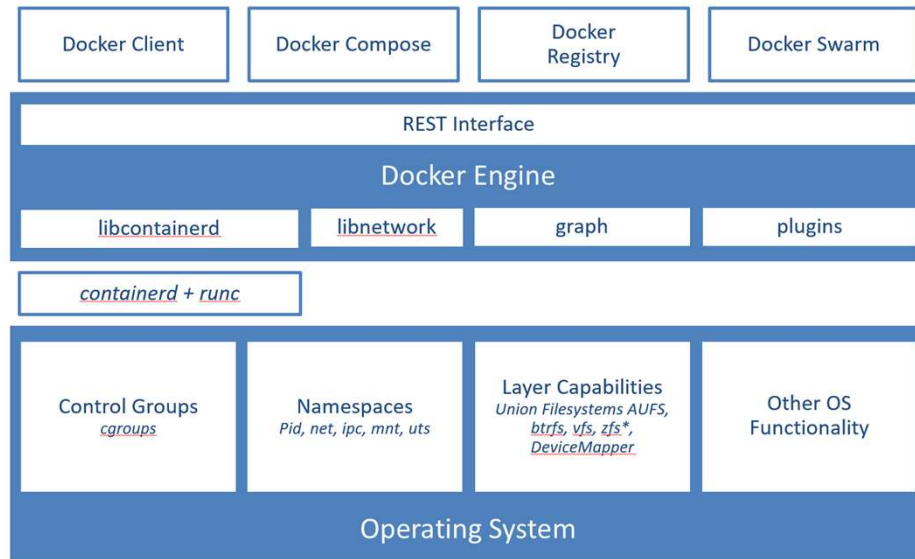
Docker Container vs. Openstack Virtual Machine vs. Bare Metal Server



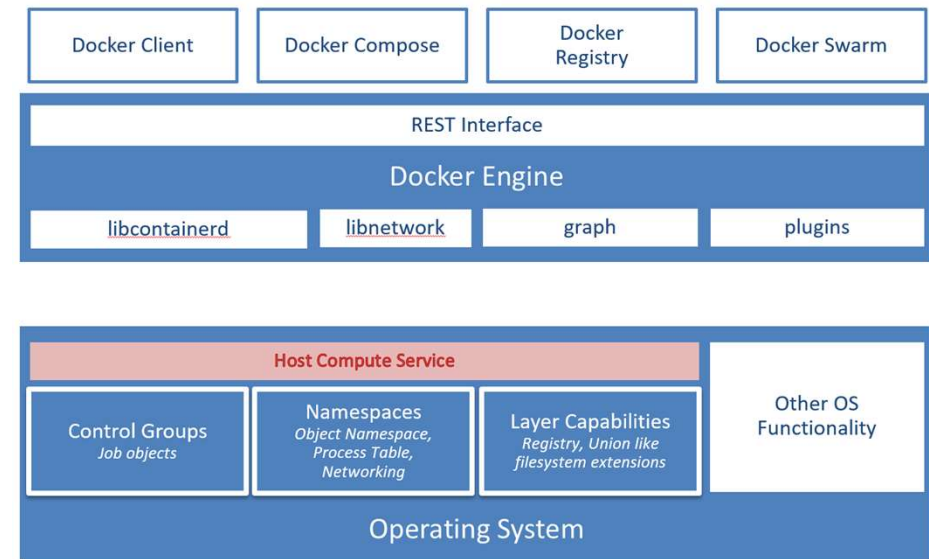
※ 참고 : <http://ijeecs.iaescore.com/index.php/IJECS/article/view/7925>

Docker Container Architecture

Architecture In Linux



Architecture In Windows



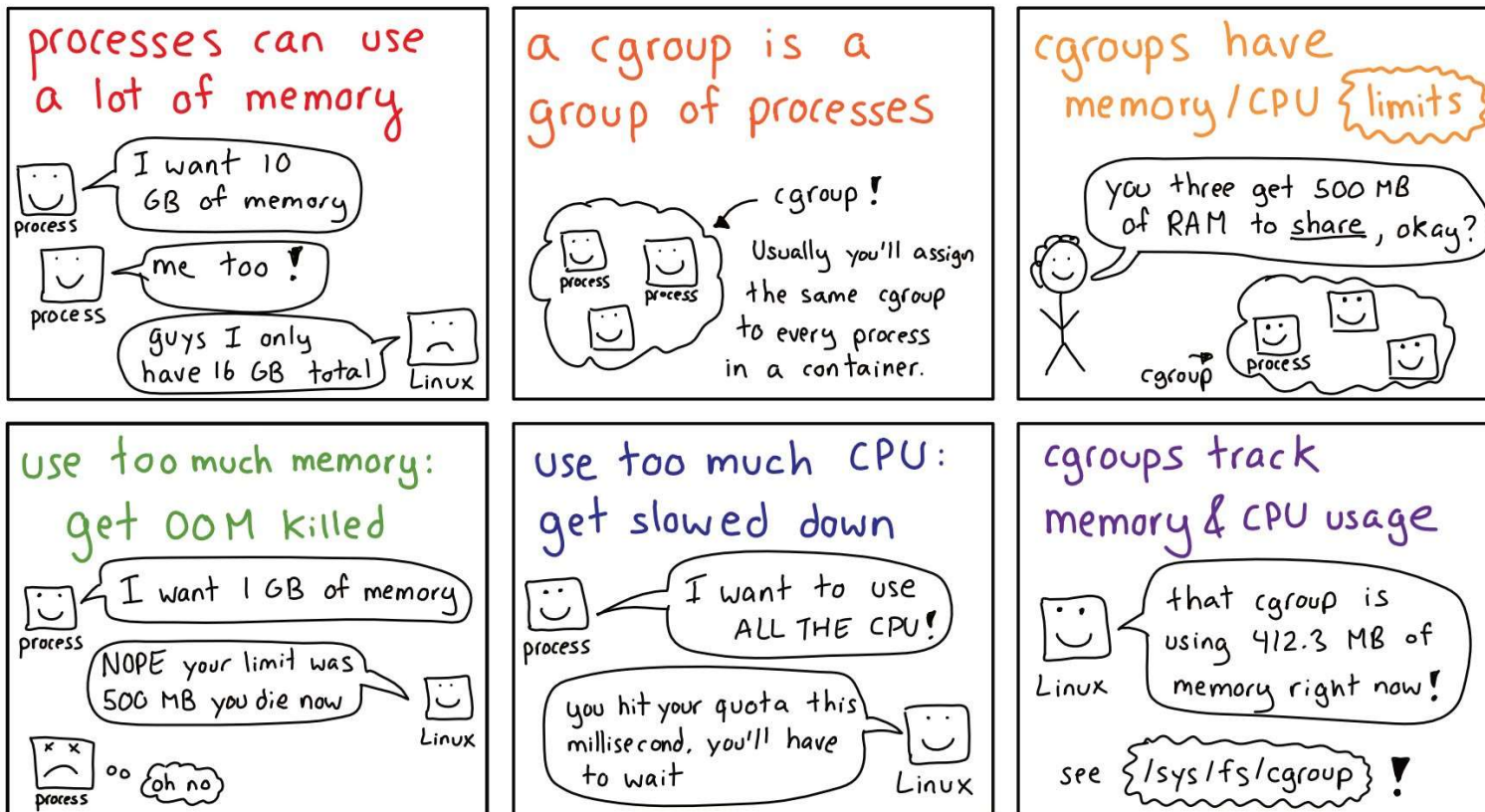
▶ HCS (Host Compute Service) : Hyper-V의 저수준 컨테이너 관리 API

※ 참고 : <https://docs.microsoft.com/ko-kr/virtualization/windowscontainers/deploy-containers/containerd>

cgroup

JULIA EVANS
@b0rk

cgroups



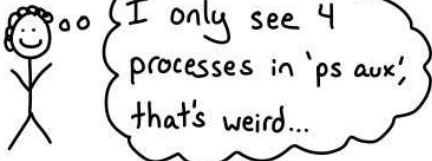
※ 참고 : <https://twitter.com/b0rk/status/1214341831049252870>

namespace

JULIA EVANS
@b0rk

namespaces

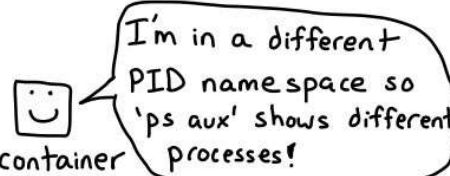
inside a container, things look different



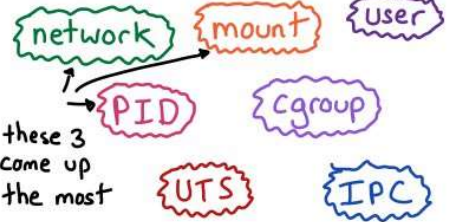
Commands that will look different

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

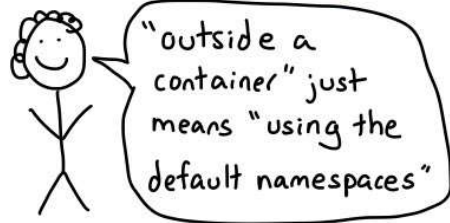
Why those commands look different:
≡ namespaces ≡



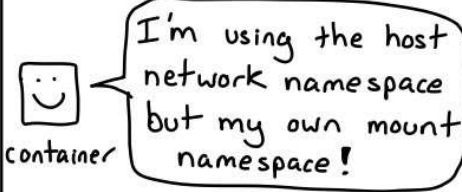
every process has 7 kinds of namespaces



there's a default ("host") namespace

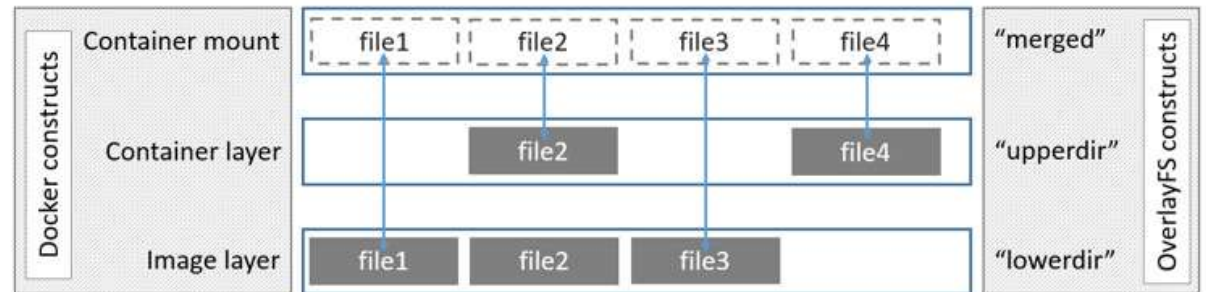
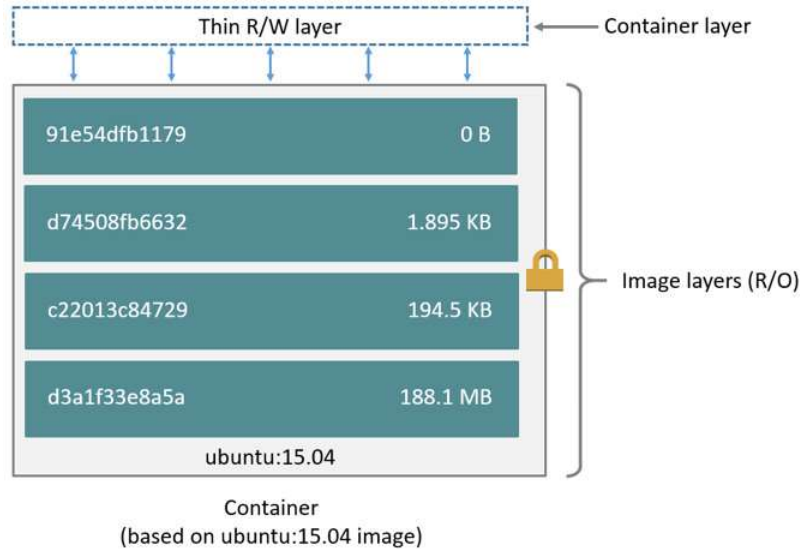


processes can have any combination of namespaces



♥ this? more at wizardzines.com

Docker Storage Driver



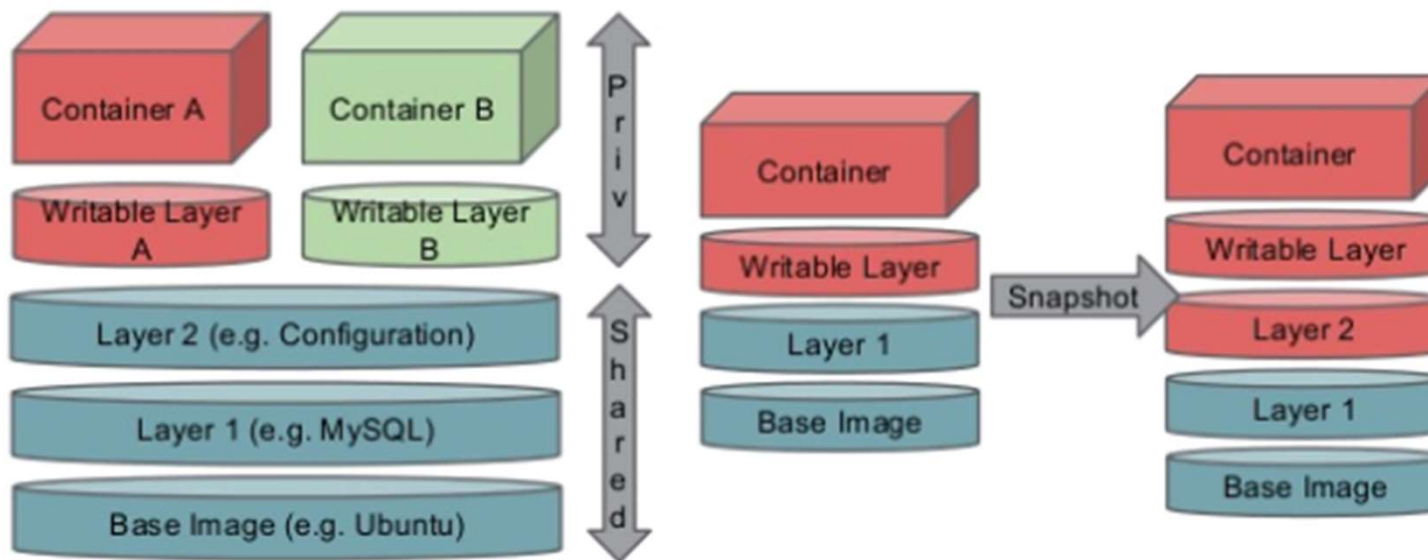
How the overlay driver works

[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

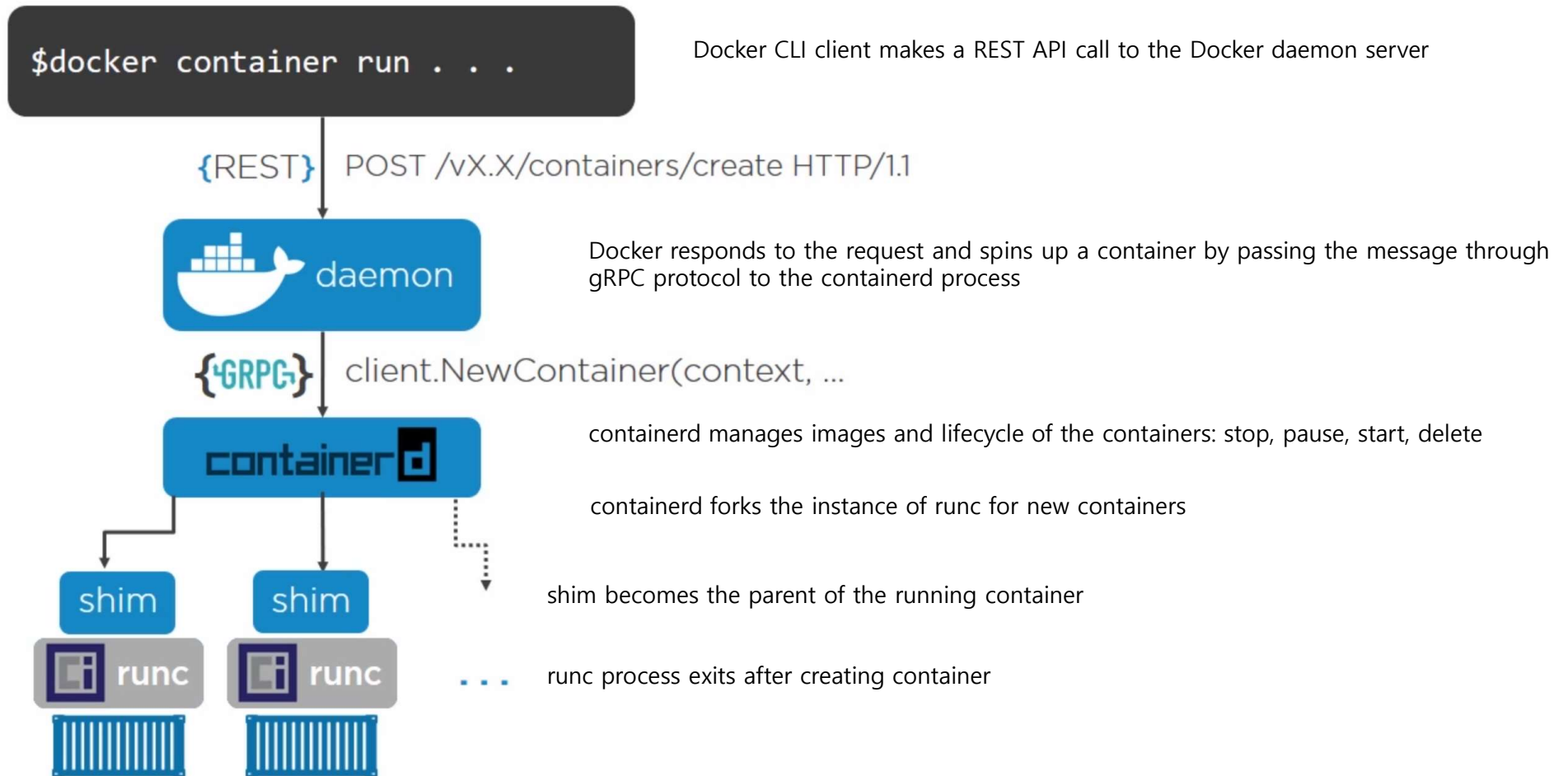
※ 참고 : <https://docs.docker.com/storage/storagedriver/overlayfs-driver/>

Docker Storage Driver



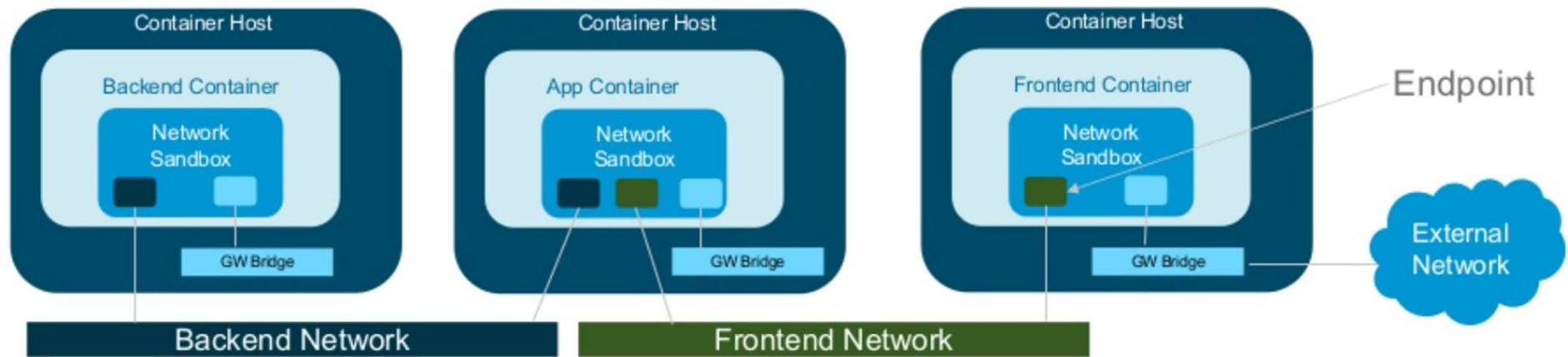
※ 참고 : <https://docs.docker.com/storage/storagedriver/overlayfs-driver/>

Docker engine architecture



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

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

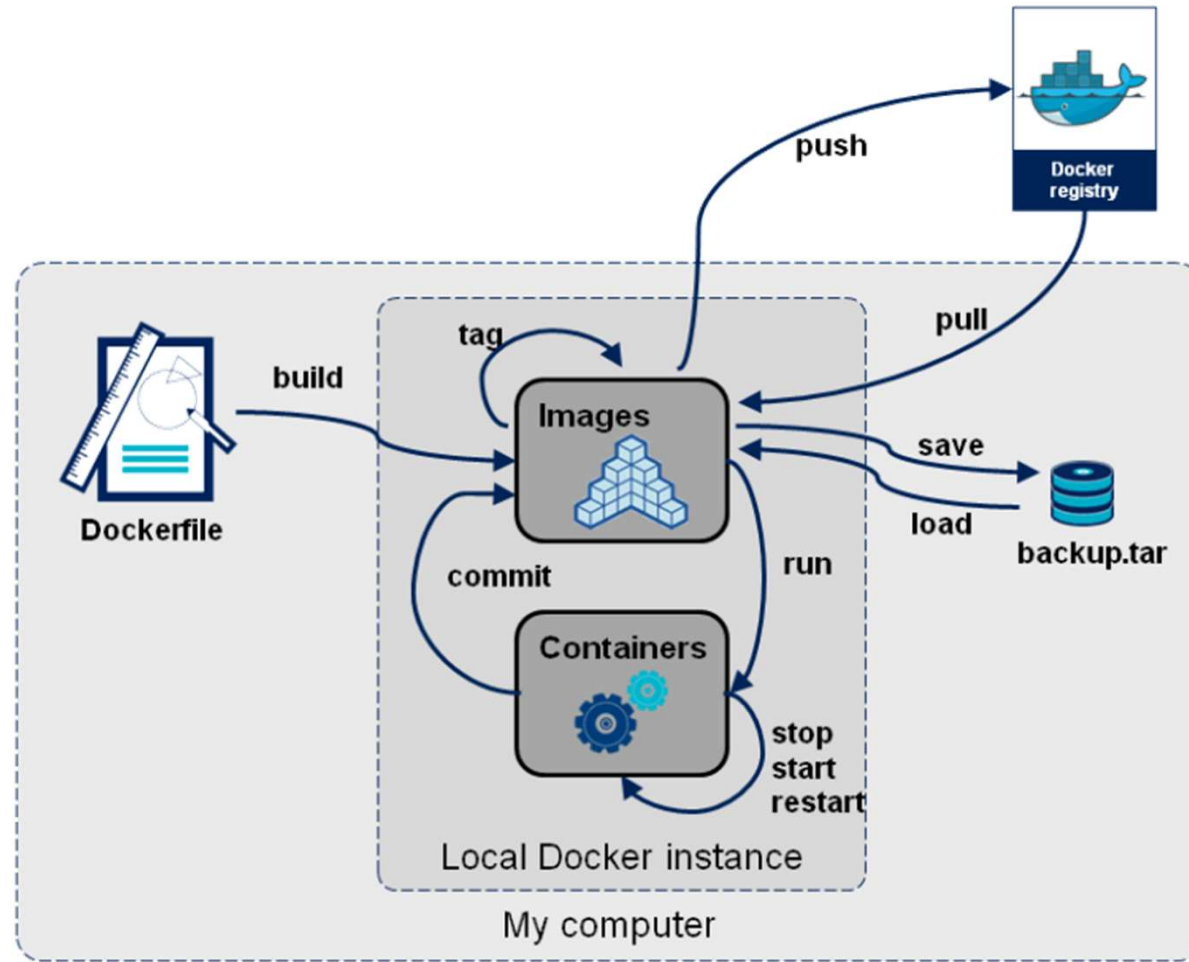
- An Endpoint joins a Sandbox to a Network.
- An implementation of an Endpoint could be a veth pair, an Open vSwitch internal port or similar

► 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>

Dockerfile

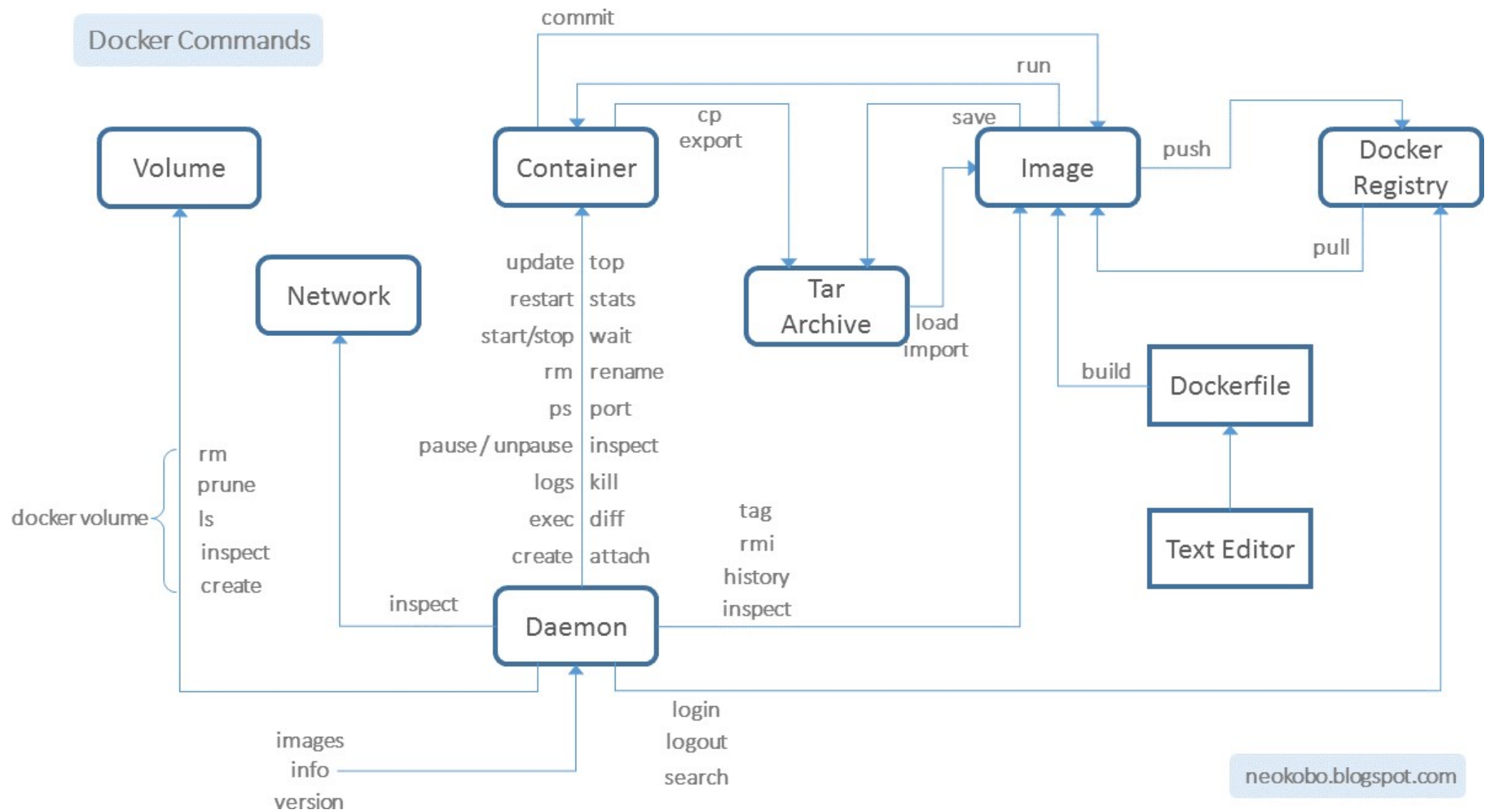


※ 참고 : <https://blog.wonizz.tk/2019/07/31/docker-dockerfile/>

Dockerfile - sample

```
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/>



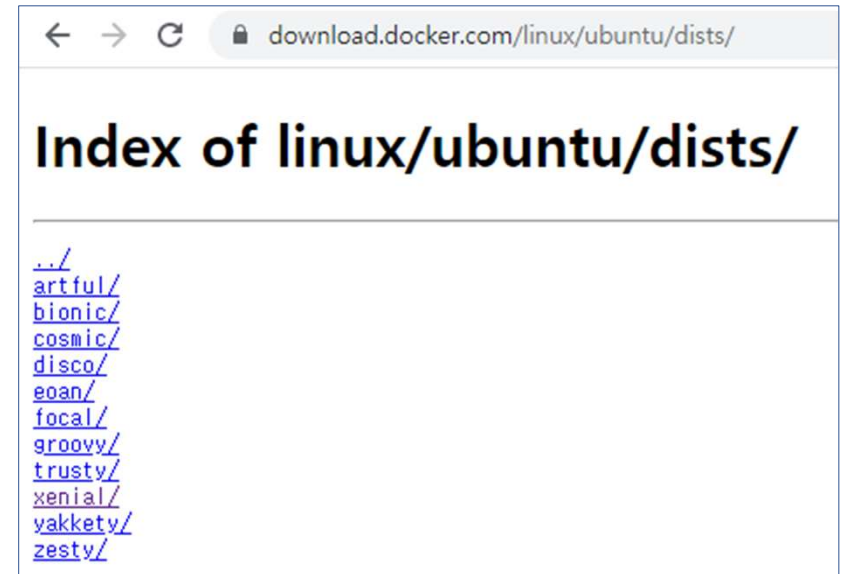
※ 참고 : <http://neokobo.blogspot.com/2017/12/docker-command-flowchart.html>

Docker

실습

Install

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




```
$ wget https://download.docker.com/linux/ubuntu/dists/focal/pool/stable/amd64/containerd.io_1.4.3-1_amd64.deb
$ wget https://download.docker.com/linux/ubuntu/dists/focal/pool/stable/amd64/docker-ce-cli_20.10.1~3-0~ubuntu-focal_amd64.deb
$ wget https://download.docker.com/linux/ubuntu/dists/focal/pool/stable/amd64/docker-ce_20.10.1~3-0~ubuntu-focal_amd64.deb
```

```
$ sudo dpkg --install ./containerd.io_1.4.3-1_amd64.deb
$ sudo dpkg --install ./docker-ce-cli_20.10.1~3-0~ubuntu-focal_amd64.deb
$ sudo dpkg --install ./docker-ce_20.10.1~3-0~ubuntu-focal_amd64.deb
```

```
$ sudo usermod -aG docker $USER
```

Katacoda

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

 KATACODA OVERVIEW & SOLUTIONS

CLAIM YOUR PROFILE [LOG OUT >](#)

Docker Playground

Helpful Links

Start using Docker with `docker`.

Launch containers with `docker run redis`. Use `CTRL + C` to stop the running container.

View port at <https://2886795276-cykoria04.environments.katacoda.com>

Interested in writing your own Docker scenarios and demos? Visit www.katacoda.com/teach

[CONTINUE](#)

Terminal +

```
$ docker images
```

REPOSITORY	TAG	IMAGE ID	CREATED	SIZE
redis	latest	4760dc956b2d	3 years ago	107MB
ubuntu	latest	f975c5035748	3 years ago	112MB
alpine	latest	3fd9065eaf02	3 years ago	4.14MB

```
$
```

Nginx – 1/2

index.html

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

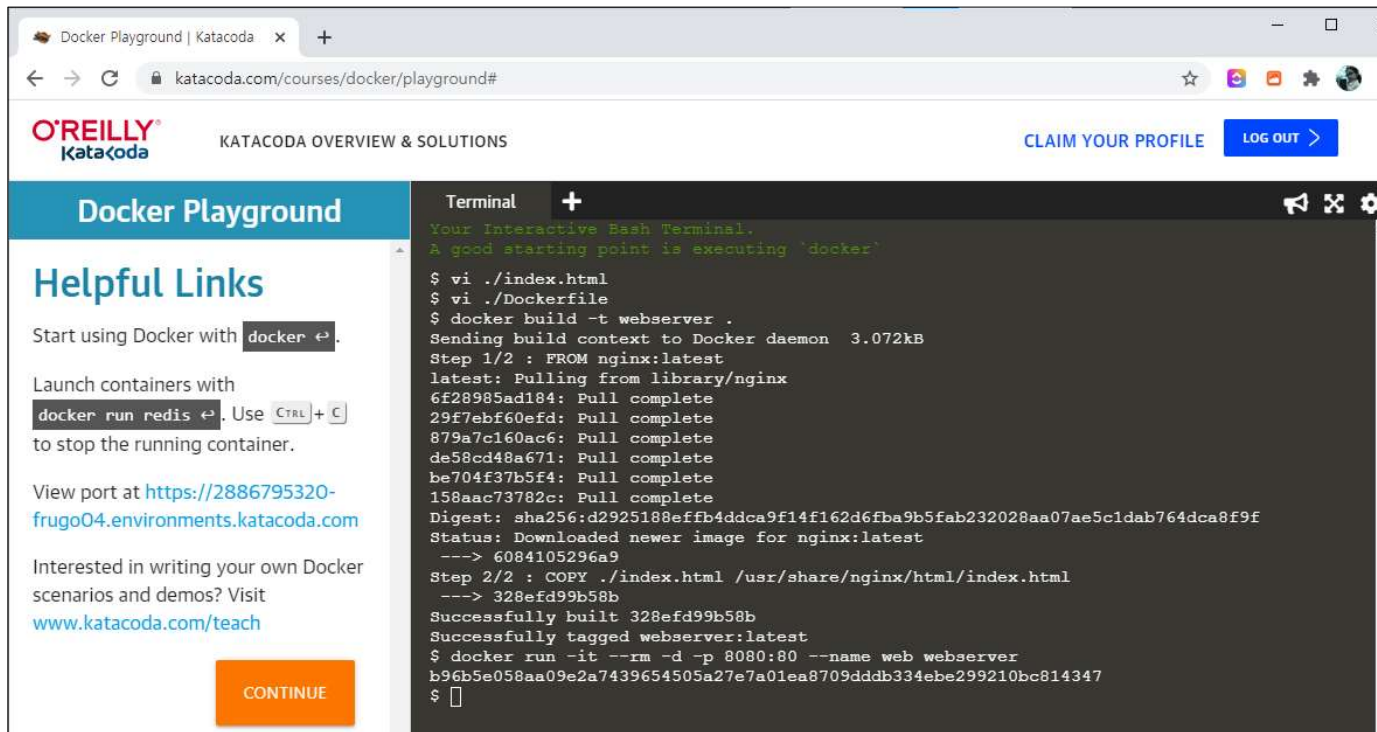
Dockerfile

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

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

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

Nginx – 2/2



Docker Playground | Katacoda

katacoda.com/courses/docker/playground#

O'REILLY Katacoda KATACODA OVERVIEW & SOLUTIONS CLAIM YOUR PROFILE LOG OUT >

Docker Playground

Helpful Links

Start using Docker with `docker`.

Launch containers with `docker run redis`. Use `CTRL + C` to stop the running container.

View port at <https://2886795320-frugo04.environments.katacoda.com>

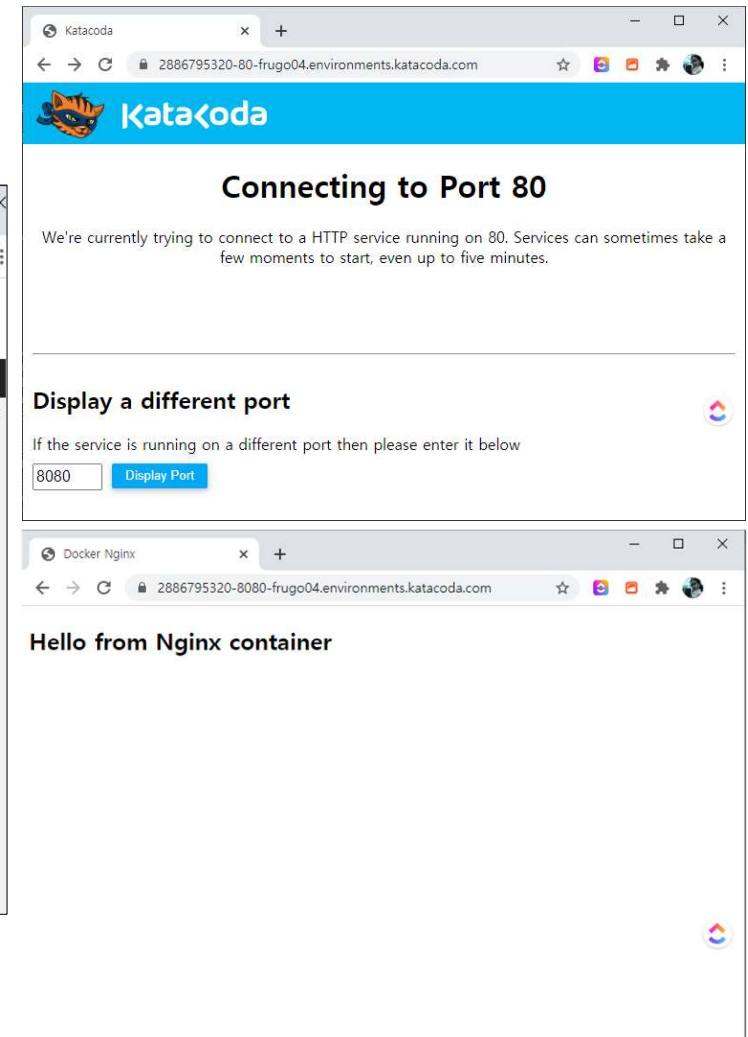
Interested in writing your own Docker scenarios and demos? Visit www.katacoda.com/teach

CONTINUE

Terminal

```
Your Interactive Bash Terminal.
A good starting point is executing 'docker'

$ vi ./index.html
$ vi ./Dockerfile
$ docker build -t webserver .
Sending build context to Docker daemon 3.072kB
Step 1/2 : FROM nginx:latest
latest: Pulling from library/nginx
6f28985ad184: Pull complete
29f7ebf60efd: Pull complete
879a7c160ac6: Pull complete
de58cd48a671: Pull complete
be704f37b5f4: Pull complete
158aac73782c: Pull complete
Digest: sha256:d2925188effb4ddca9f14f162d6fba9b5fab232028aa07ae5c1dab764dca8f9f
Status: Downloaded newer image for nginx:latest
--> 6084105296a9
Step 2/2 : COPY ./index.html /usr/share/nginx/html/index.html
--> 328efd99b58b
Successfully built 328efd99b58b
Successfully tagged webserver:latest
$ docker run -it --rm -d -p 8080:80 --name web webserver
b96b5e058aa09e2a7439654505a27e7a01ea8709dddb334ebe299210bc814347
$
```



Katacoda

Connecting to Port 80

We're currently trying to connect to a HTTP service running on 80. Services can sometimes take a few moments to start, even up to five minutes.

Display a different port

If the service is running on a different port then please enter it below

8080 Display Port

Docker Nginx

Hello from Nginx container

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

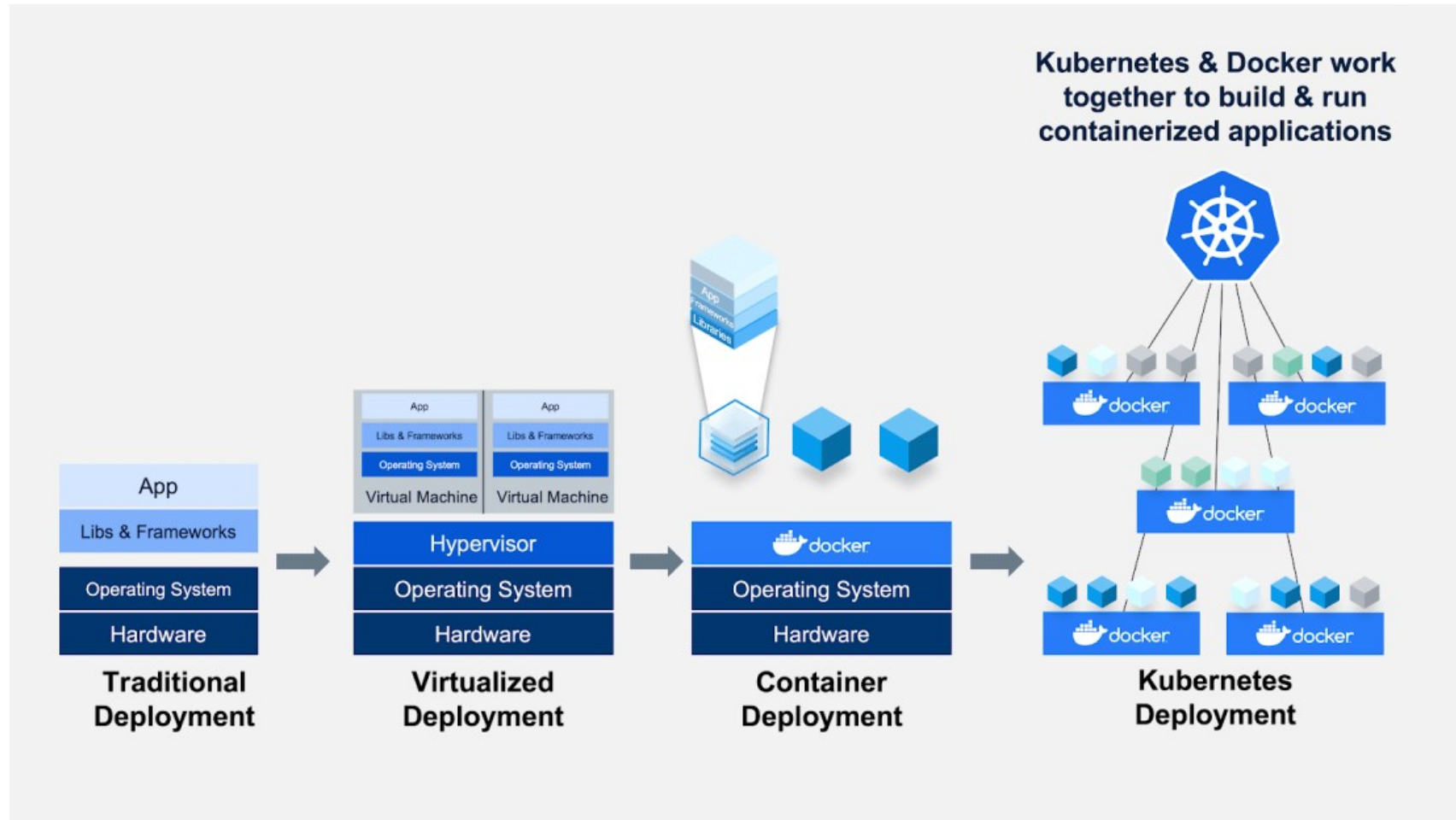
Kubernetes - Overview

Kubernetes is ...

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

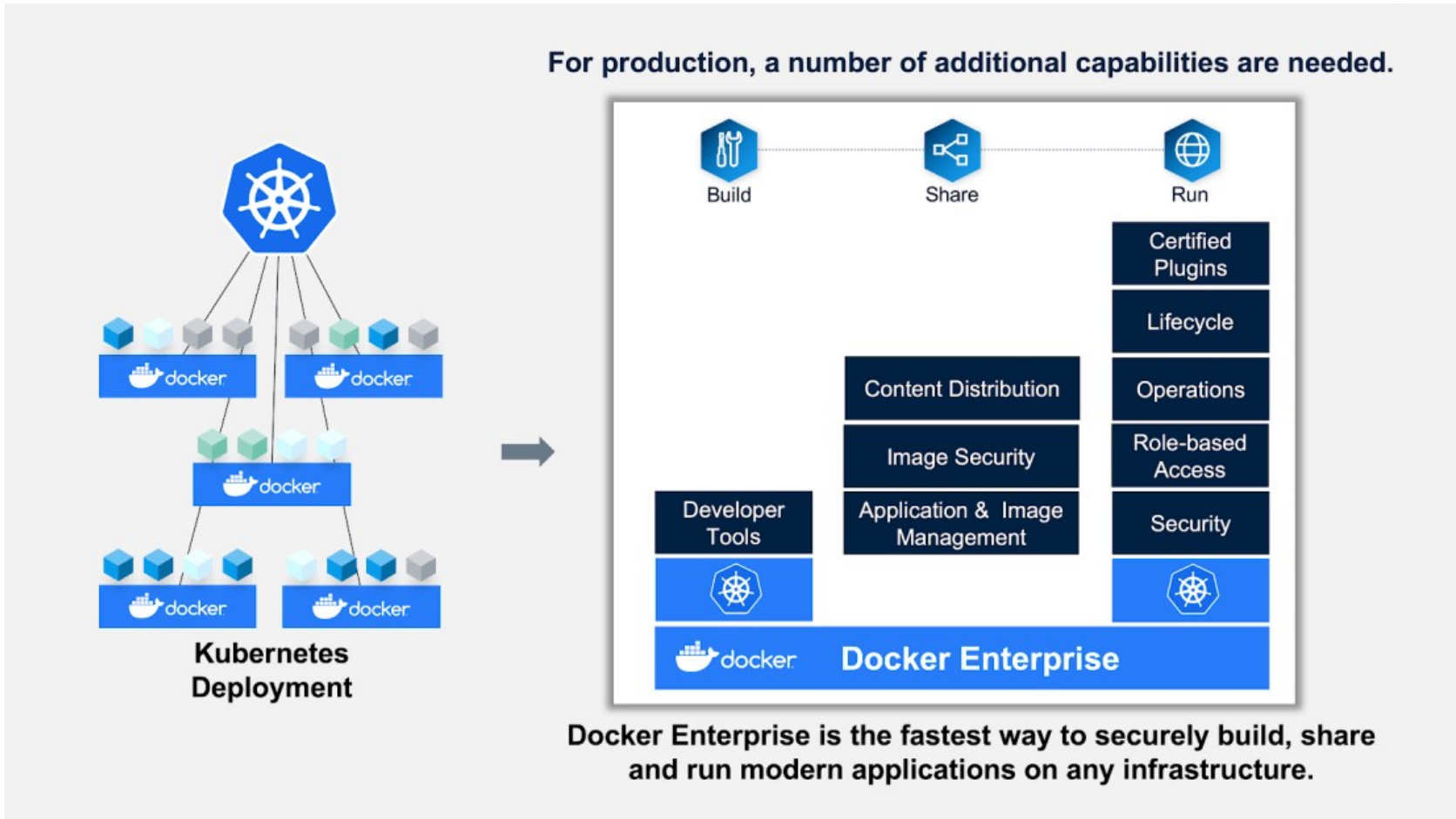
※ 참고 : <https://kubernetes.io/>

Docker & Kubernetes



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

Docker & Kubernetes



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

Features of Kubernetes

Kubernetes Features You Must Know About



1 Automatic
Binpacking

2 Service Discovery
& Load Balancing

3 Storage
Orchestration

4 Self Healing

6 Batch Execution

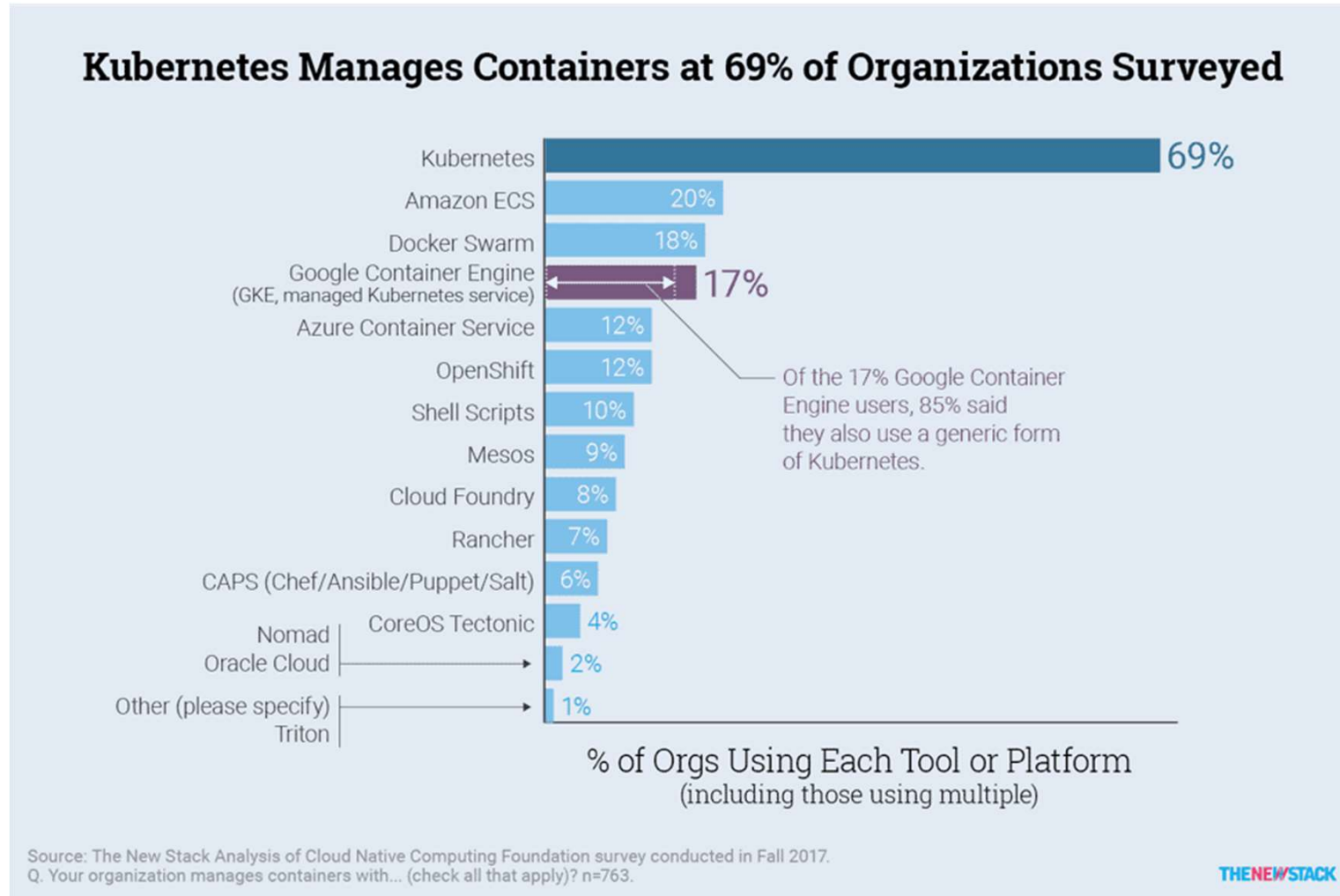
7 Horizontal
Scaling

5 Secret & Configuration
Management

8 Automatic Rollbacks
& Rollouts

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

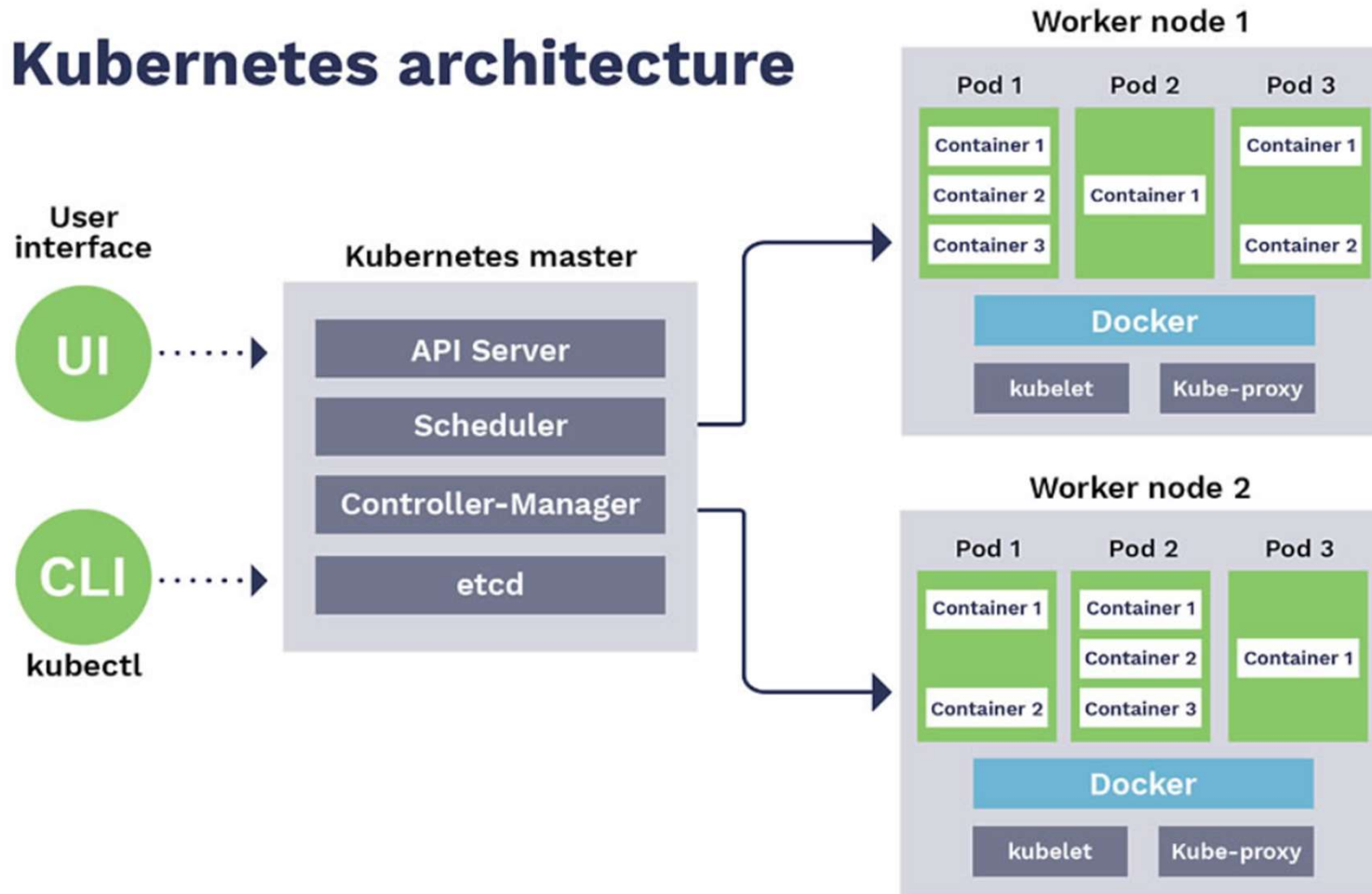
Kubernetes M/S



※ 참고 : <https://www.cncf.io/blog/2018/03/07/cncf-sponsors-new-free-kubernetes-deployment-and-security-patterns-ebook-from-the-new-stack/>

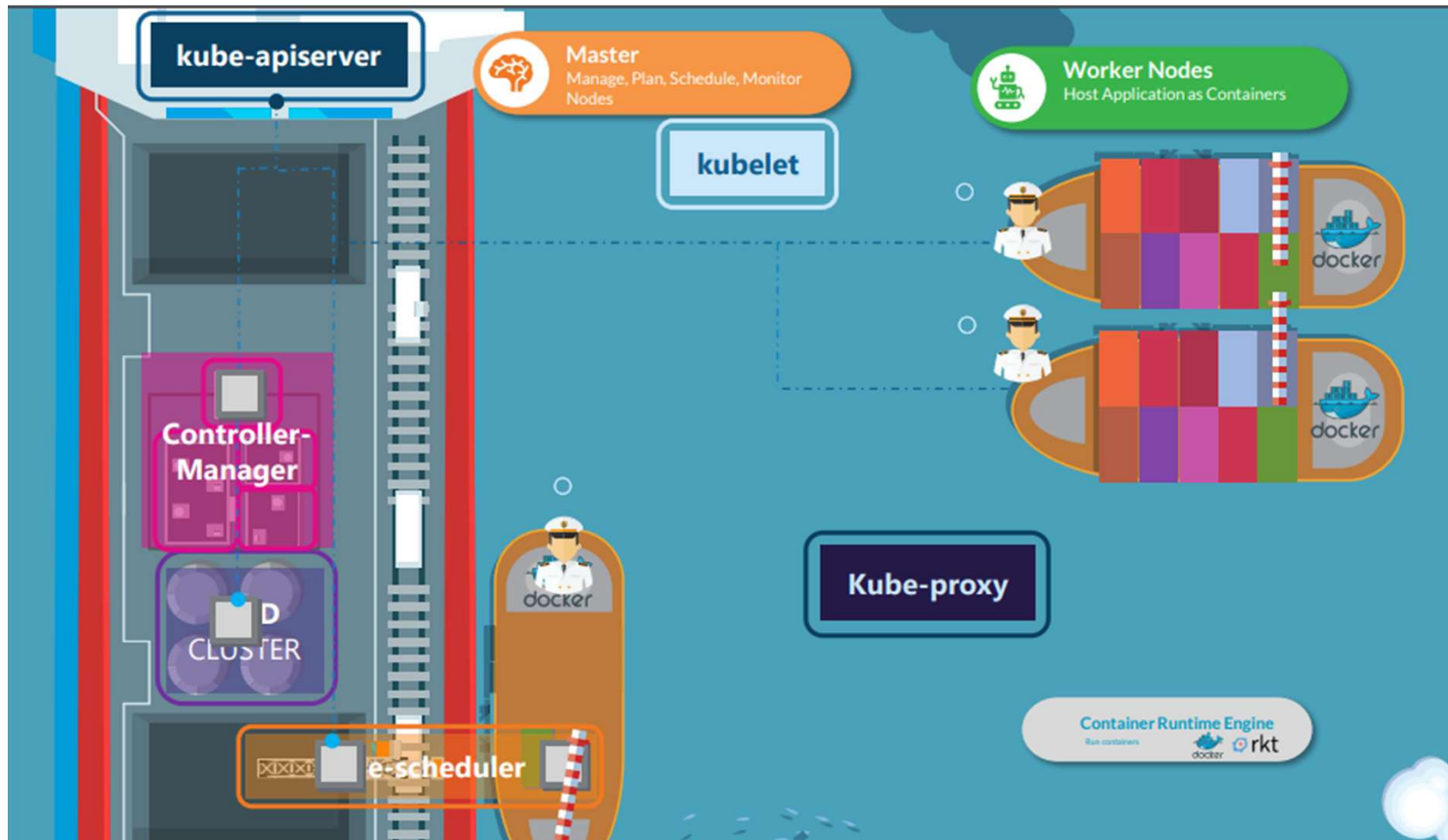
Kubernetes Architecture

Kubernetes architecture



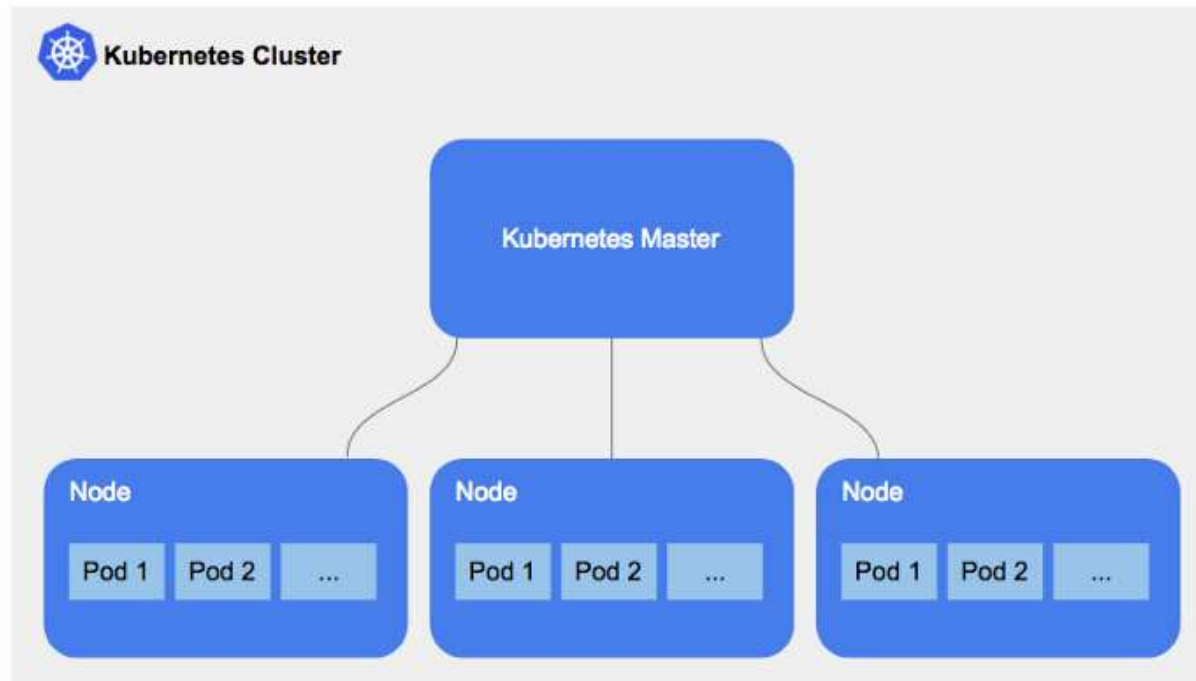
※ 참고 : <https://keetmalin.wixsite.com/keetmalin/post/understanding-container-orchestration-with-kubernetes>

Cluster



※ 참고 : <https://github.com/kodekloudhub/certified-kubernetes-administrator-course/blob/master/docs/02-Core-Concepts/02-Cluster-Architecture.md>

Cluster

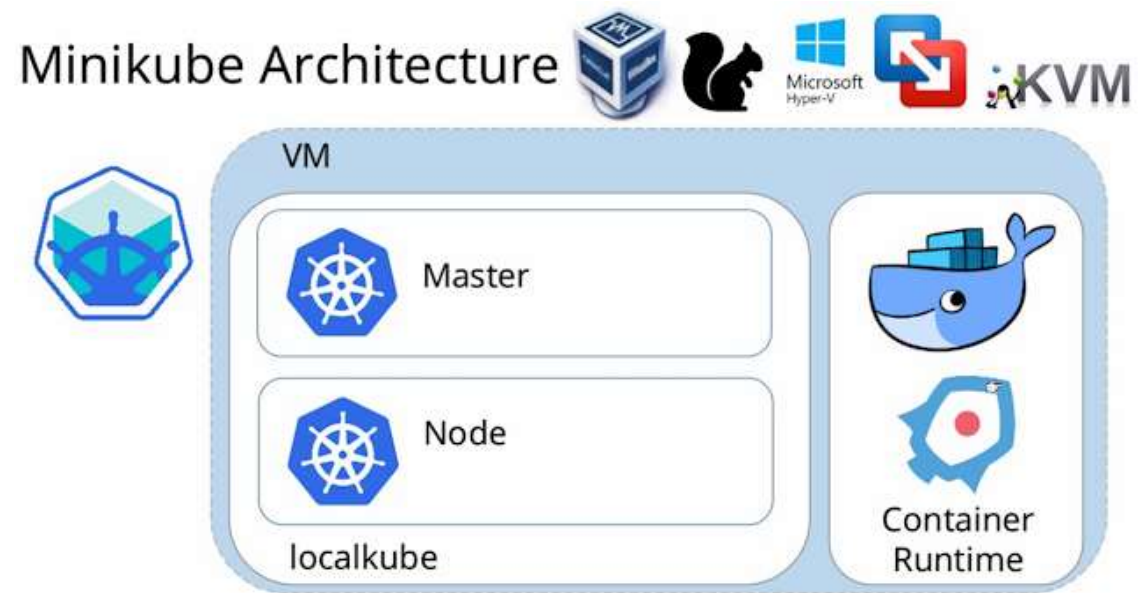


※ 참고 : <https://medium.com/@tomerrf/so-you-want-to-configure-the-perfect-db-cluster-inside-a-kubernetes-cluster-a4d2c26aca7a>

Kubernetes - Overview

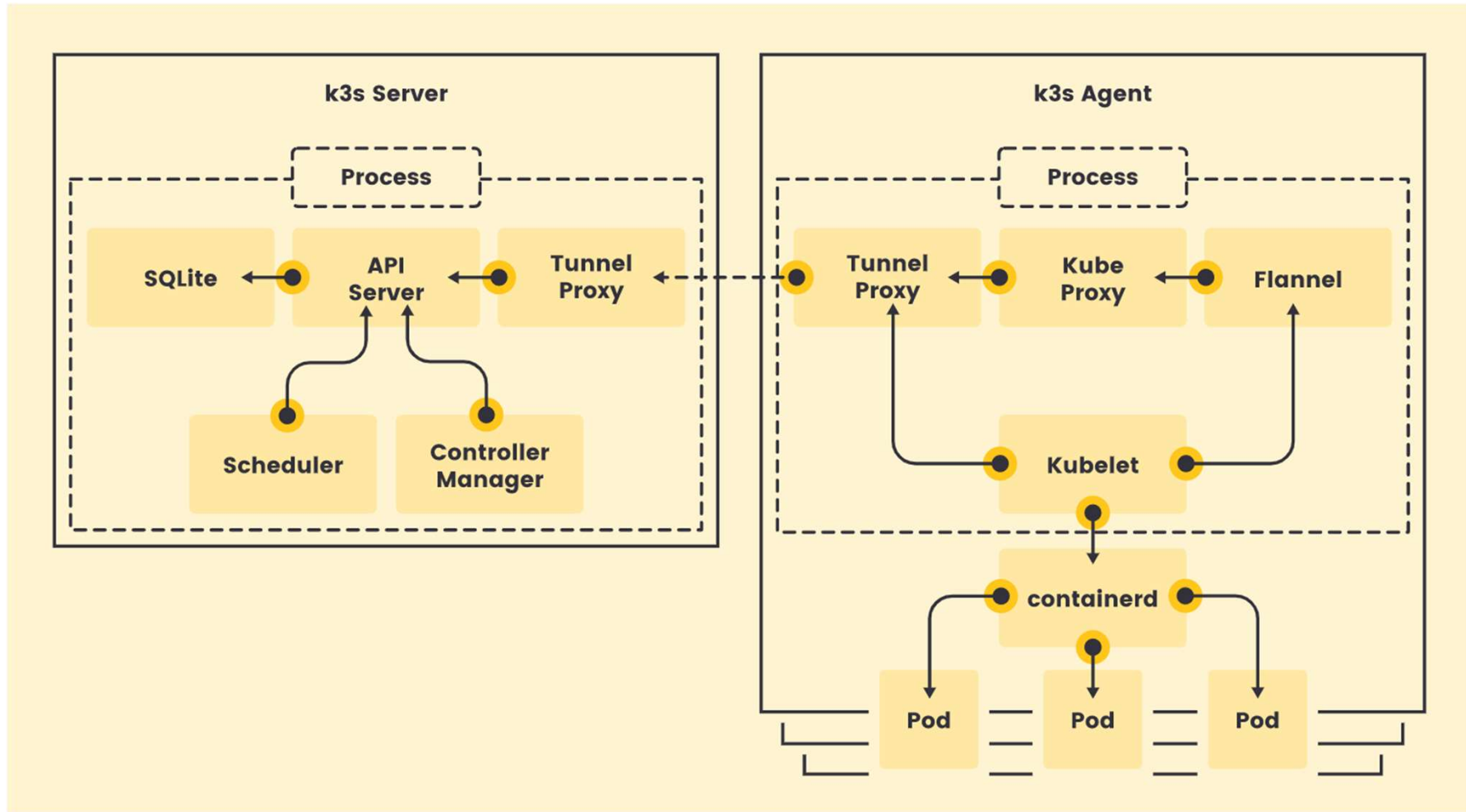
실습

Minikube



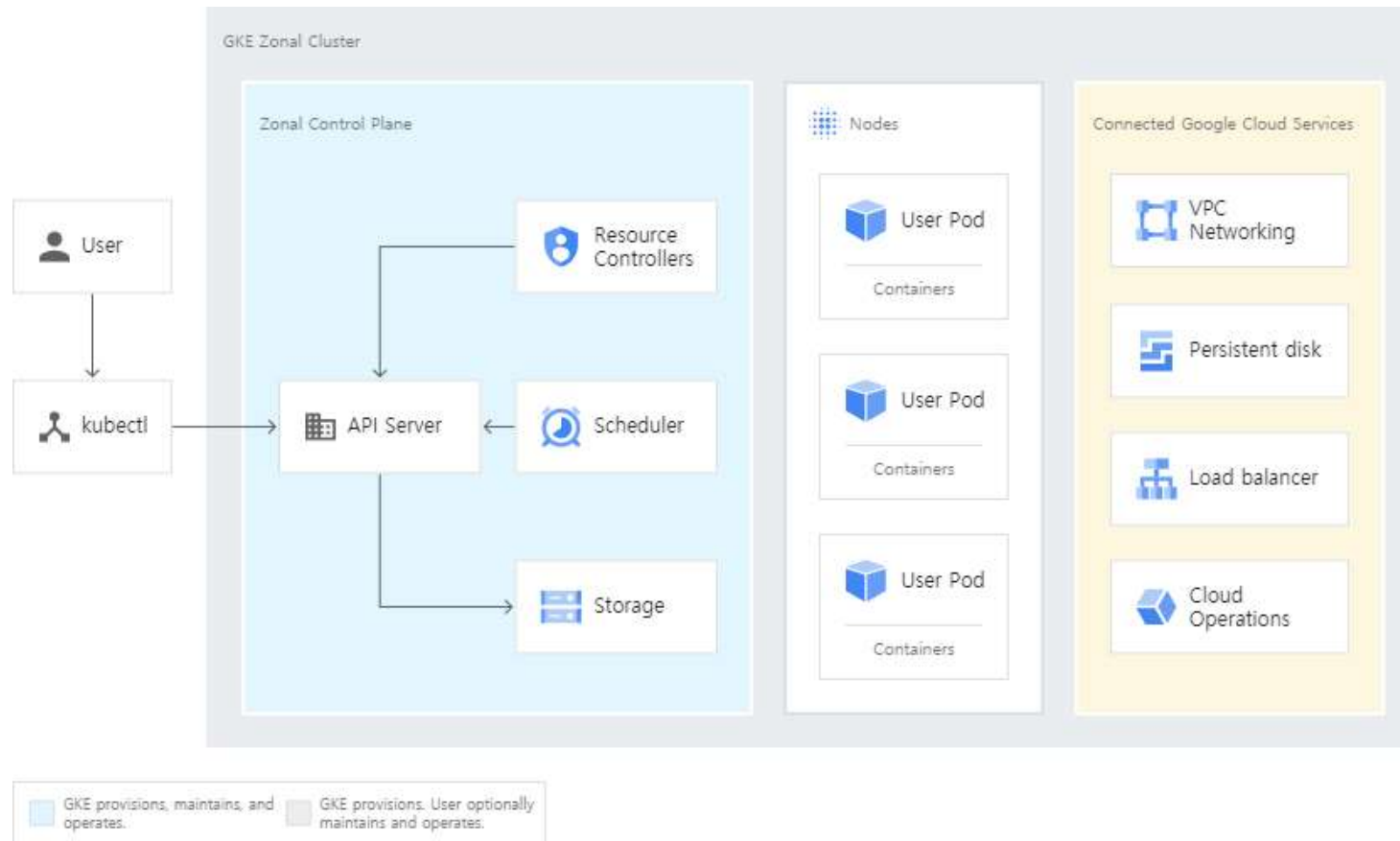
※ 참고 : <https://blog.codonomics.com/2019/02/loadbalancer-support-with-minikube-for-k8s.html>

k3s



※ 참고 : <https://k3s.io/>

GKE



※ 참고 : <https://cloud.google.com/kubernetes-engine/docs/concepts/cluster-architecture?hl=ko>

katacoda

Kubernetes Playground

Launch Cluster

`launch.sh`

This will create a two node Kubernetes cluster.

Health Check

`kubectl cluster-info`

Interested in writing your own Kubernetes scenarios and demos? Visit www.katacoda.com/teach

CONTINUE

Terminal Host 1

```
controlplane $ kubectl cluster-info
Kubernetes master is running at https://172.17.0.26:6443
KubeDNS is running at https://172.17.0.26:6443/api/v1/namespaces/kube-system/services/kube-dns:dns/proxy

To further debug and diagnose cluster problems, use 'kubectl cluster-info dump'.
controlplane $ kubectl get nodes
NAME             STATUS    ROLES    AGE   VERSION
controlplane     Ready    master   78s   v1.18.0
node01           Ready    <none>    46s   v1.18.0
controlplane $
```

Terminal Host 2

```
Your Interactive Bash Terminal.

node01 $
```

※ 참고 : <https://www.katacoda.com/courses/kubernetes/playground>

<https://kahoot.it/>