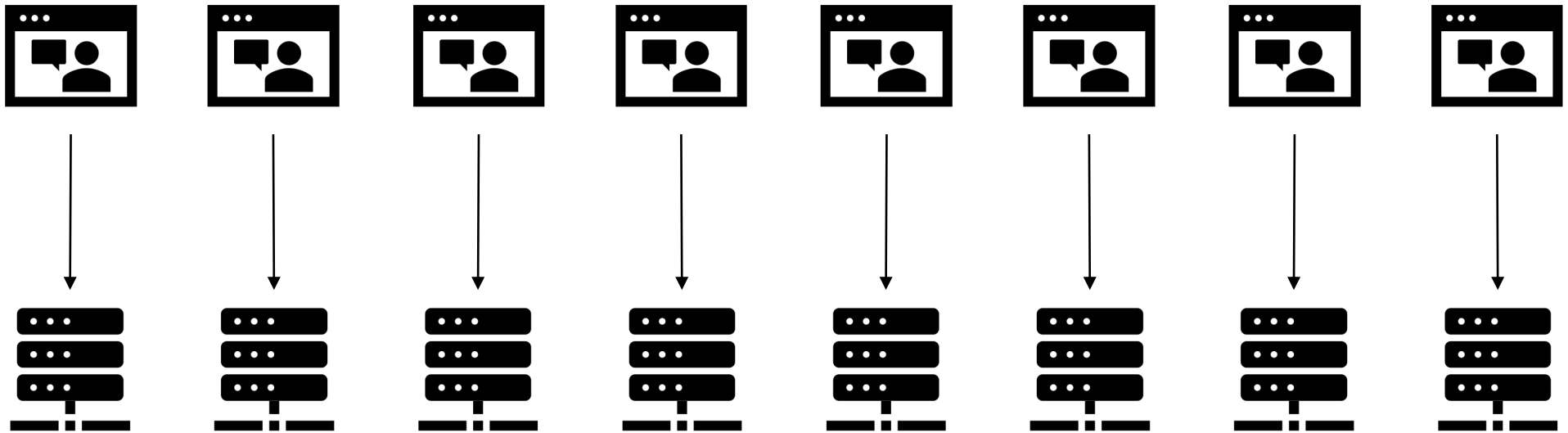
container 기술과 docker 이해하기

강사 : 윤성국

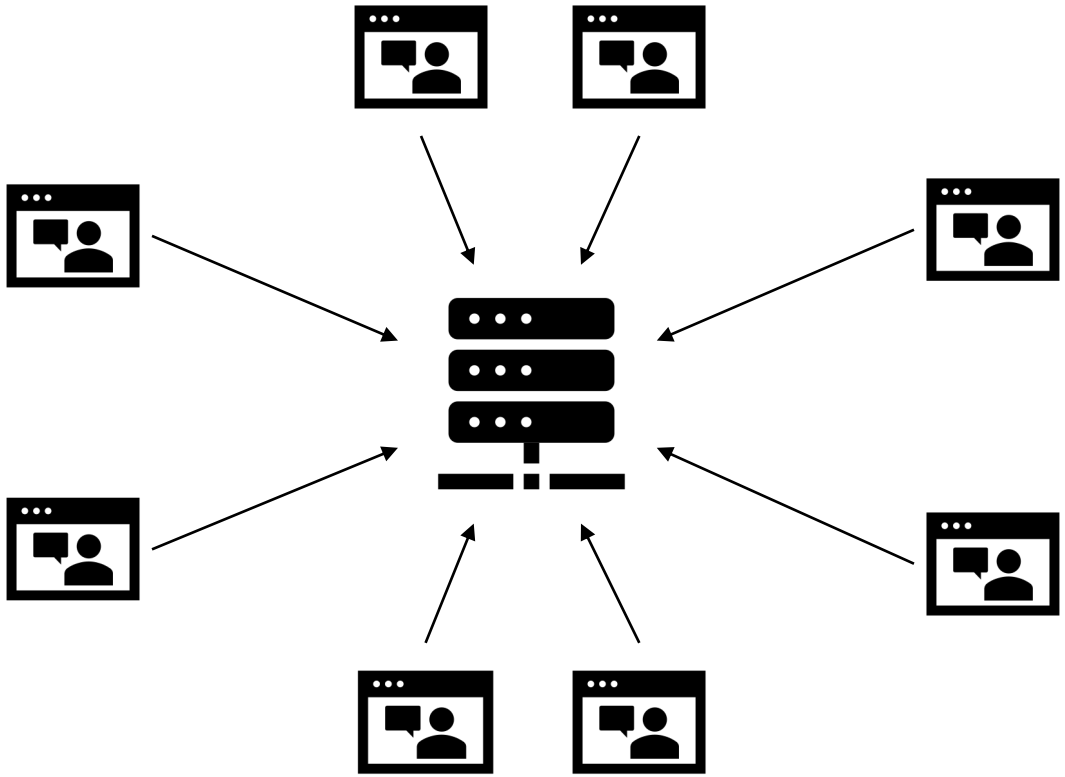
가상화는 왜 필요할까요?

프록시, 서버 application, DB 등을 구동하기 위해서 각각을 별도의 물리서버에 구동

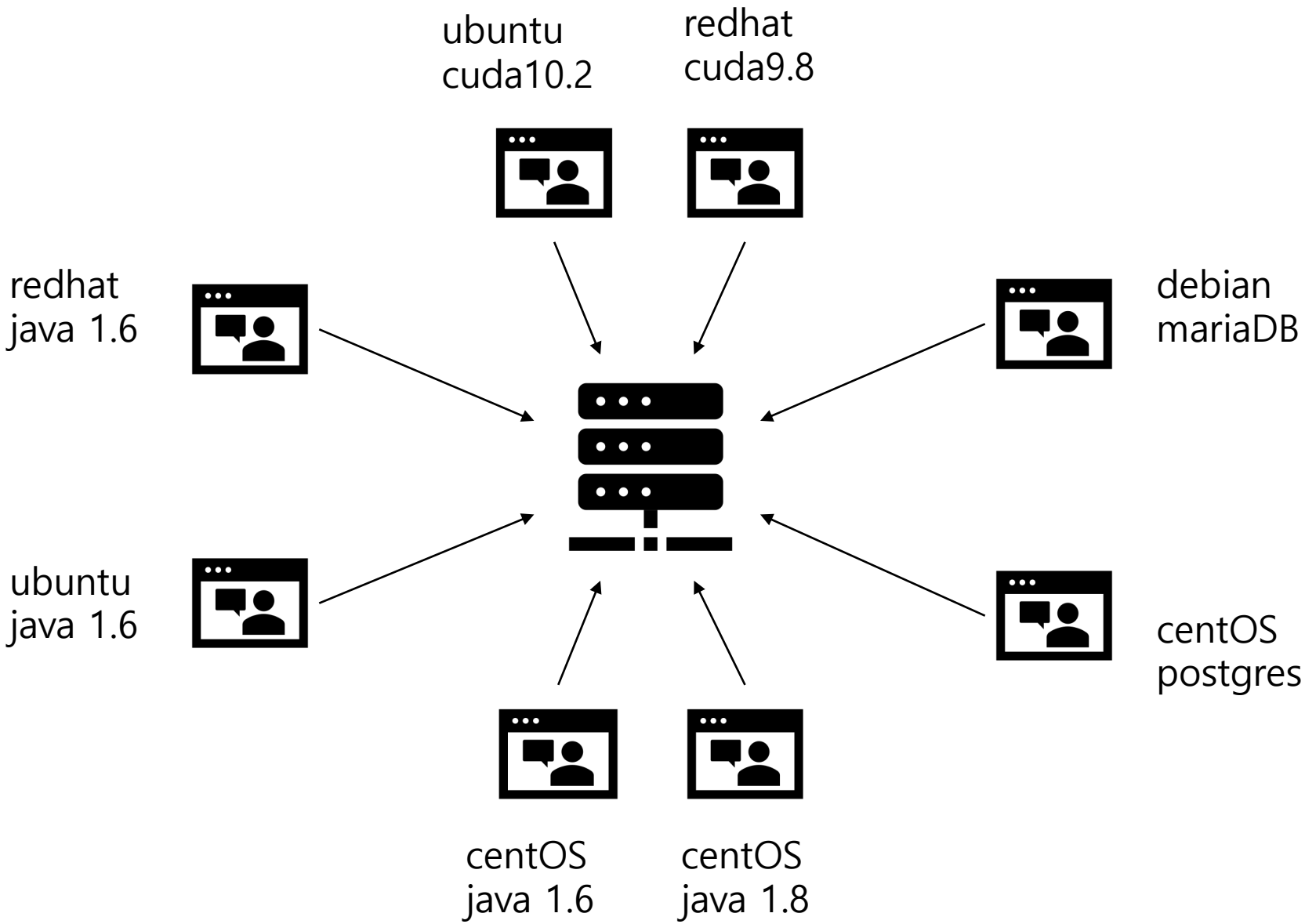


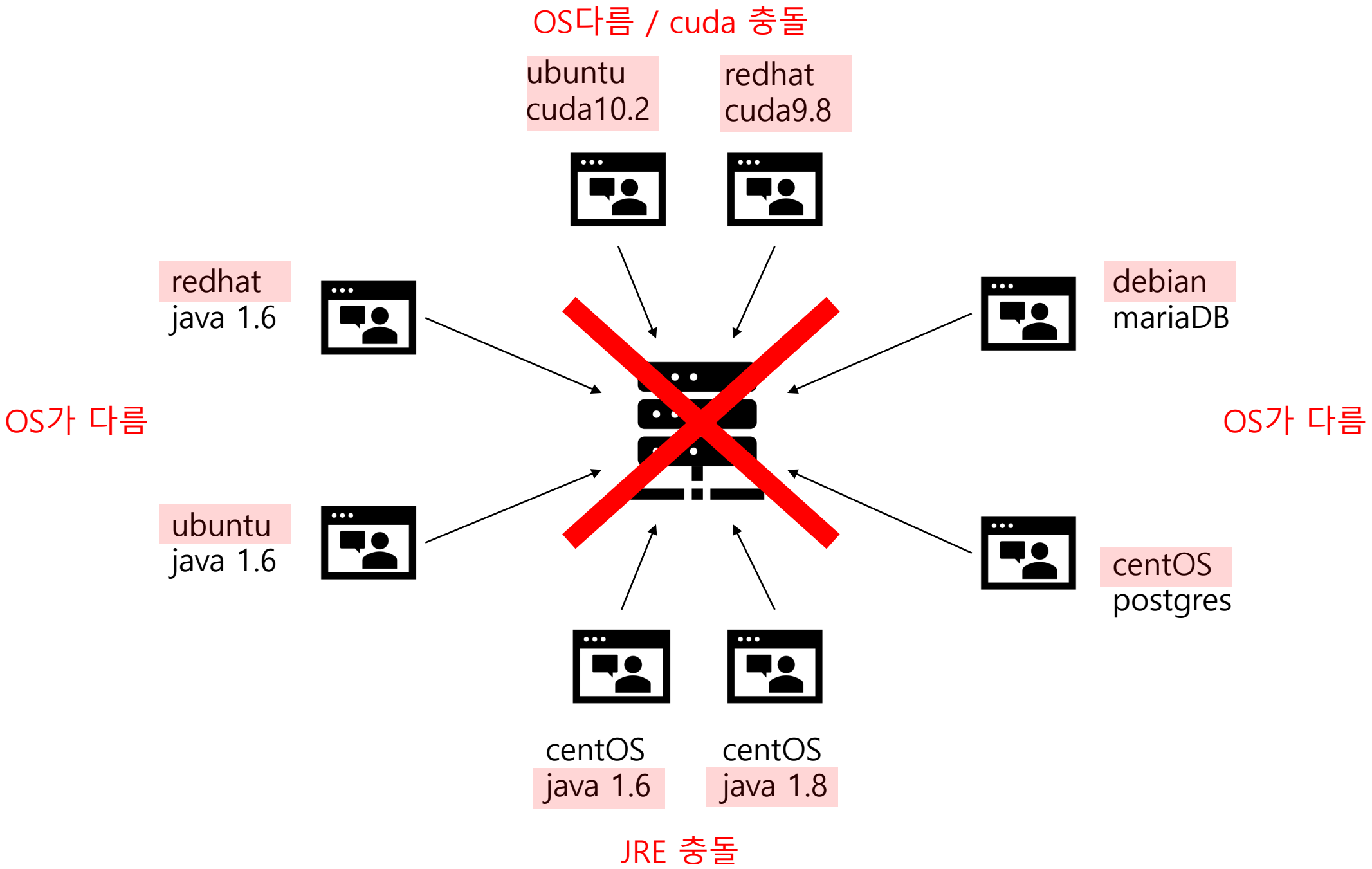
- 물리서버 구입 및 관리에 대한 비용
- 소프트웨어 추가 시, 추가적인 물리서버에 대한 비용 추가
- 네트워크 및 물리적 공간 등의 부대비용 또한 증가
- 서버 추가 시, 환경세팅에 많은 시간과 인력이 소요

프록시, 서버 application, DB 등을 구동하기 위해서 각각을 하나의 물리서버에 구동

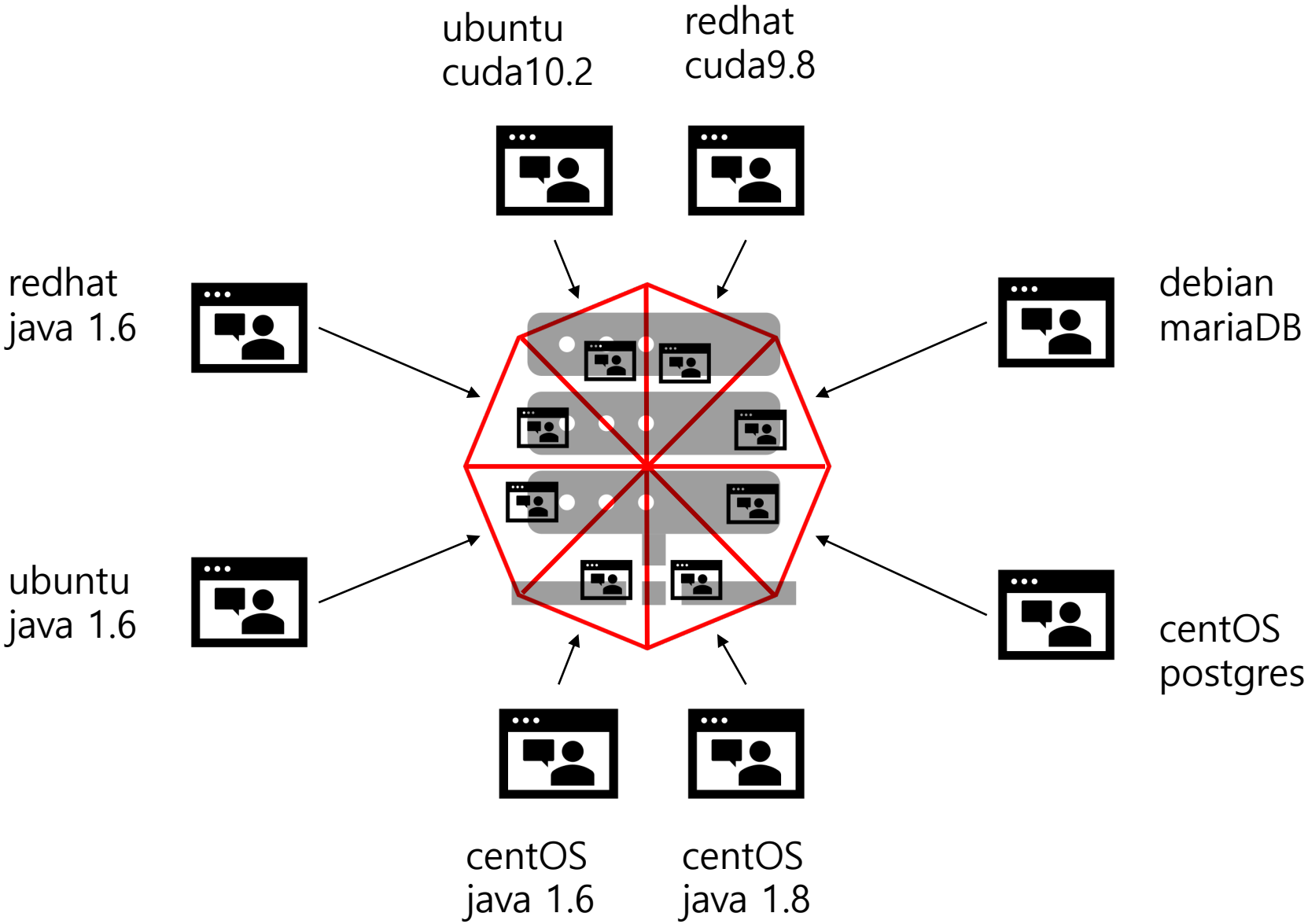


- 하드웨어 비용 절감
- 관리에 대한 편의
- 네트워크 및 물리적 공간 등의 부대비용 절감


ubuntu
cuda10.2
redhat
cuda9.8
redhat
java 1.6
debian
mariaDB
ubuntu
java 1.6
centOS
postgres
centOS
java 1.6
centOS
java 1.8

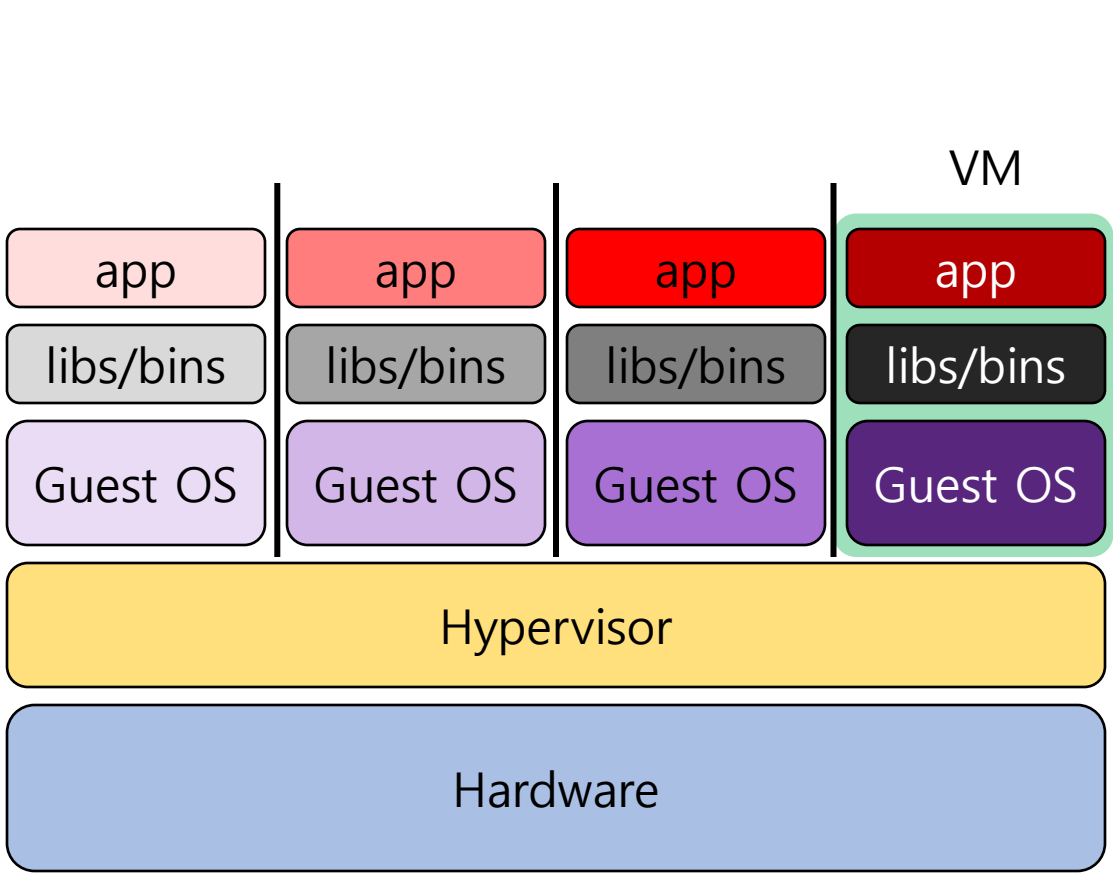

OS다름 / cuda 충돌
ubuntu
cuda10.2
redhat
cuda9.8
redhat
java 1.6
debian
mariaDB
OS가 다름
OS가 다름
ubuntu
java 1.6
centOS
postgres
centOS
java 1.6
centOS
java 1.8
JRE 충돌

isolation

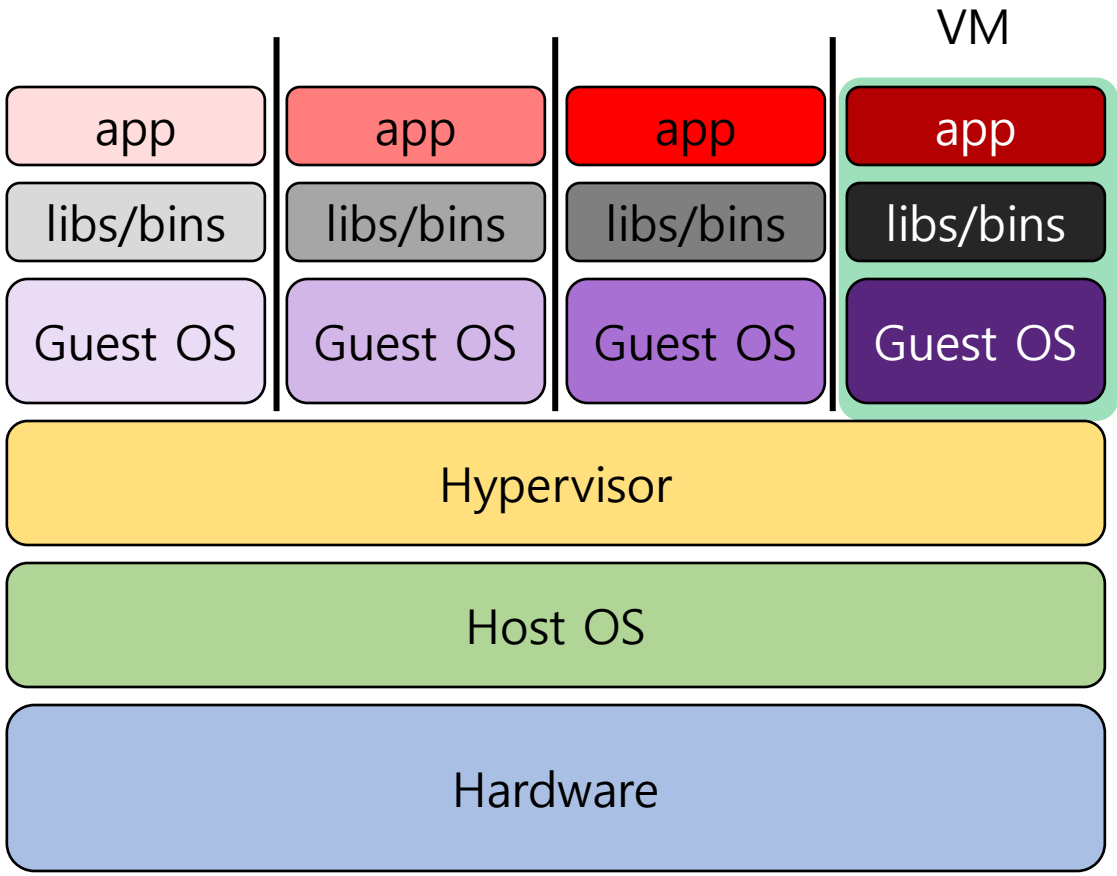

ubuntu
cuda10.2
redhat
cuda9.8
redhat
java 1.6
debian
mariaDB
ubuntu
java 1.6
centOS
postgres
centOS
java 1.6
centOS
java 1.8

그래서 필요한 것이

Hypervisor

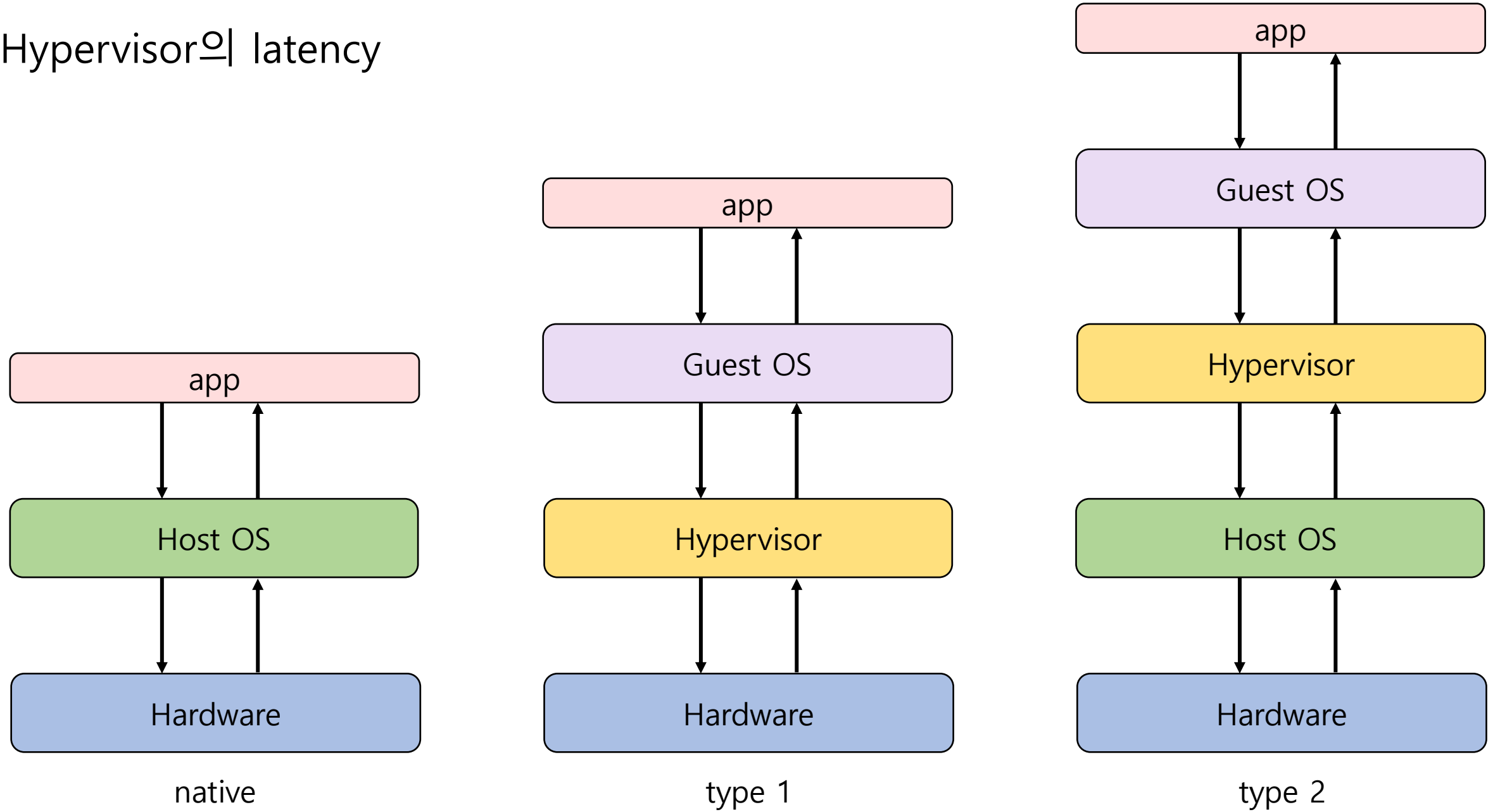


type 1



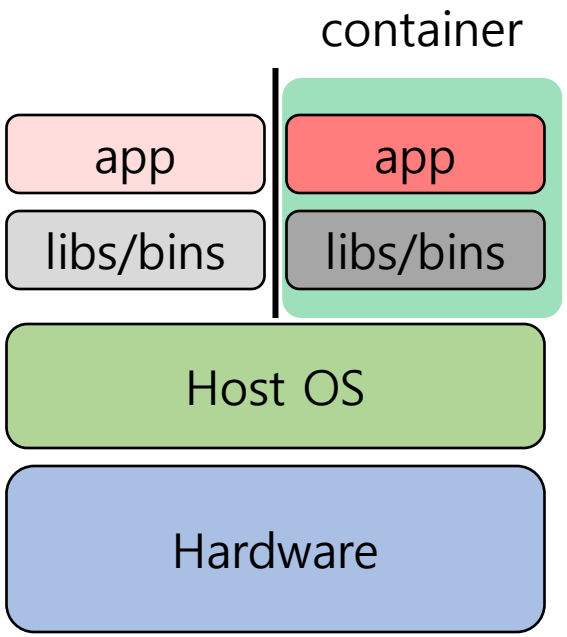
type 2

Hypervisor의 latency

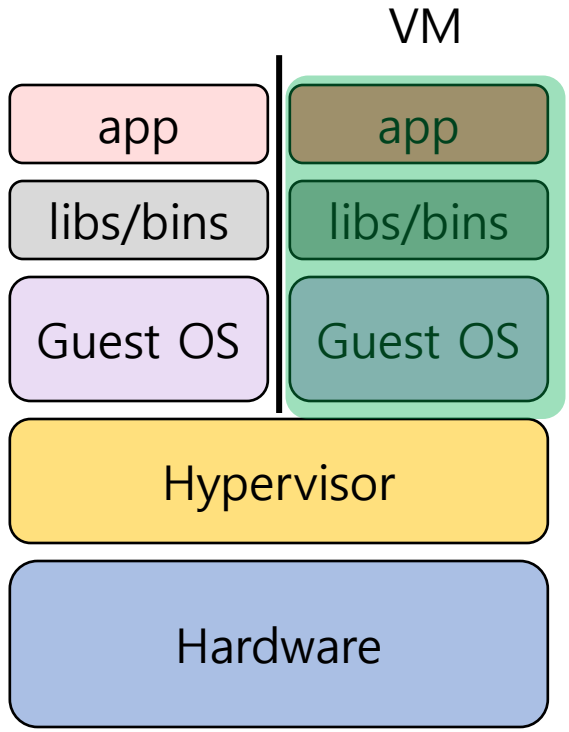


그래서 만들어진 것이 바로

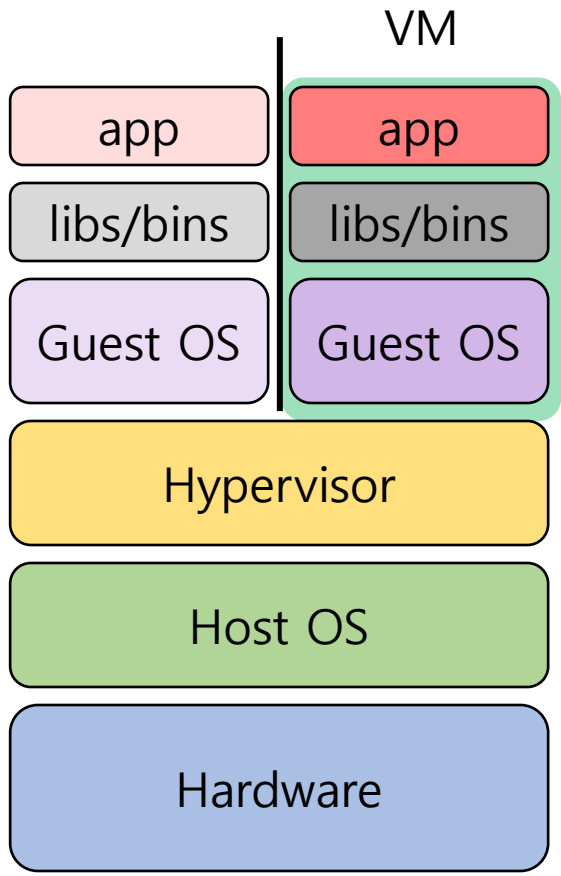
LXC



LXD

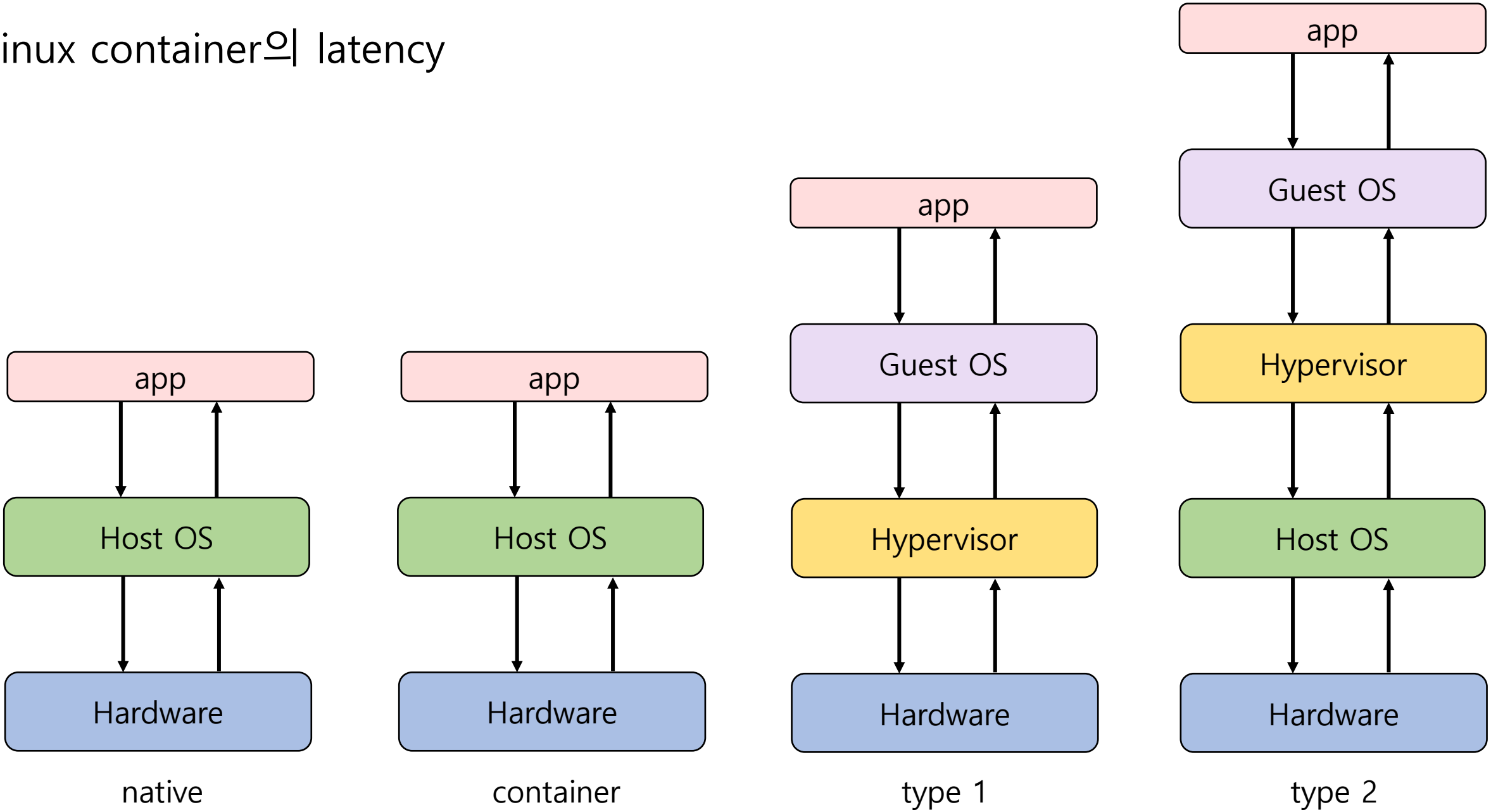


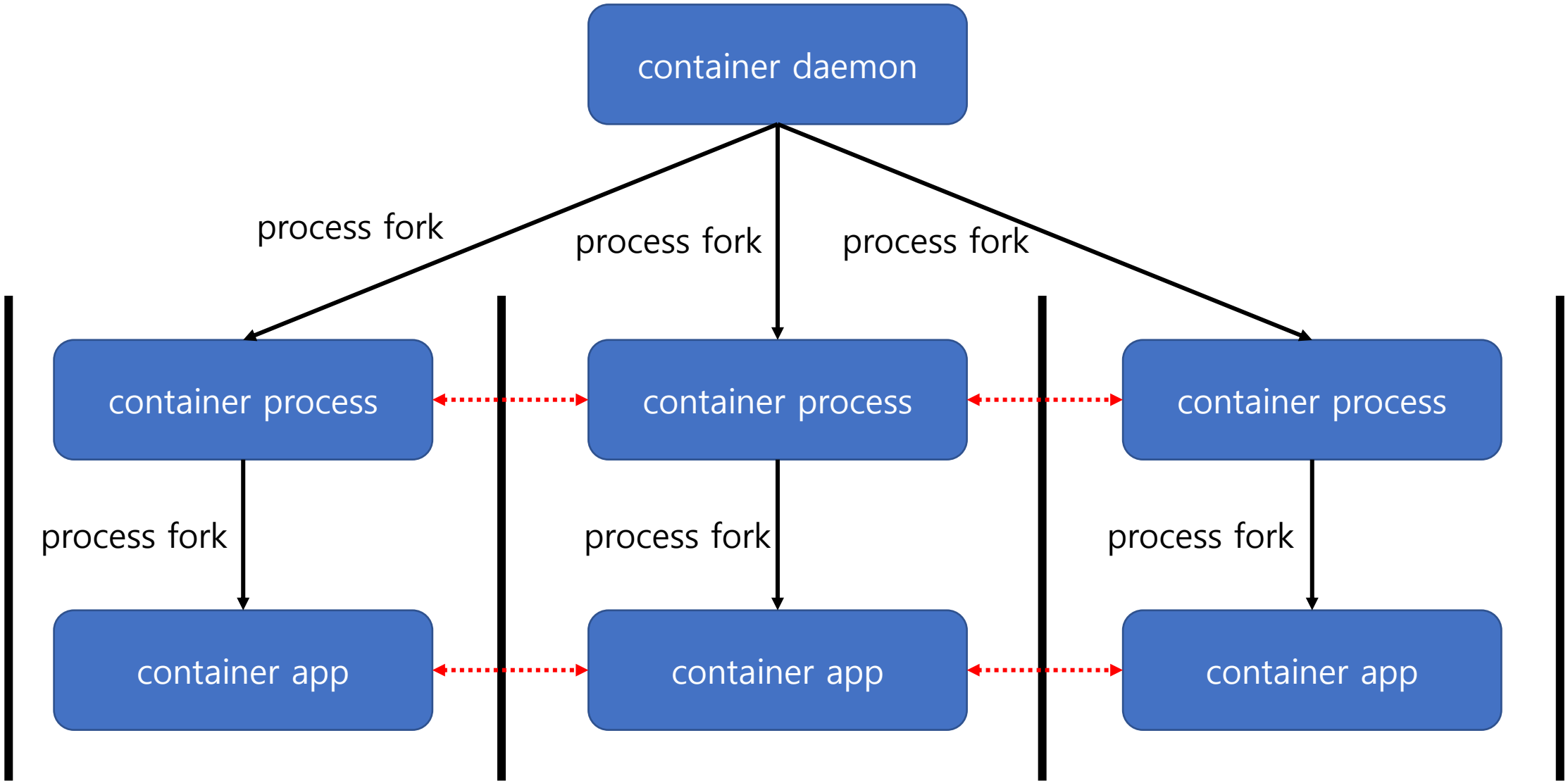
type 1



type 2

linux container의 latency





어떻게 격리가 된 것처럼 속일 수 있을까?

container의 virtualization 및 isolat

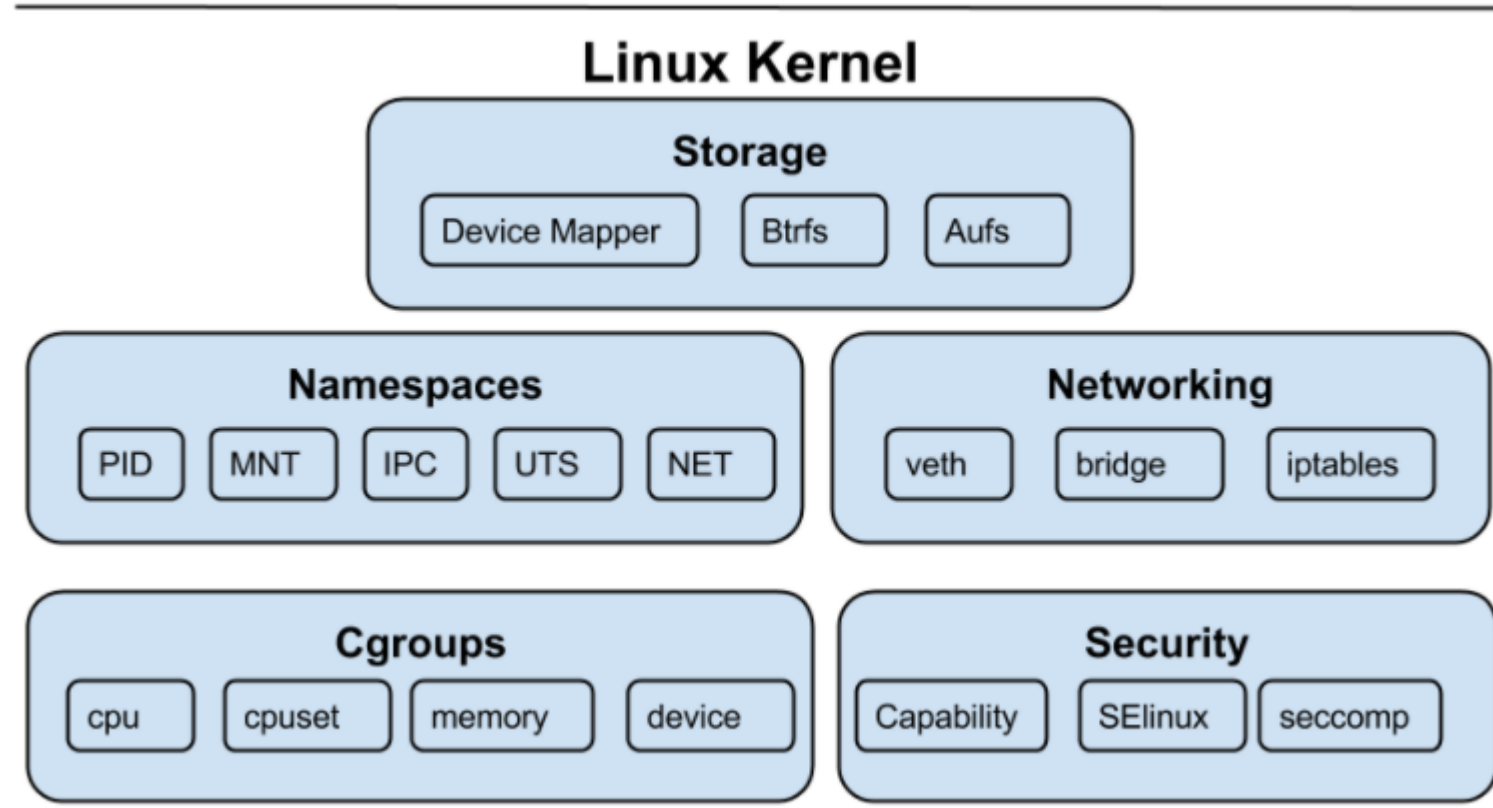
cgroups

na

selin

capabilities

apparmor



container를 지탱하는 기술

Namespaces

Mount namespaces

UTS namespaces

IPC namespaces

PID namespaces

Net namespaces

User namespaces

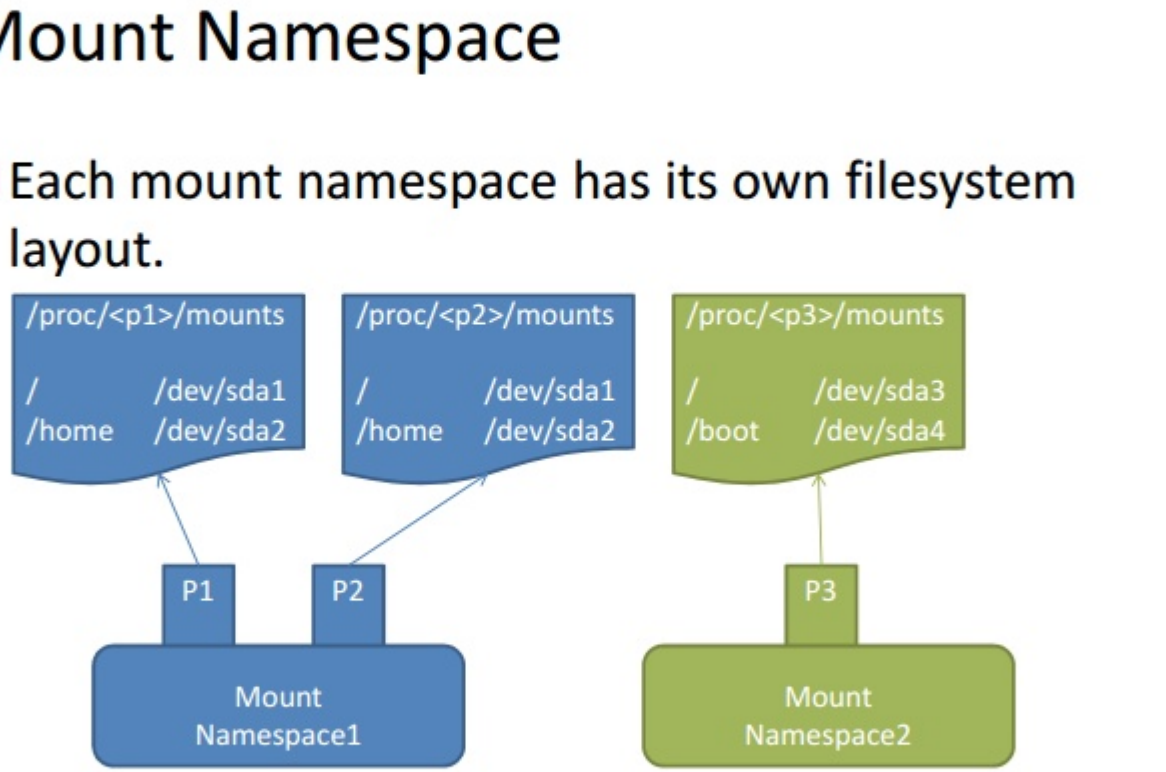


Image.
Lightweight Virtual System Mechanism: Linux Container - Gao Feng, Fujitsu
p://events.linuxfoundation.org/events/archive/2013/cloudopen-japan/program/presentations
http://events.linuxfoundation.org/sites/events/files/cojp13_feng.pdf

Namespaces

Mount namespaces

UTS namespaces

IPC namespaces

PID namespaces

Net namespaces

User namespaces

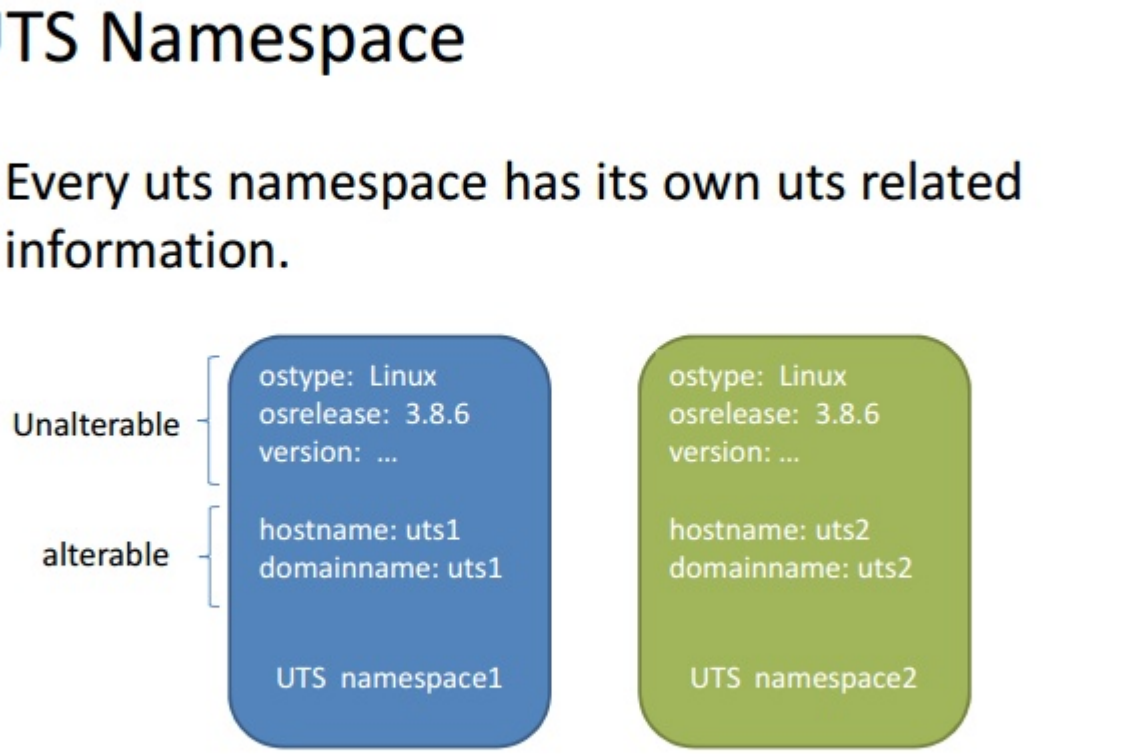


Image.
Lightweight Virtual System Mechanism: Linux Container - Gao Feng, Fujitsu
http://events.linuxfoundation.org/events/archive/2013/cloudopen-japan/program/presentations
http://events.linuxfoundation.org/sites/events/files/cojp13_feng.pdf

Namespaces

Mount namespaces

UTS namespaces

IPC namespaces

PID namespaces

Net namespaces

User namespaces

IPC Namespace

- IPC namespce isolates the interprocess communication resource(shared memory, semaphore, message queue)

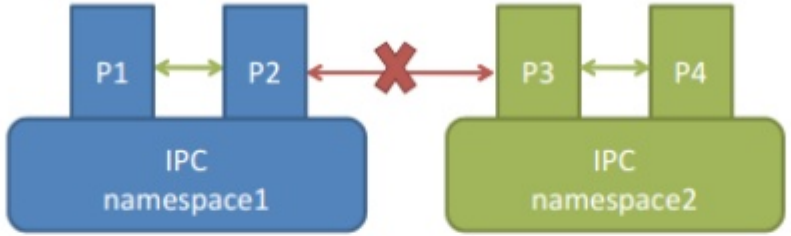


Image.
Lightweight Virtual System Mechanism: Linux Container - Gao Feng, Fujitsu
http://events.linuxfoundation.org/events/archive/2013/cloudopen-japan/program/presentations
http://events.linuxfoundation.org/sites/events/files/cojp13_feng.pdf

Namespaces

Mount namespaces

UTS namespaces

IPC namespaces

PID namespaces

Net namespaces

User namespaces

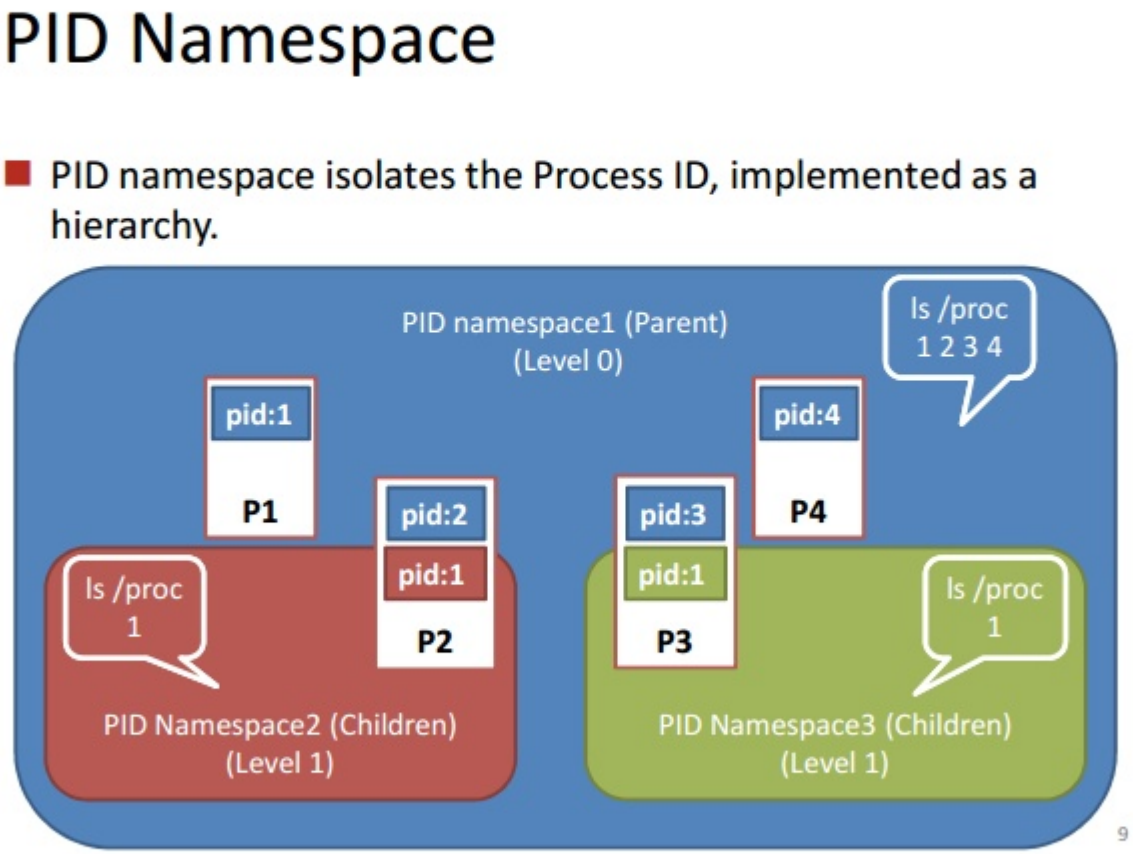


Image.
Lightweight Virtual System Mechanism: Linux Container - Gao Feng, Fujitsu
http://events.linuxfoundation.org/events/archive/2013/cloudopen-japan/program/presentations
http://events.linuxfoundation.org/sites/events/files/cojp13_feng.pdf

Namespaces

Mount namespaces

UTS namespaces

IPC namespaces

PID namespaces

Net namespaces

User namespaces

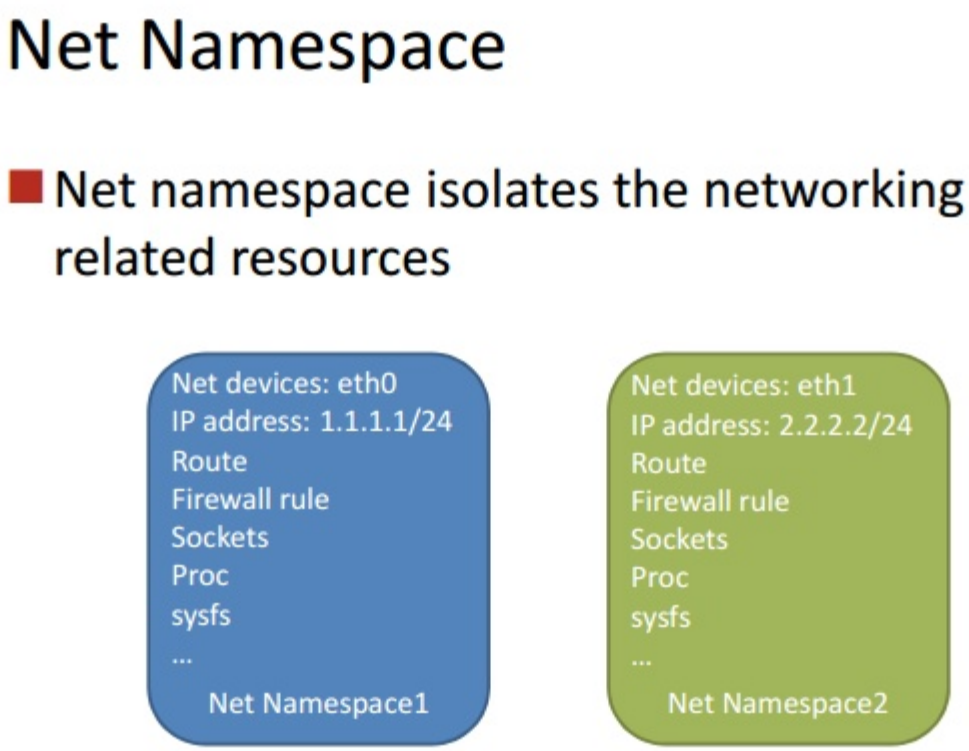


Image.
Lightweight Virtual System Mechanism: Linux Container - Gao Feng, Fujitsu
http://events.linuxfoundation.org/events/archive/2013/cloudopen-japan/program/presentations
http://events.linuxfoundation.org/sites/events/files/cojp13_feng.pdf

Namespaces

Mount namespaces

UTS namespaces

IPC namespaces

PID namespaces

Net namespaces

User namespaces

User Namespace

- kuid/kgid: Original uid/gid, Global
- uid/gid: user id in user namespace, will be translated to kuid/kgid finally

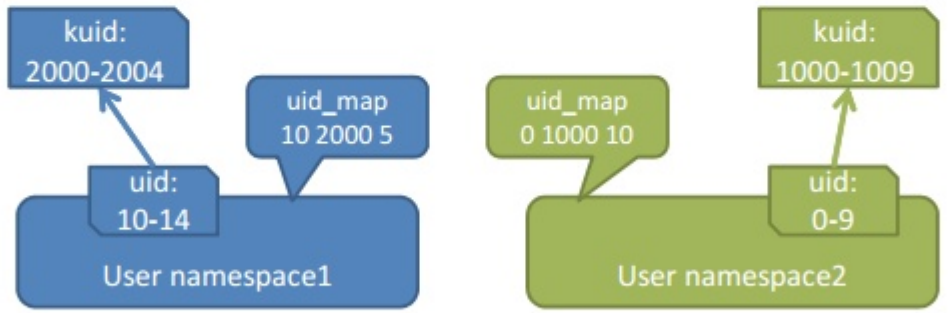
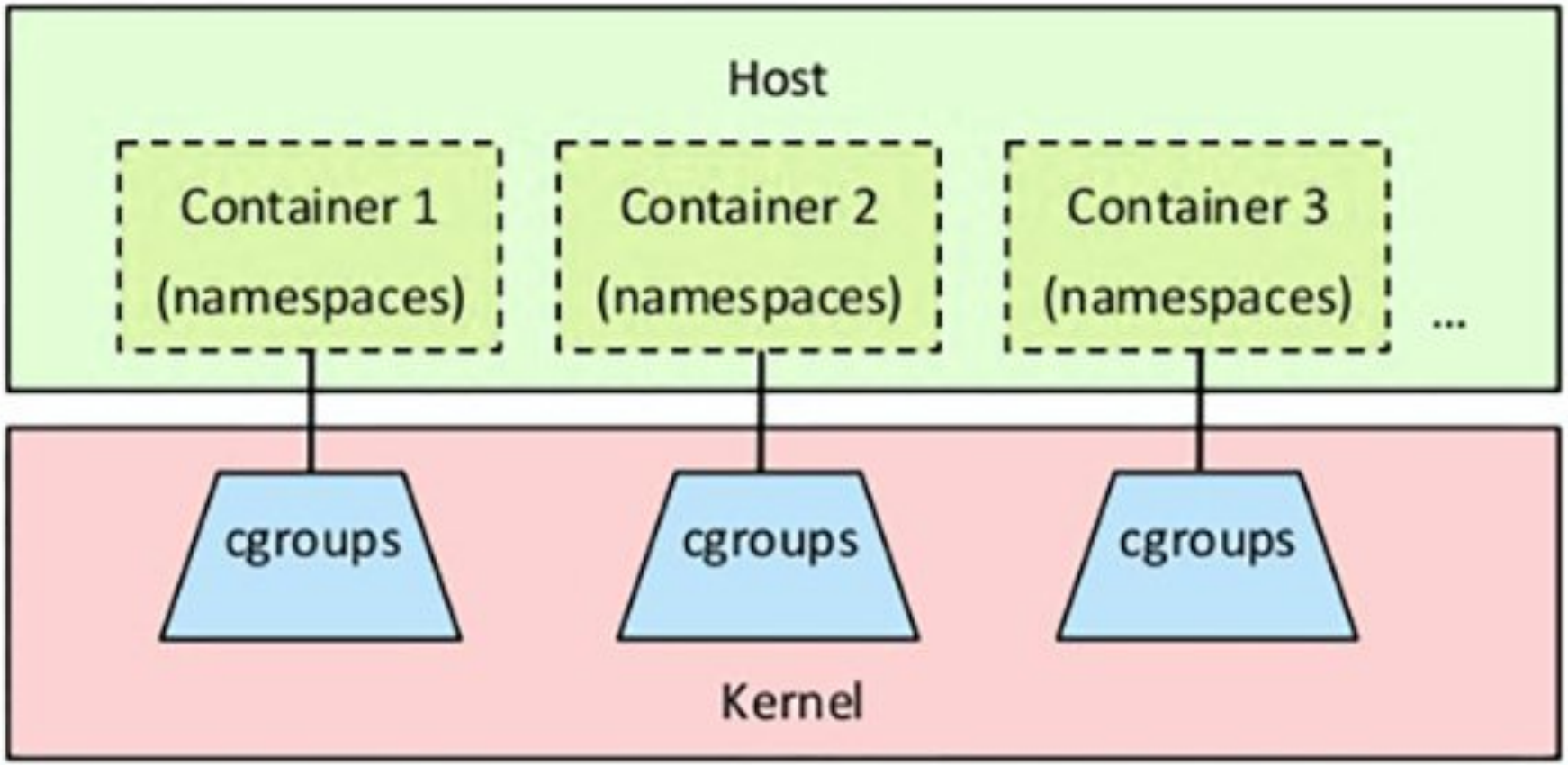


Image.
Lightweight Virtual System Mechanism: Linux Container - Gao Feng, Fujitsu
http://events.linuxfoundation.org/events/archive/2013/cloudopen-japan/program/presentations
http://events.linuxfoundation.org/sites/events/files/cojp13_feng.pdf

Cgroups

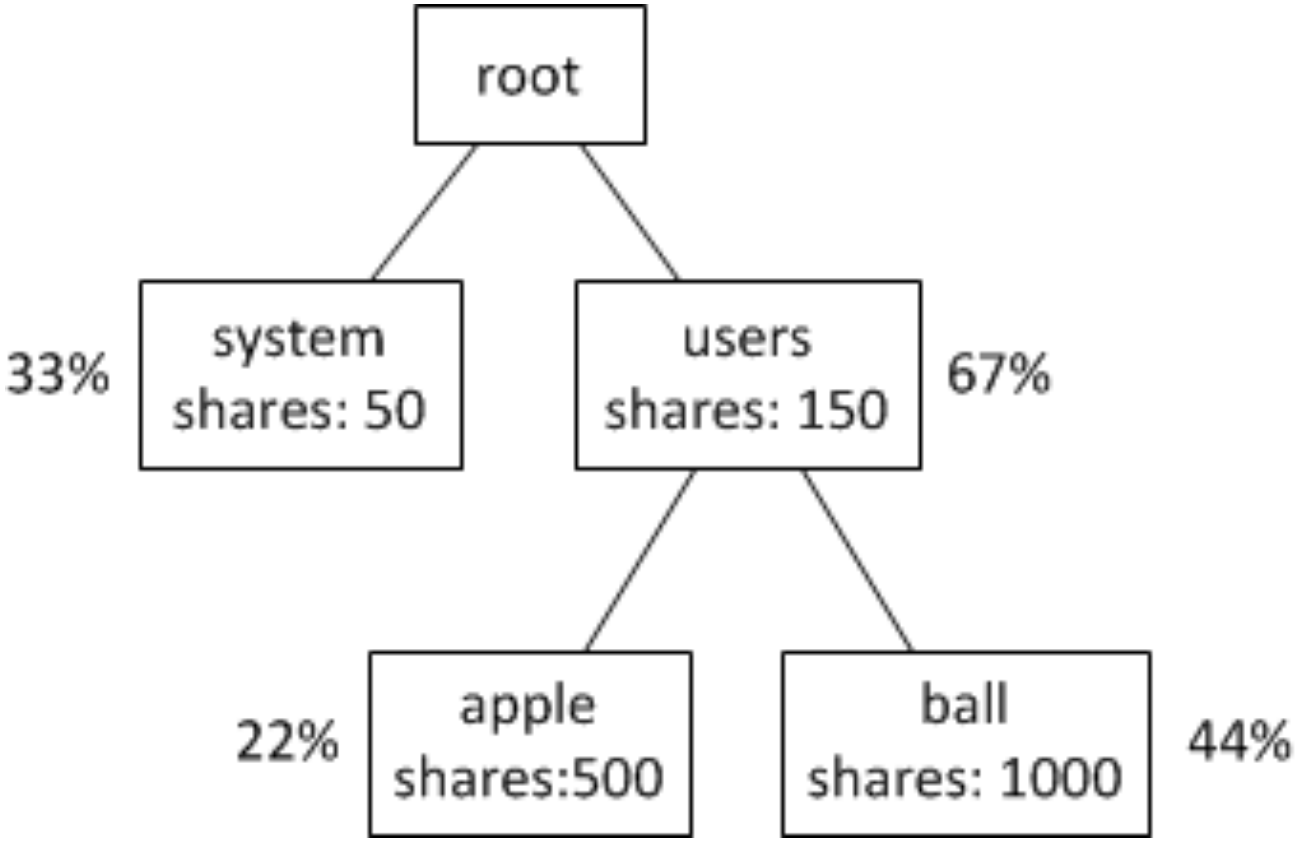
namespace는 프로세스를 분리 및 격리
cgroups 각 프로세스에 리소스를 분리 및 할당



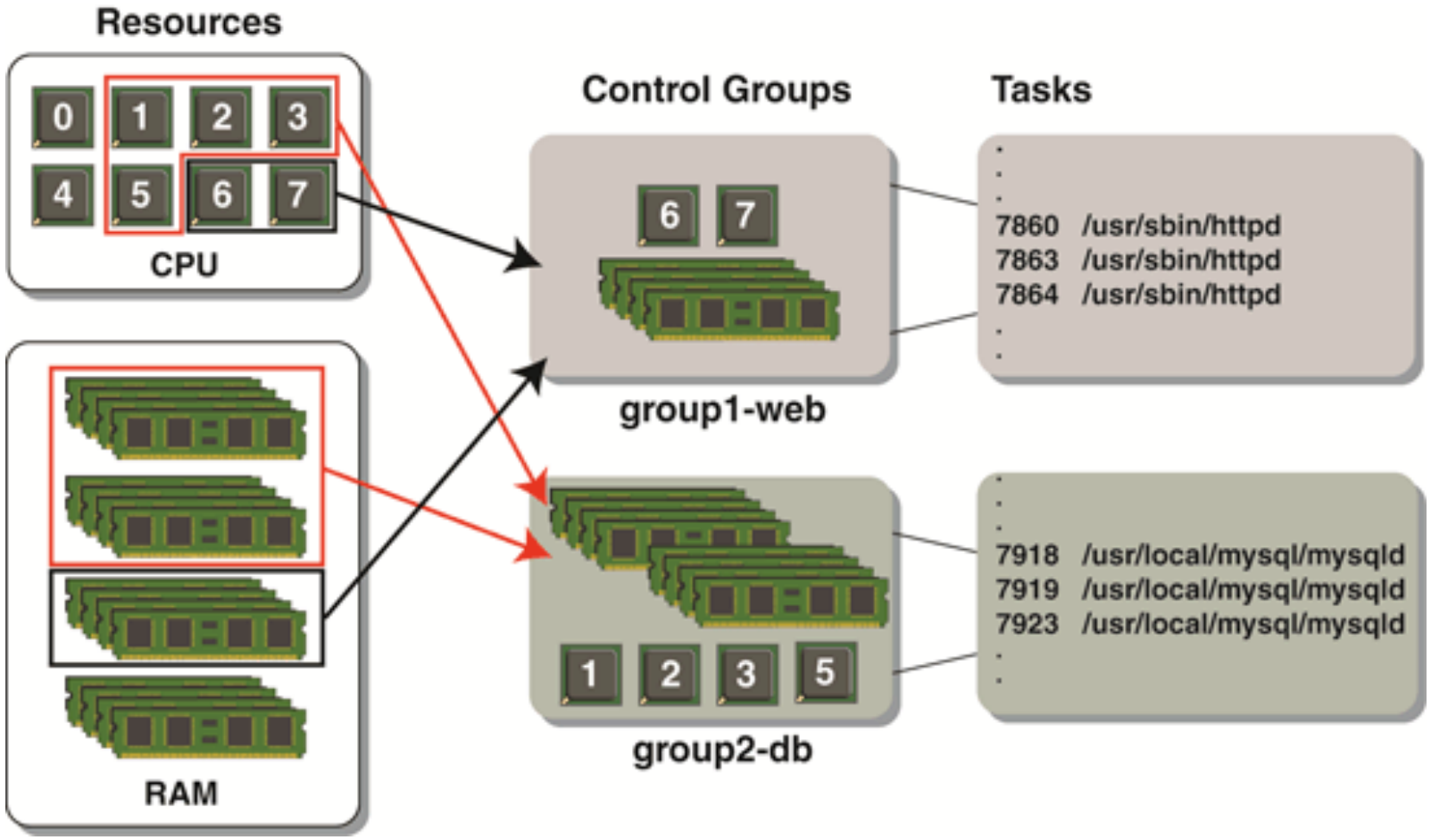
Subsystem	Tunable Parameters
blkio	- Weighted proportional block I/O access. Group wide or per device. - Per device hard limits on block I/O read/write specified as bytes per second or IOPS per second.
cpu	- Time period (microseconds per second) a group should have CPU access. - Group wide upper limit on CPU time per second. - Weighted proportional value of relative CPU time for a group.
cpuset	- CPUs (cores) the group can access. - Memory nodes the group can access and migrate ability. - Memory hardwall, pressure, spread, etc.
devices	- Define which devices and access type a group can use.
freezer	- Suspend/resume group tasks.
memory	- Max memory limits for the group (in bytes). - Memory swappiness, OOM control, hierarchy, etc..
hugetlb	- Limit HugeTLB size usage. - Per cgroup HugeTLB metrics.
net_cls	- Tag network packets with a class ID. - Use tc to prioritize tagged packets.
net_prio	- Weighted proportional priority on egress traffic (per interface).

http://jake.dothome.co.kr/control-groups/

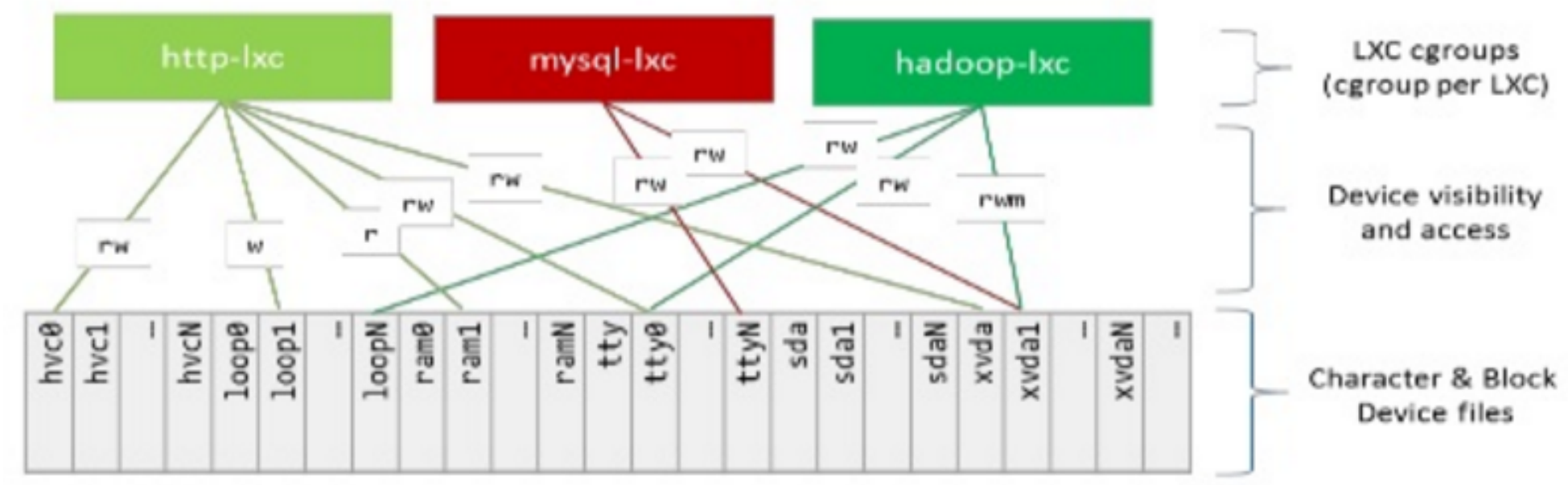
cgroups의 계층관리



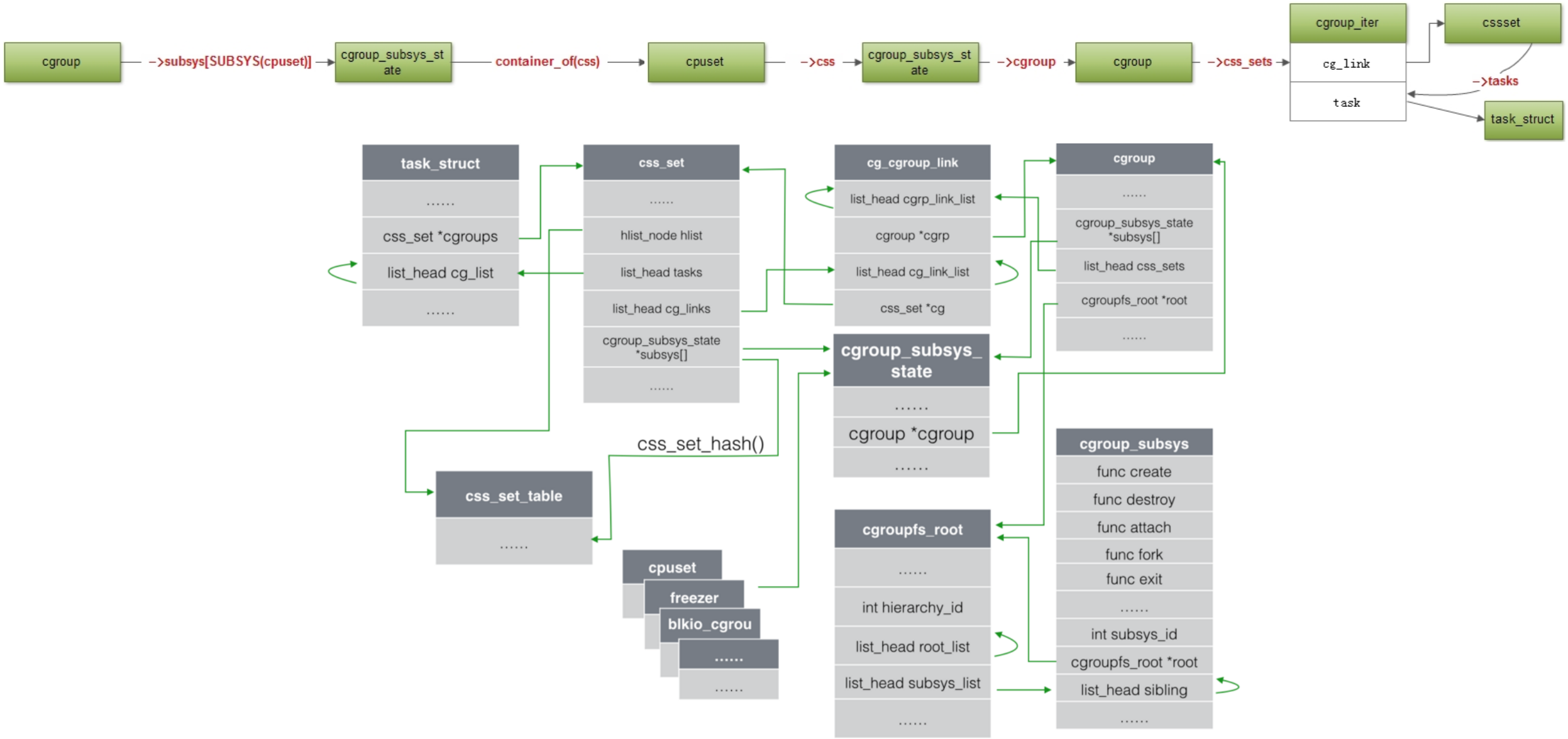
cgroups로 container 각각 따로 cpu, memory 할당 가능



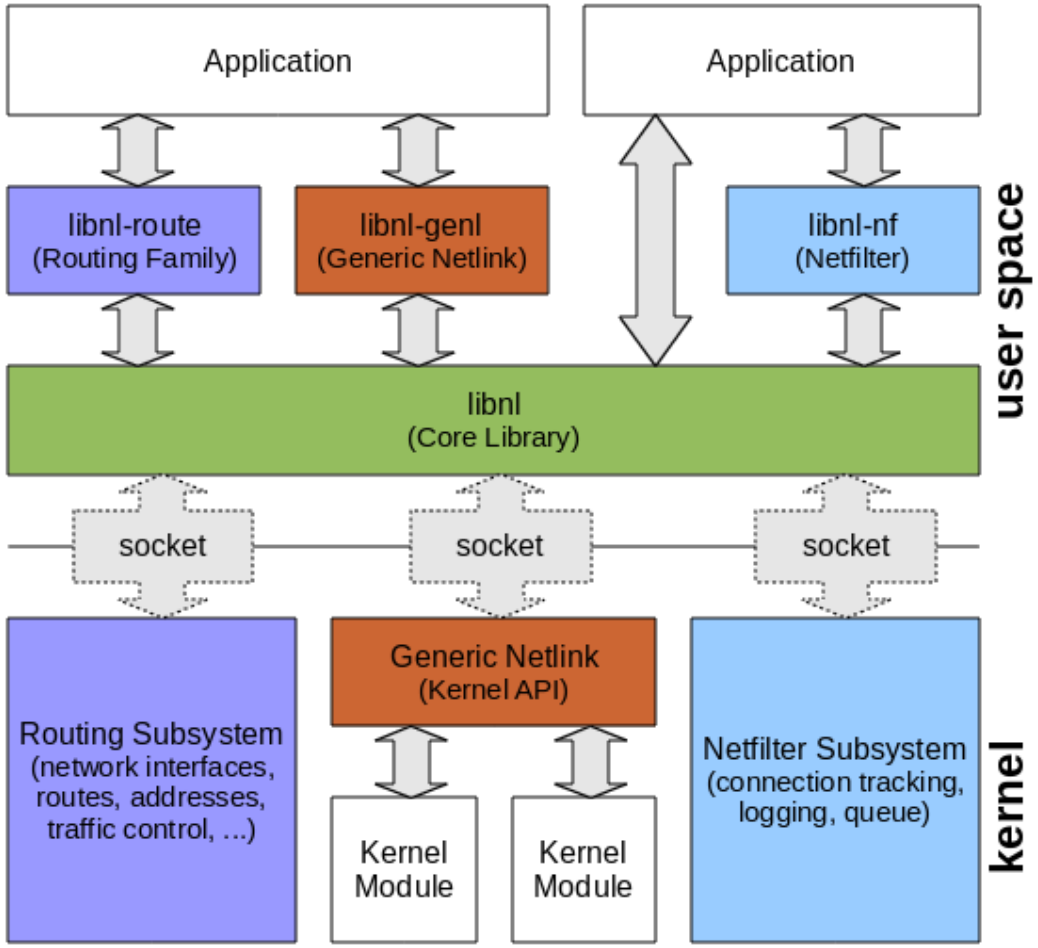
cgroups로 container 각각 따로 device 접근



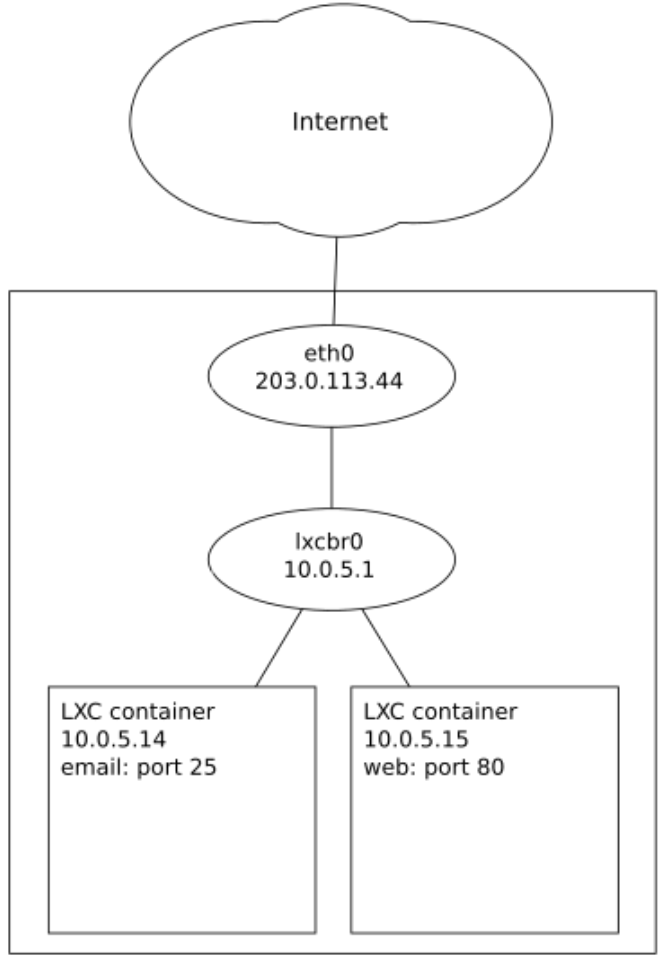
cgroups 내부 구조



container의 network

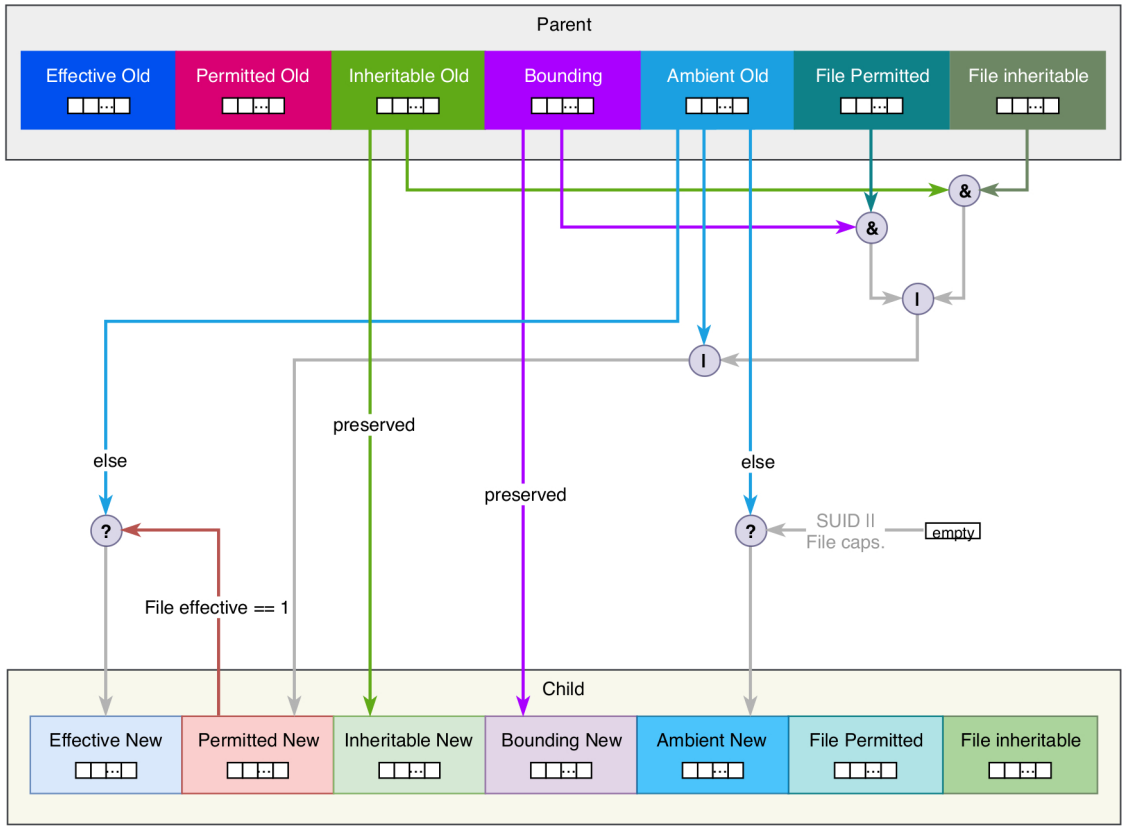


netlink



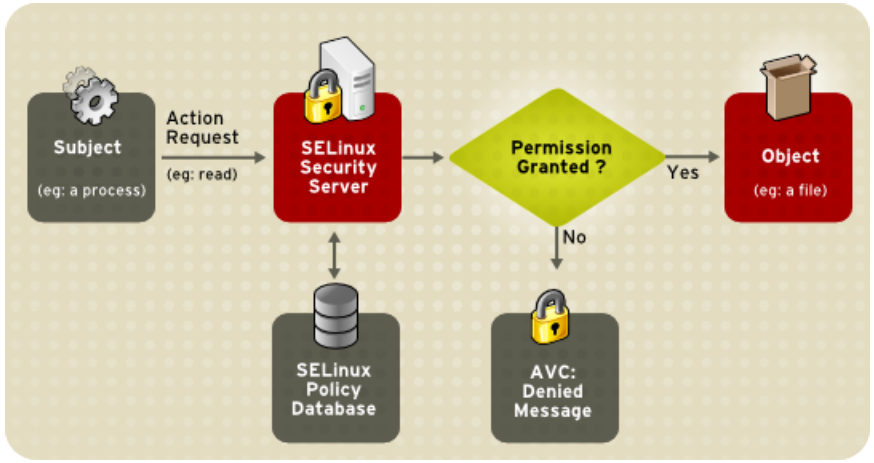
netfilter

container의 보안



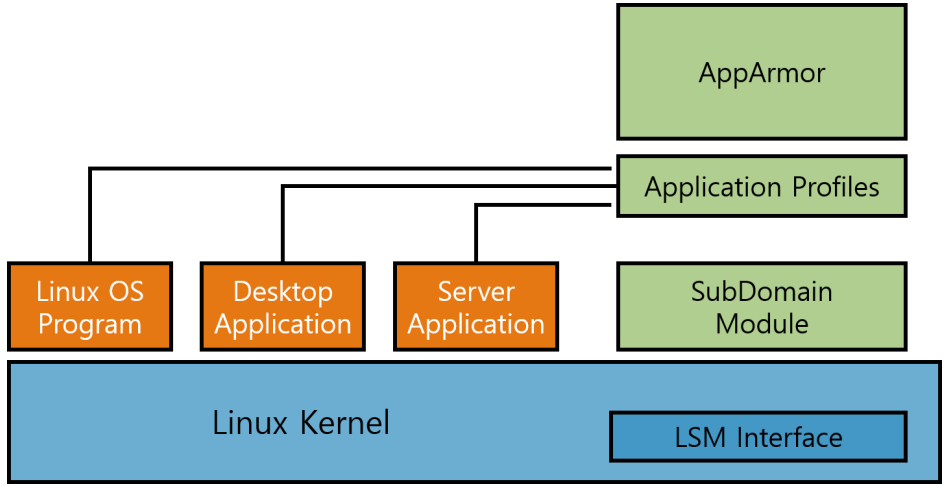
capacities

유효 UID와 GID의 권한 부여 및 권한 분할



SELinux

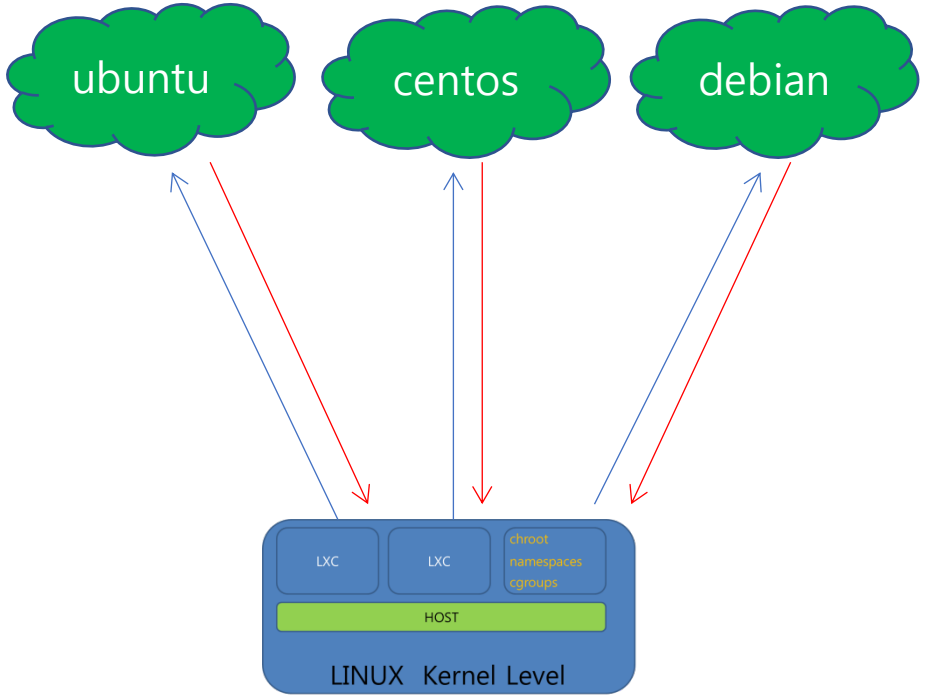
MAC (강제접근제어, Mandatory Access Control)이용해
시스템 전체 오브젝트에 대한 보안 설정



AppArmor

MAC (강제접근제어, Mandatory Access Control)이용해
개별 프로그램과 개별 프로그램의 커널 접근의 보안모델 구현

LXC가 해결해야 하는 두가지 문제

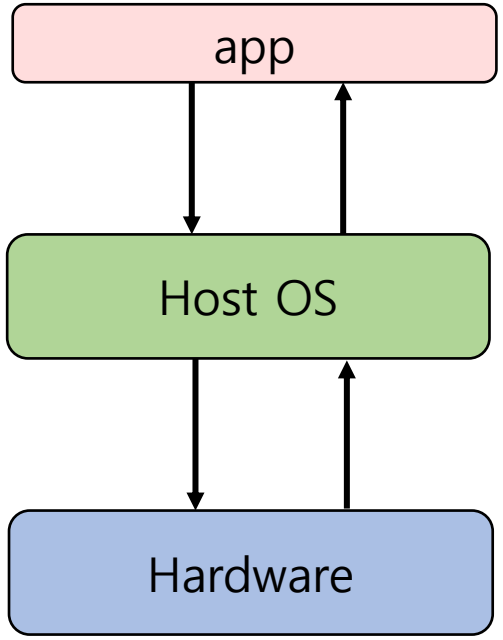


LXC 이미지 관리의 문제

리눅스 배포판 이미지 주요 사용

이미지 스냅샷은 가능하지만 정교한 이미지의 패키징 및 버전 관리의 어려움

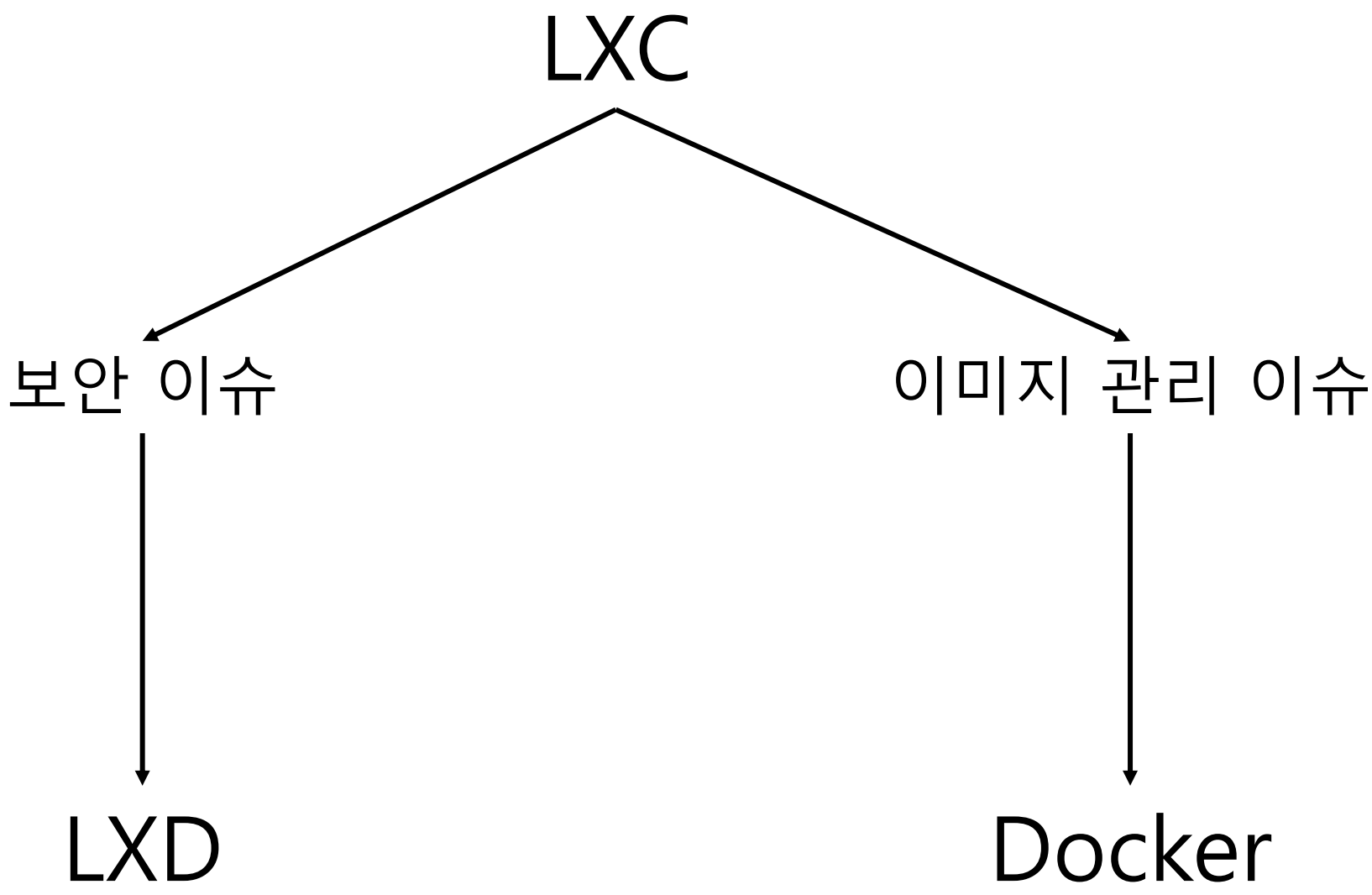
이미지의 전달에 대한 문제



LXC 보안의 문제

VM은 CPU 및 메모리를 에뮬레이팅 하므로 메모리 주소가 실제 커널과는 다르게 가상화 되어 있어 보안상 이점

커널을 직접 접근하는 컨테이너 태생적 보안 이슈


LXC
보안 이슈
이미지 관리 이슈
LXD
Docker

LXD basics

LXD runs on any architecture (Intel, AMD, ARM, POWER, IBM Mainframe) and in any cloud

Completely compatible with existing Linux container technologies

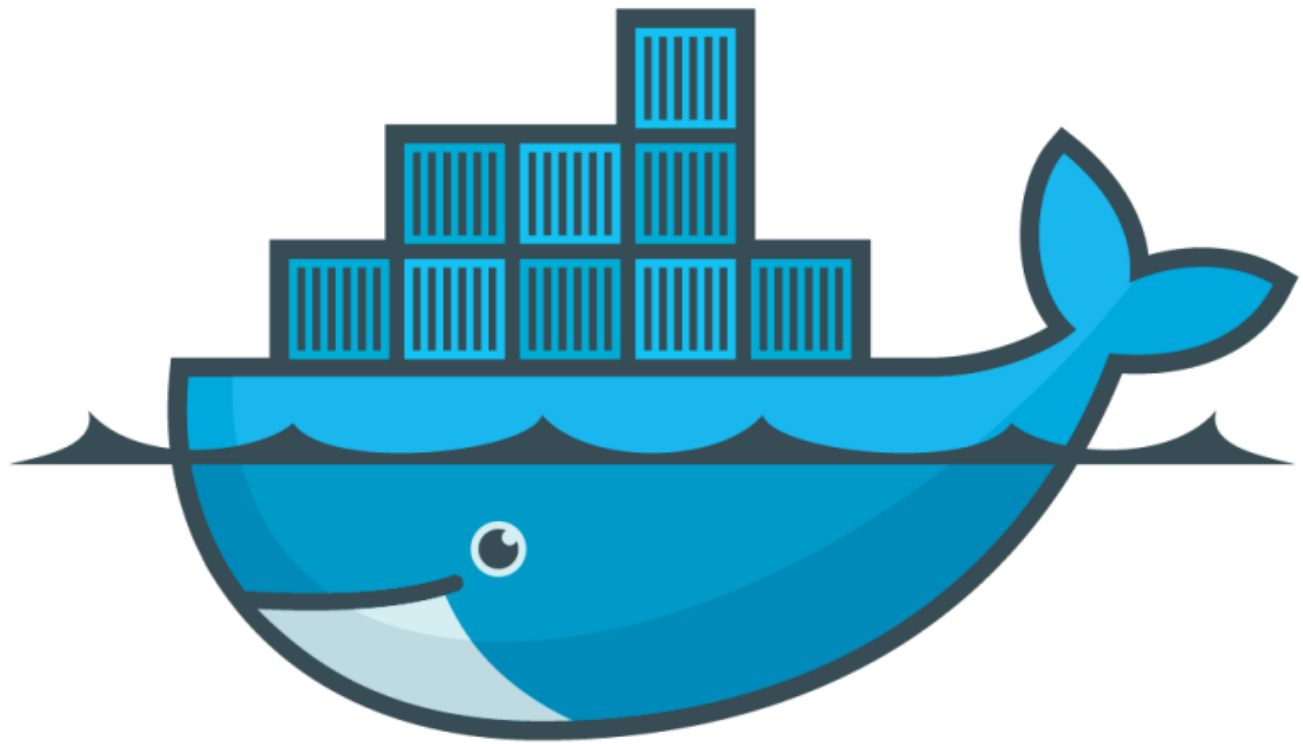
Any Linux distro can run in an Ubuntu LXD Machine container (e.g. Ubuntu/ Red Hat/ CentOS etc.)

Fast, dense, and secure container management

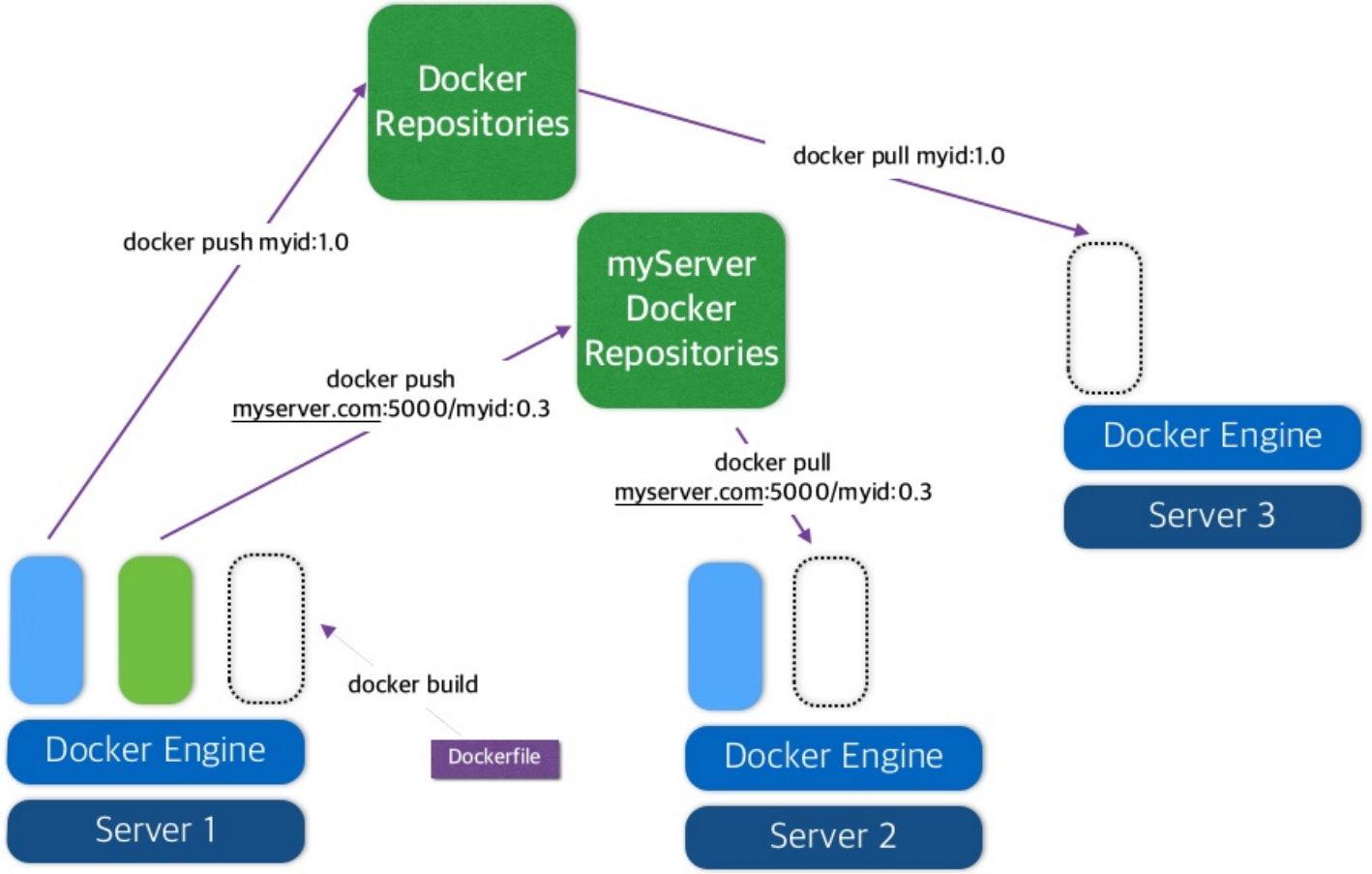
Gives you the density of containers with the manageability of VMs

Brings storage, network, and remote API interfaces to containers

그리고 드디어



Application of Docker : Packaging Delivery

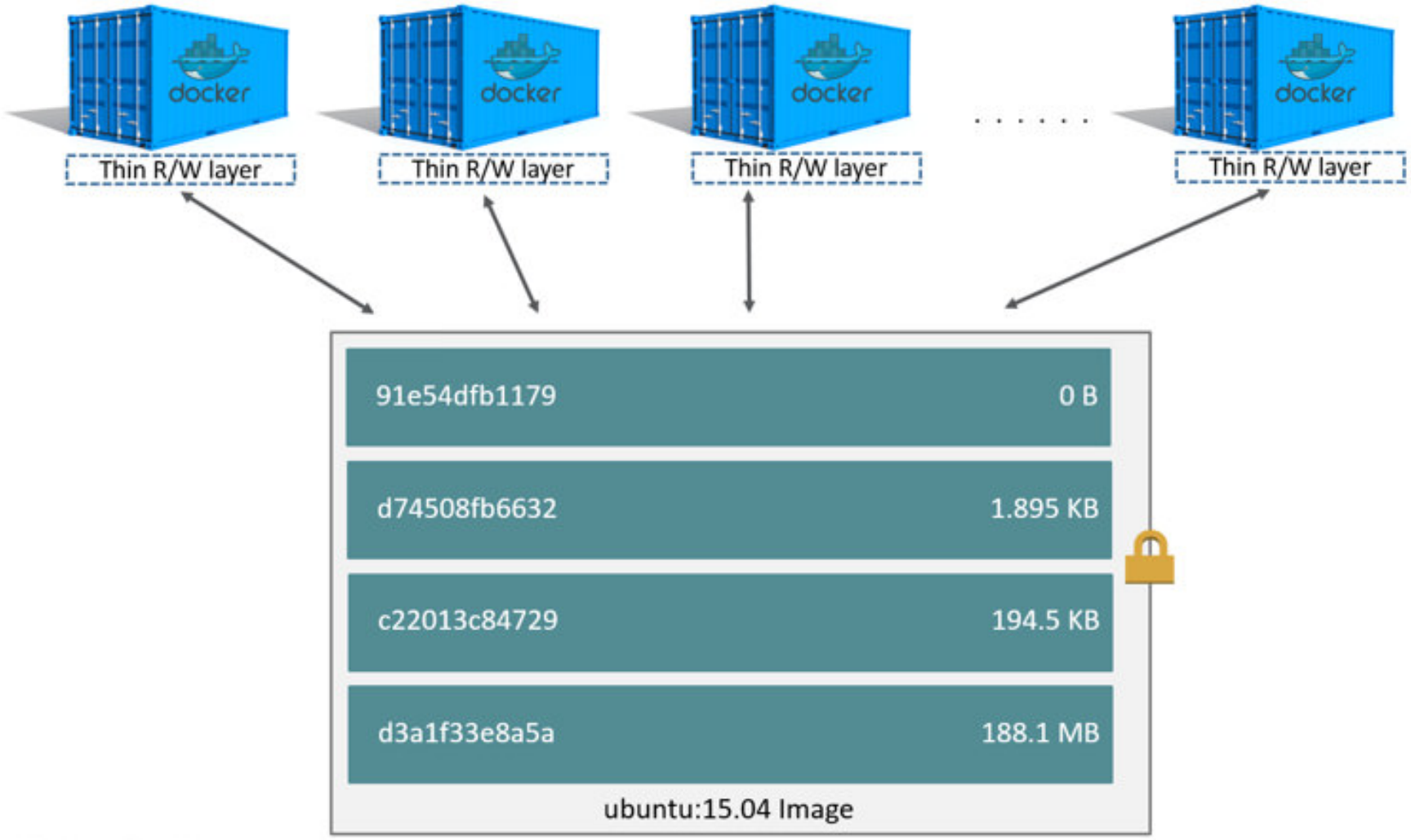


container -> docker image file -> release(by repo)

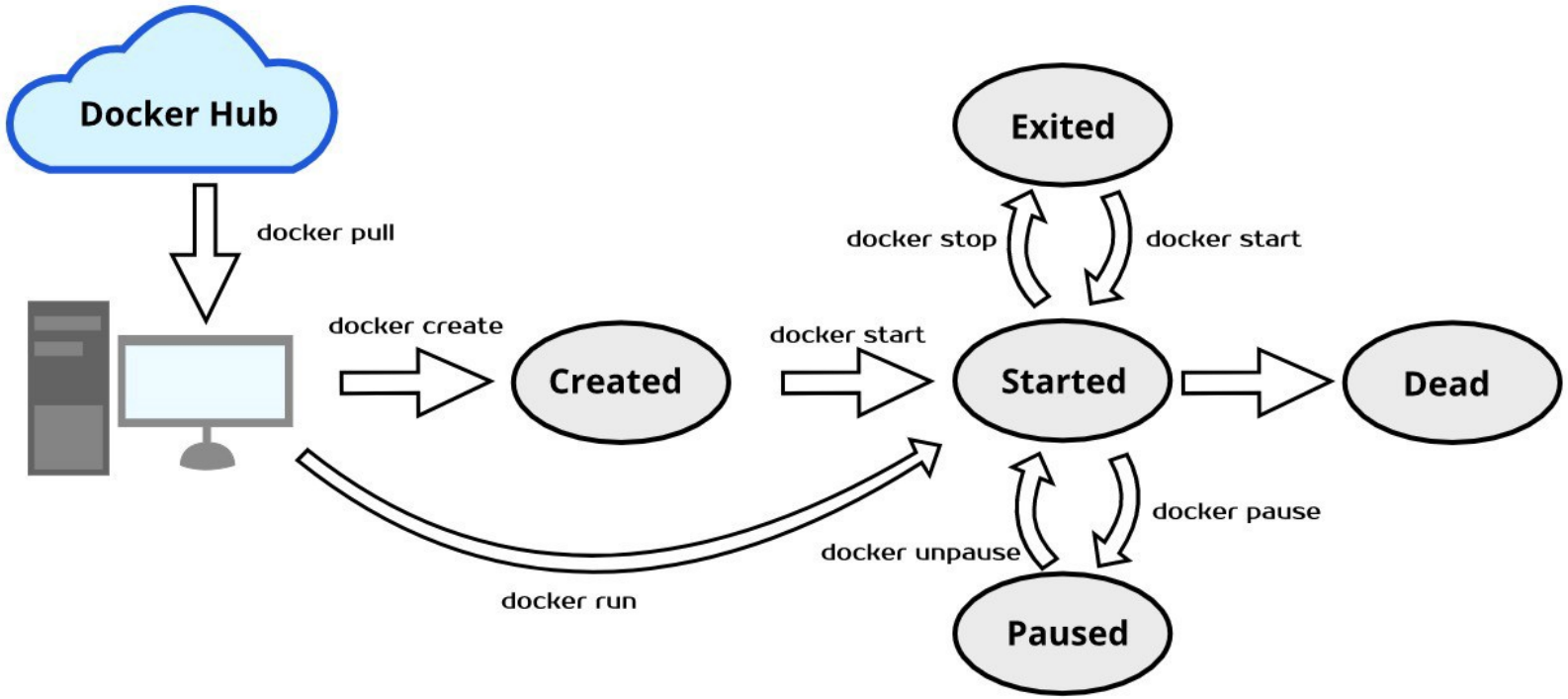
packaging

Delivery

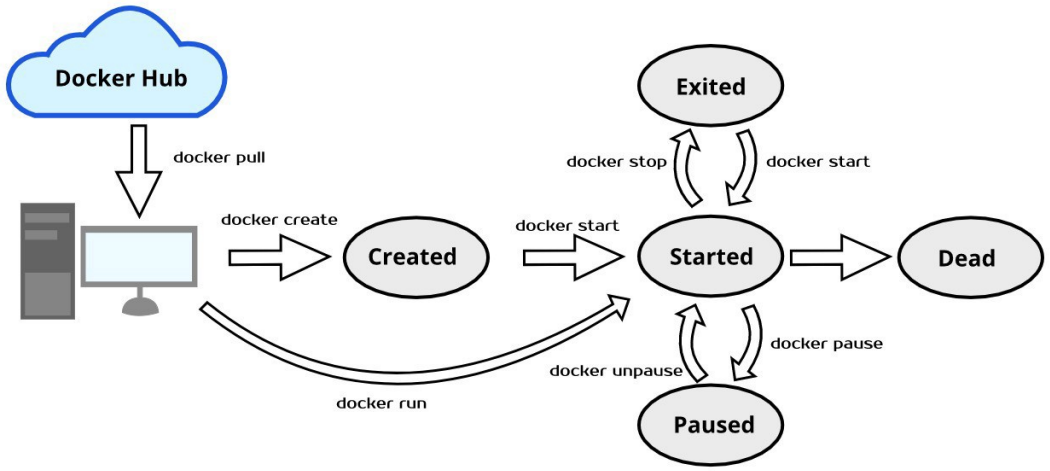
Docker container and layers

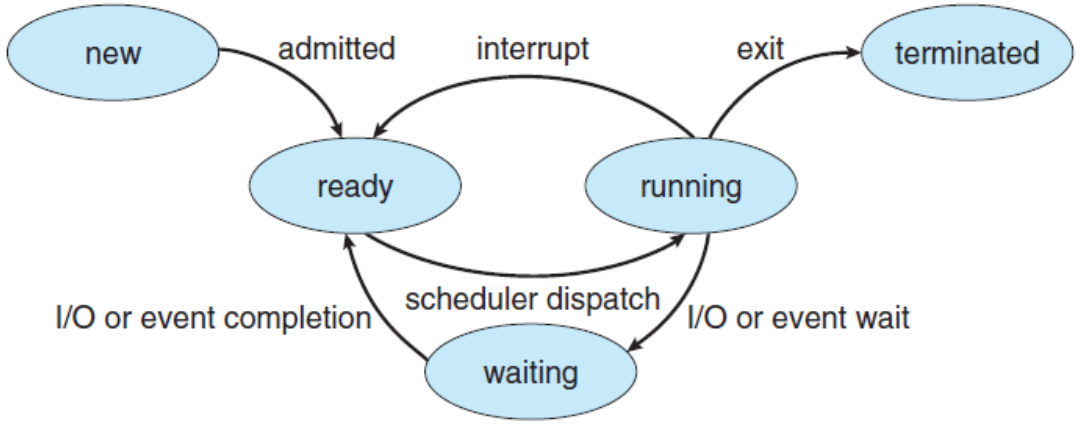


Docker container의 생명주기



Docker container의 생명주기





Docker container의 생명주기

Docker

```
koock@koock-HP-ProBook-4540s: ~
koock@koock-HP-ProBook-4540s:~$ sudo docker search tensorflow
[sudo] password for koock:
NAME                                  DESCRIPTION                                     STARS     OFFICIAL
  AUTOMATED
tensorflow/tensorflow                 Official docker images for deep learning f...   446
kblaster/tensorflow-jupyter           Dockerized Jupyter with tensorflow              39
  [OK]
satoshun/tensorflow-notebook          Jupyter with Tensorflow                         5
  [OK]
earthlab/r-tensorflow                 Docker container with R, RStudio, tensorfl...   4
  [OK]
chiefware/tensorflow-jupyter          tensorflow jupyter image to run under othe...   3
  [OK]
floydhub/tensorflow                   tensorflow                                      3
  [OK]
owits/tensorflow-alpine               Build tensorflow in alpine                      3
  [OK]
bottava/tensorflow                    TensorFlow images based on Alpine Linux.        2
  [OK]
eboraas/tensorflow                    TensorFlow with Jupyter Notebook, includin...   2
  [OK]
nytsk/tensorflow                      tensorflow image with matplotlib.pyplot.im...   2
  [OK]
ornew/tensorflow-android              Tensorflow for Android                          2
  [OK]
tomasbasham/tensorflow                tensorflow                                      1
  [OK]
wall72/tensorflow                     TensorFlow on Python 3.4                        1
  [OK]
clintmorgan/tensorflow                images to support using tensorflow              1
  [OK]
liqiang311/tensorflow                 tensorflow                                      0
  [OK]
acusensehub/tensorflow                Docker image for tensorflow                     0
  [OK]
mikebirdgeneau/r-tensorflow           RStudio and Tensorflow                          0
  [OK]
sunkeun/tensorflow-aws-gpu            Tensorflow AWS GPU                              0
  [OK]
hellgate75/tensorflow                 Docker Image for TensorFlow™, providing ...     0
  [OK]
bitnami/tensorflow-inception          Bitnami Docker Image for TensorFlow Inception   0
  [OK]
getwarped/tensorflow-notebook         OpenShift compatible version of Jupyter pr...   0
  [OK]
zuolan/tensorflow-go                  Tensorflow for Go                               0
  [OK]
djpetti/rpinets-tensorflow            Tensorflow container that is ready to be u...   0
  [OK]
mediadesignpractices/tensorflow       Tensorflow w/ CUDA (GPU) + extras               0
  [OK]
bitnami/tensorflow-serving            Bitnami Docker Image for TensorFlow Serving     0
  [OK]
```

Search Container image

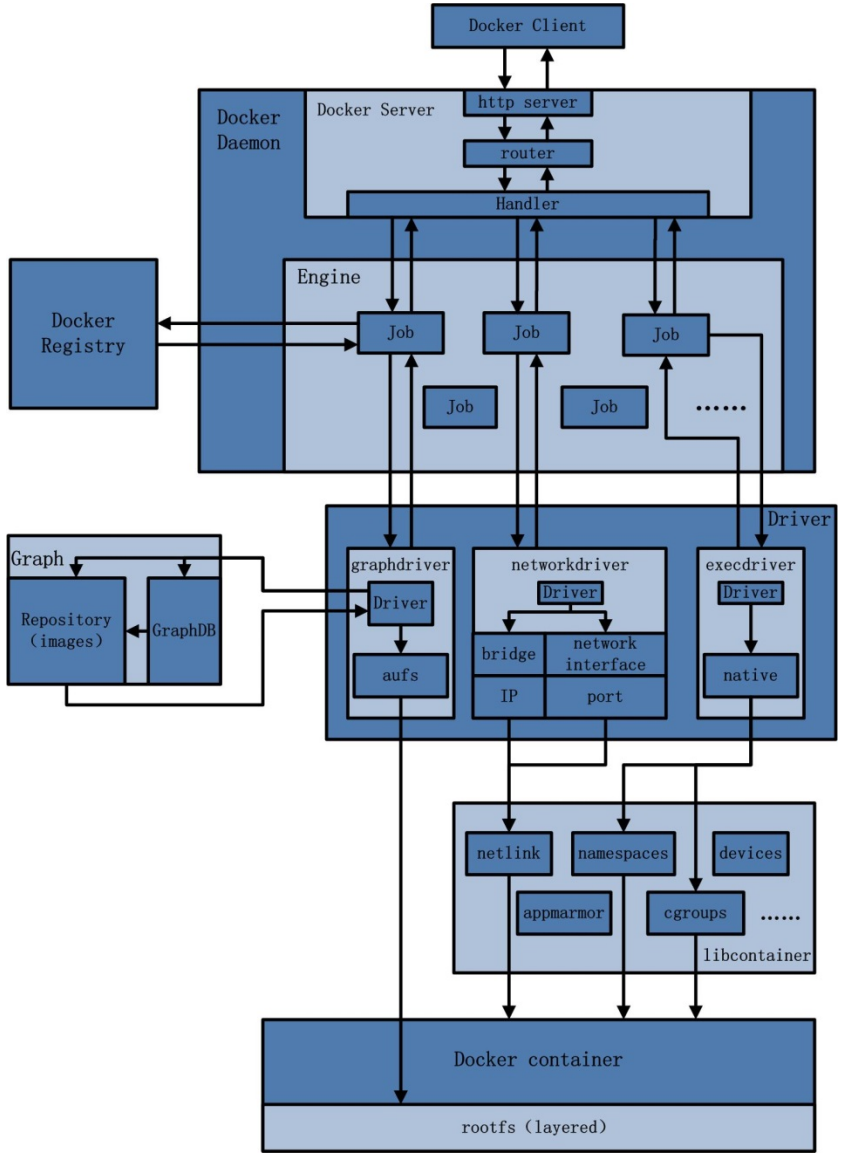
```
koock@koock-HP-ProBook-4540s:~$ sudo docker pull tensorflow/tensorflow:latest-devel-py3
latest-devel-py3: Pulling from tensorflow/tensorflow

75c416ea735c: Already exists
c6ff40b6d658: Already exists
a7050fc1f338: Already exists
f0ffb5cf6ba9: Already exists
be232718519c: Already exists
77becfc78153: Downloading 2.698 MB/284.5 MB
334ea417dbc0: Download complete
dfe6f2e9fb6e: Downloading 4.849 MB/143.8 MB
d3daccf79205: Waiting
c6d796463f9e: Waiting
b58a11e4044d: Waiting
87dfa06c91a7: Waiting
01cf86f86903: Waiting
1ce5cd7e3775: Waiting
2c31a119bac2: Waiting
```

Pull Container image

```
koock@koock-HP-ProBook-4540s:~$ sudo  docker run -it tensorflow/tensorflow:latest-devel-py3
root@abf5c5d467f2:~# python3
Python 3.5.2 (default, Nov 17 2016, 17:05:23)
[GCC 5.4.0 20160609] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> import tensorflow as tf
>>> message = tf.constant("hello world")
>>> sess = tf.Session()
2017-07-04 10:01:40.615107: W tensorflow/core/platform/cpu_feature_guard.cc:45]
The TensorFlow library wasn't compiled to use SSE4.1 instructions, but these are
 available on your machine and could speed up CPU computations.
2017-07-04 10:01:40.615241: W tensorflow/core/platform/cpu_feature_guard.cc:45]
The TensorFlow library wasn't compiled to use SSE4.2 instructions, but these are
 available on your machine and could speed up CPU computations.
2017-07-04 10:01:40.615297: W tensorflow/core/platform/cpu_feature_guard.cc:45]
The TensorFlow library wasn't compiled to use AVX instructions, but these are av
ailable on your machine and could speed up CPU computations.
>>> print(sess.run(message))
b'hello world'
>>>
```

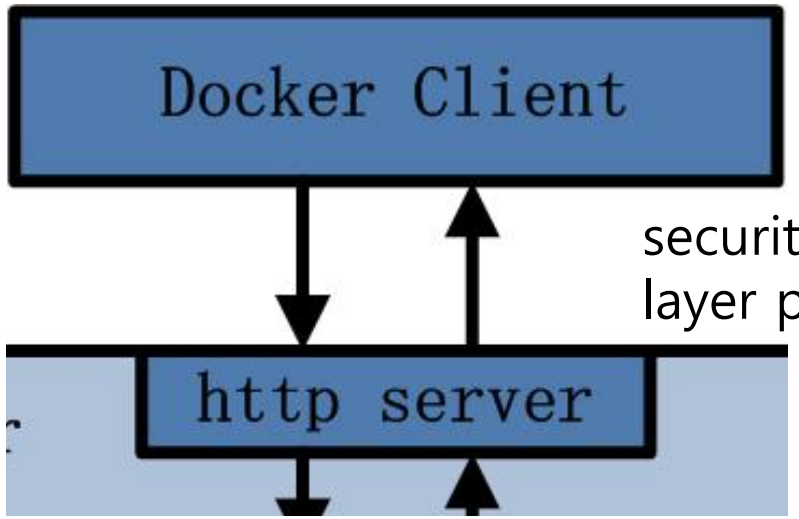
Run tensorflow on Container

Docker total architecture



Docker client

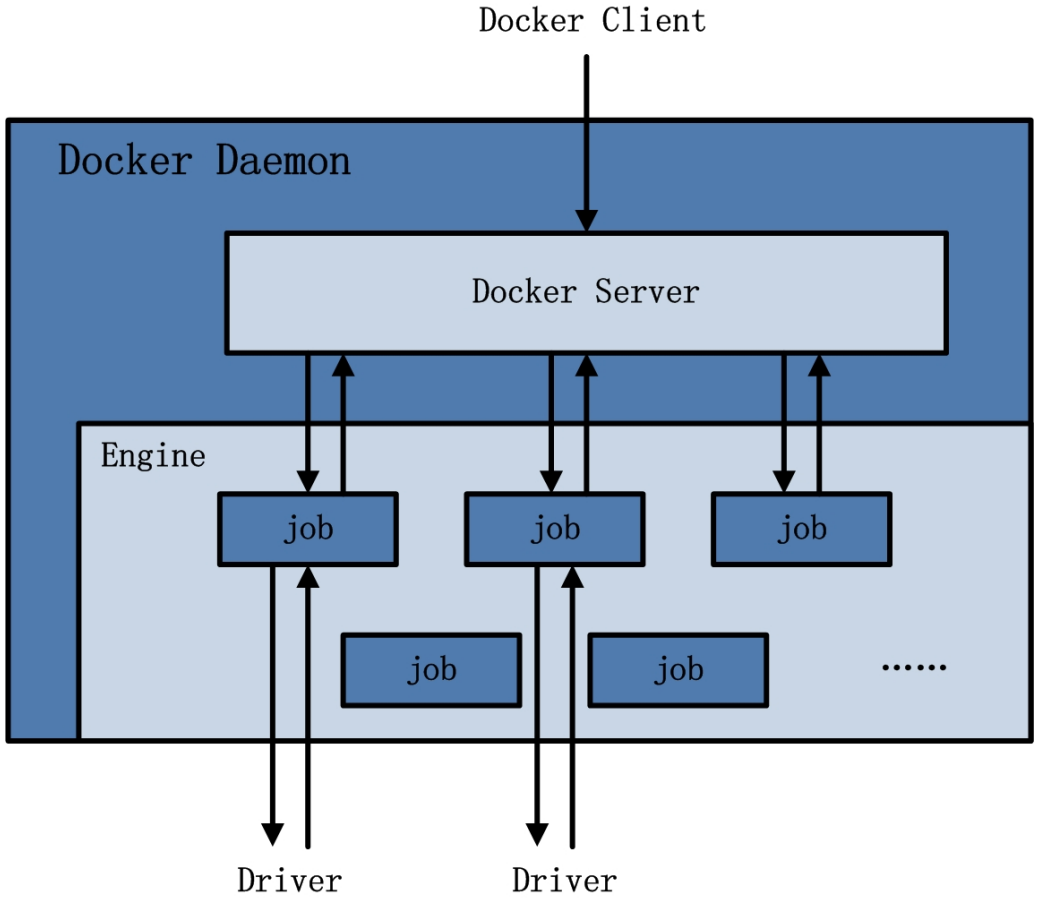
- tcp://host:port
- unix://path_to_socket
- Daemon and Docker



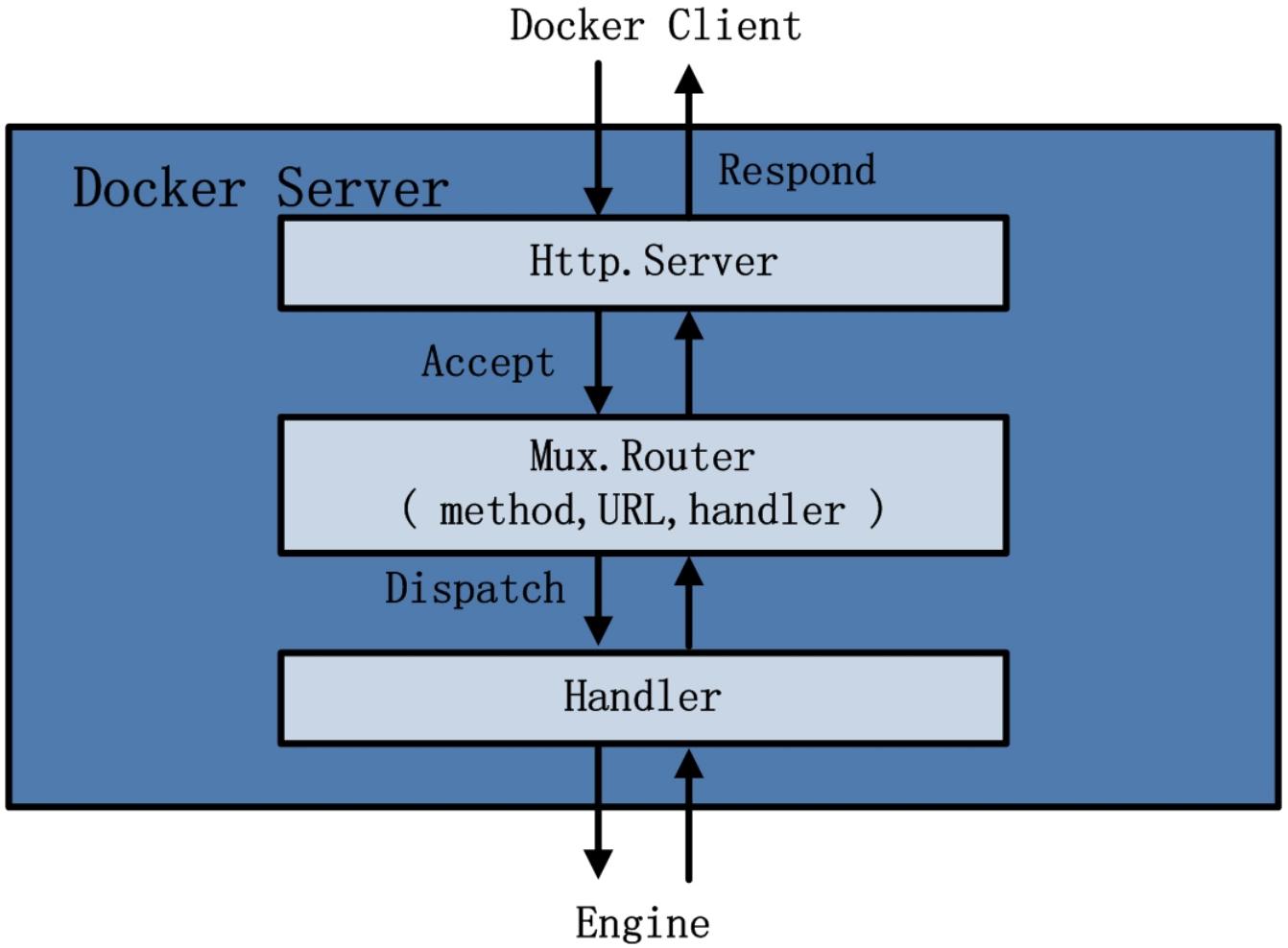
security transmission layer protocol (TLS)

Docker Daemon

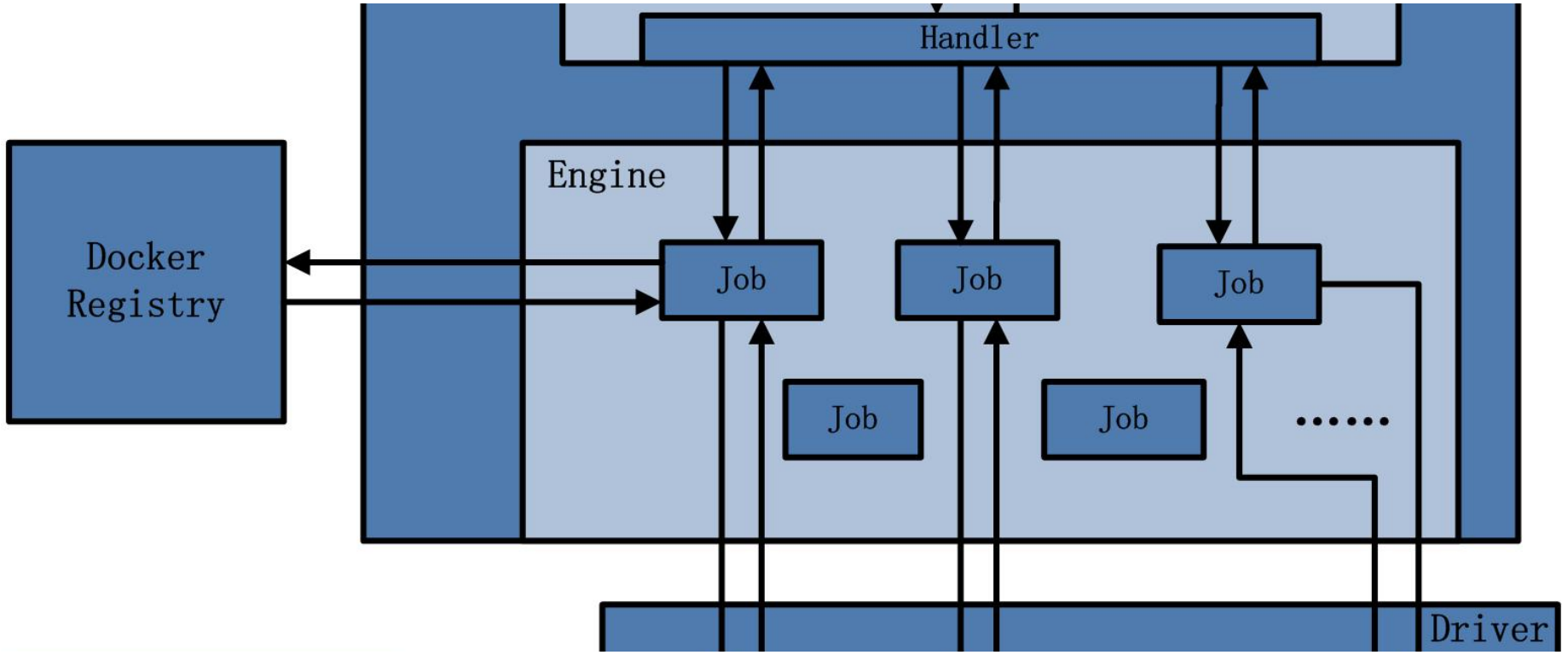
- To accept and handle the request Client Docker
- The process in the background to start a Server



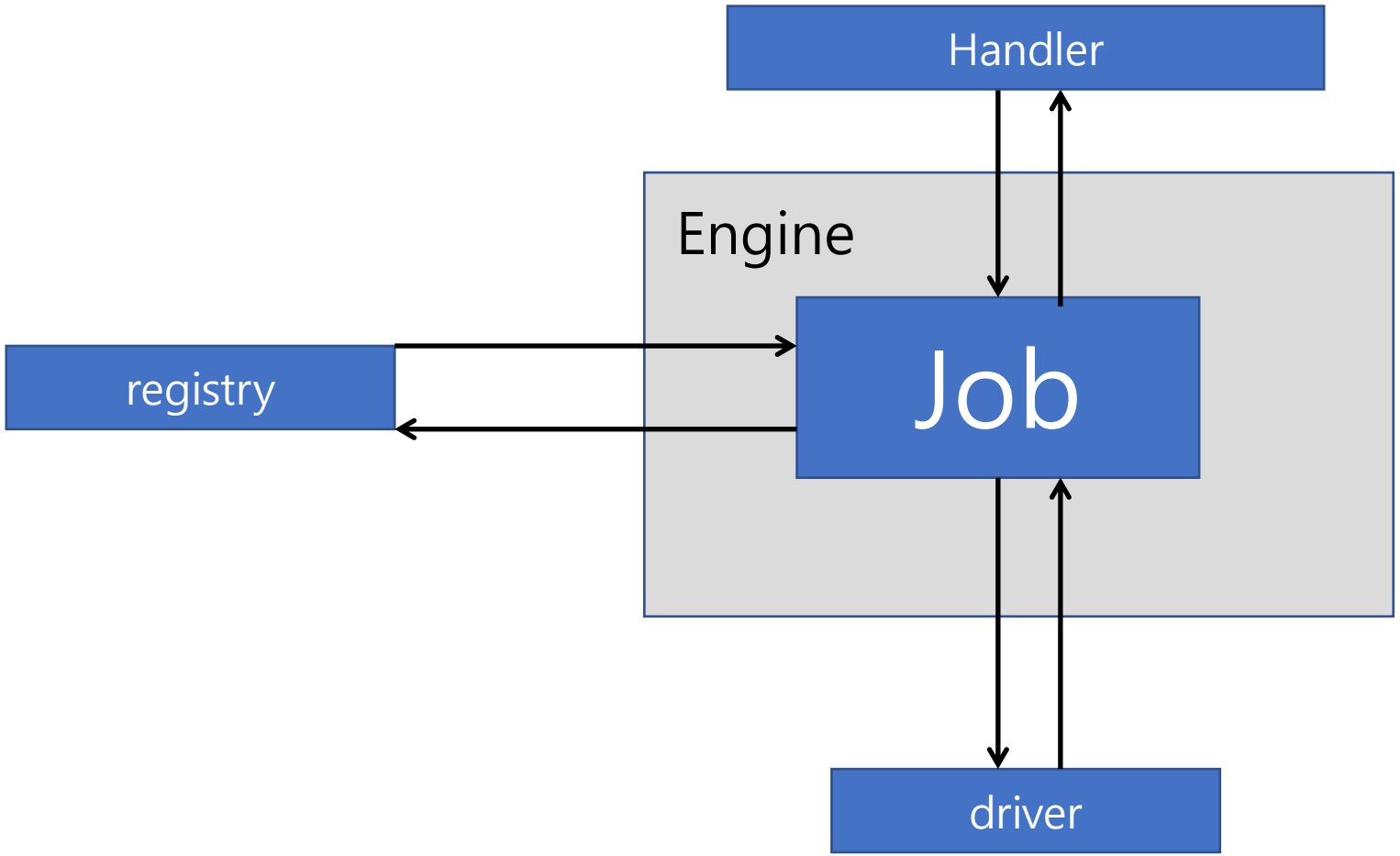
Docker Server in Daemon



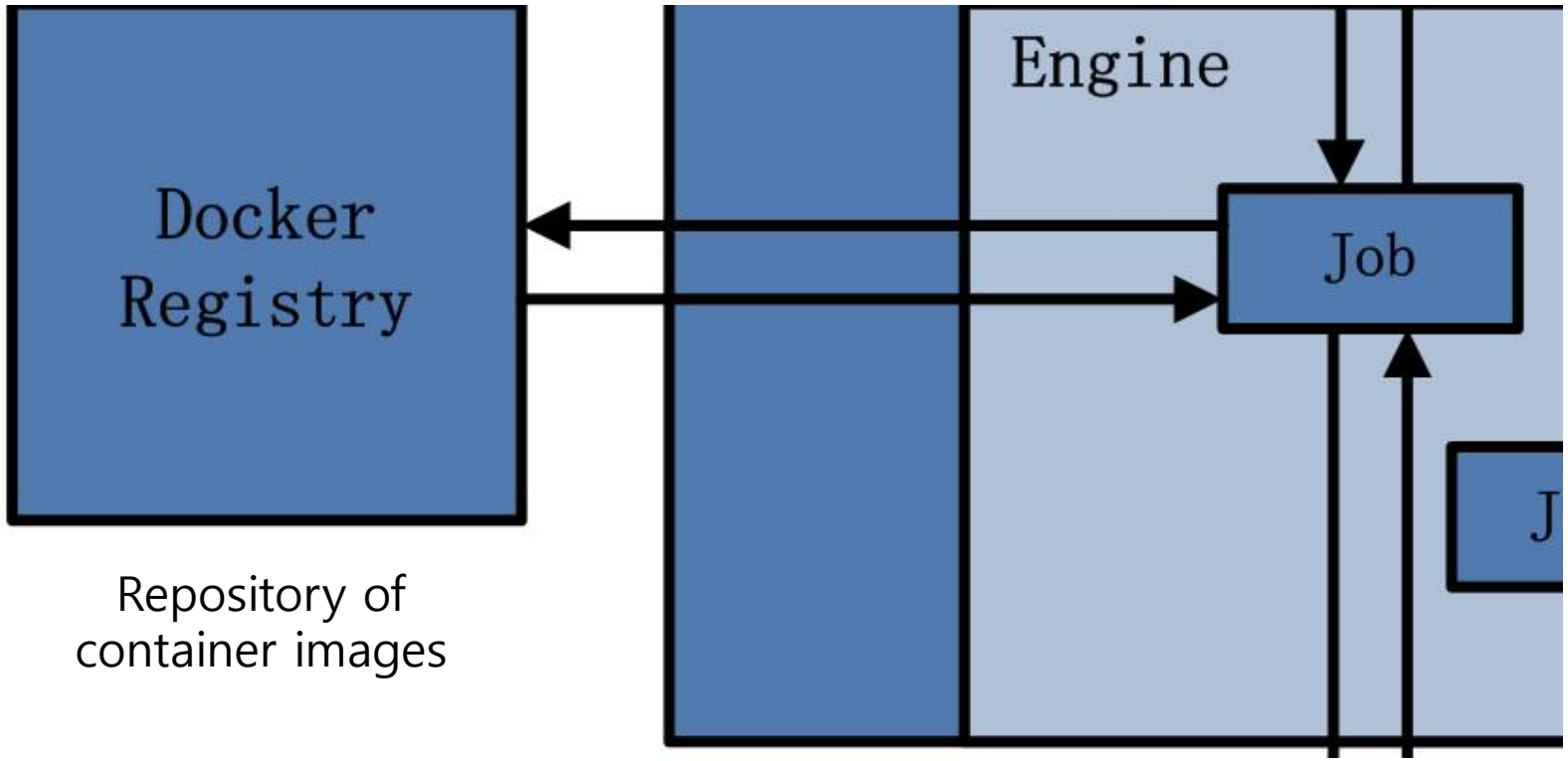
Docker Engine in Daemon



Job handled by engine



Docker Registry

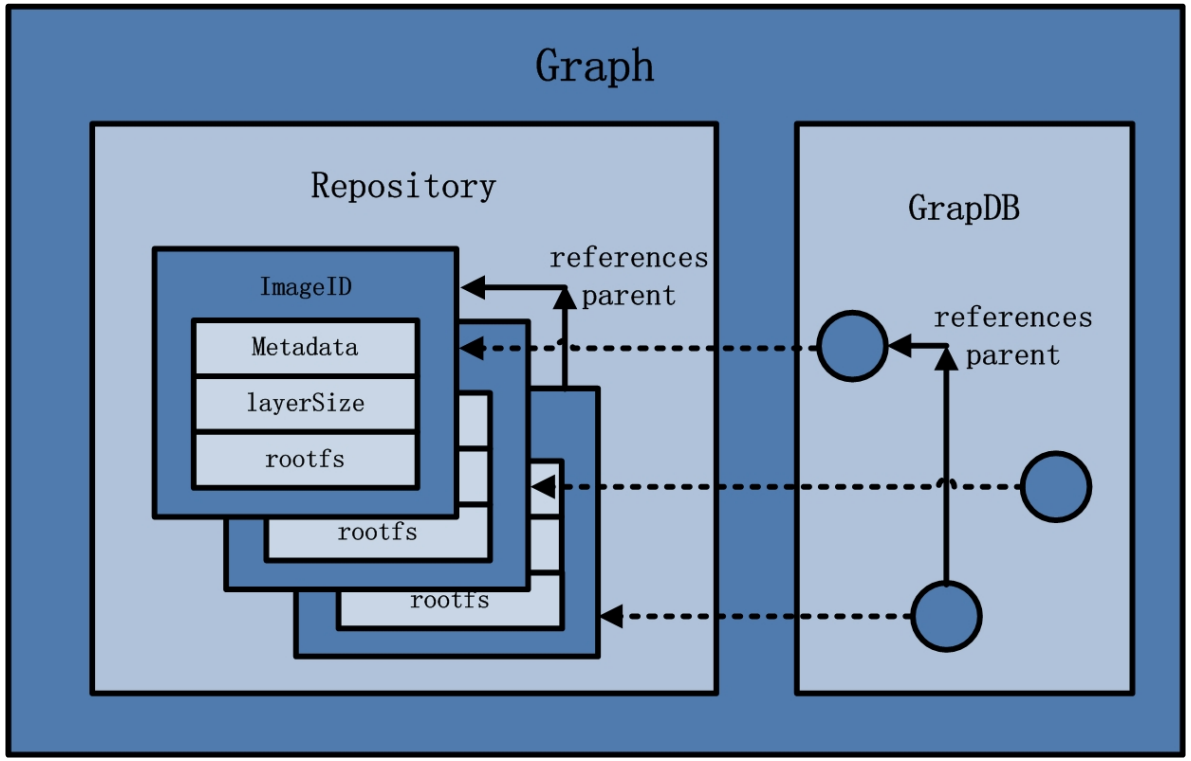


Repository of
container images

Graph

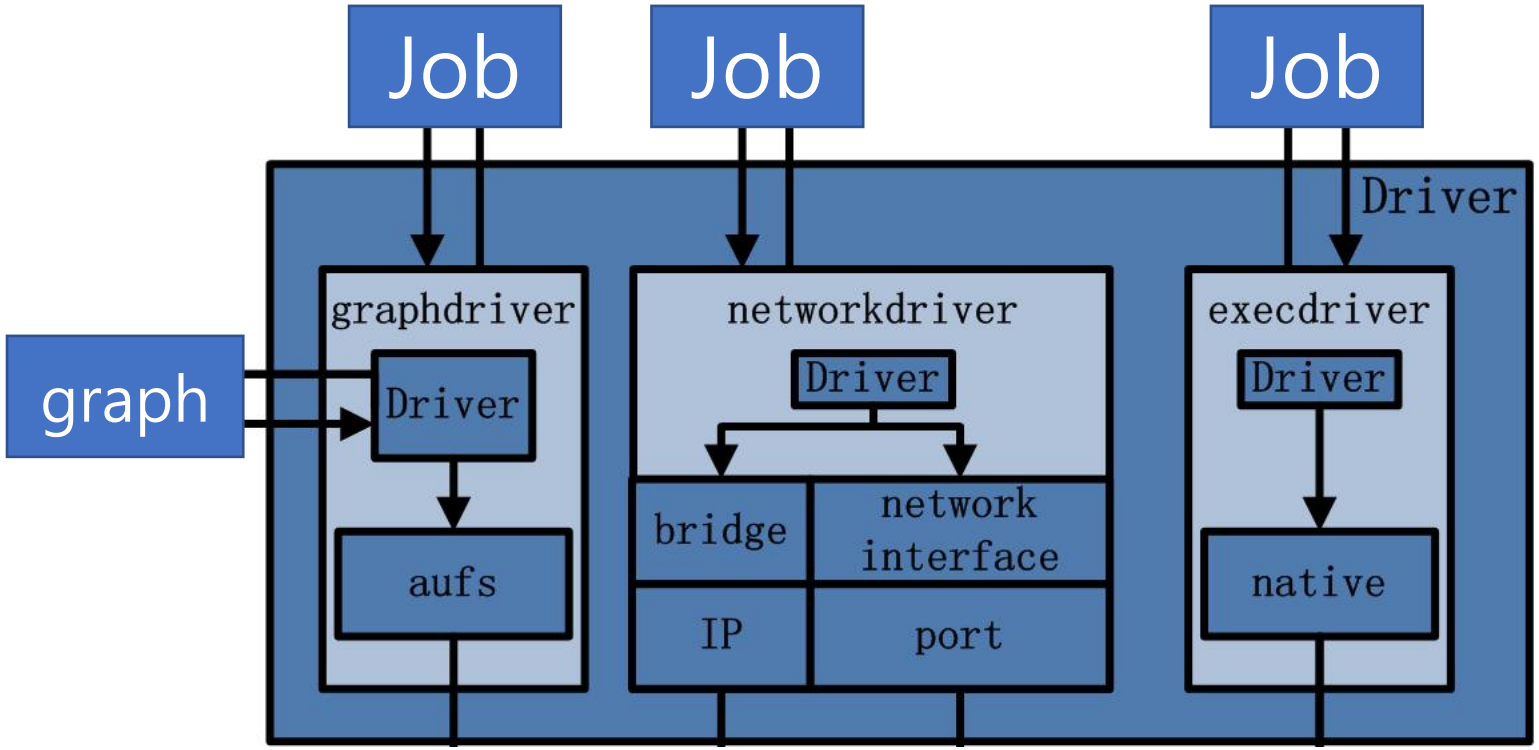
Graph plays the role of

- the storage in the Docker architecture
- the record of the relationship between the downloaded container mirror image

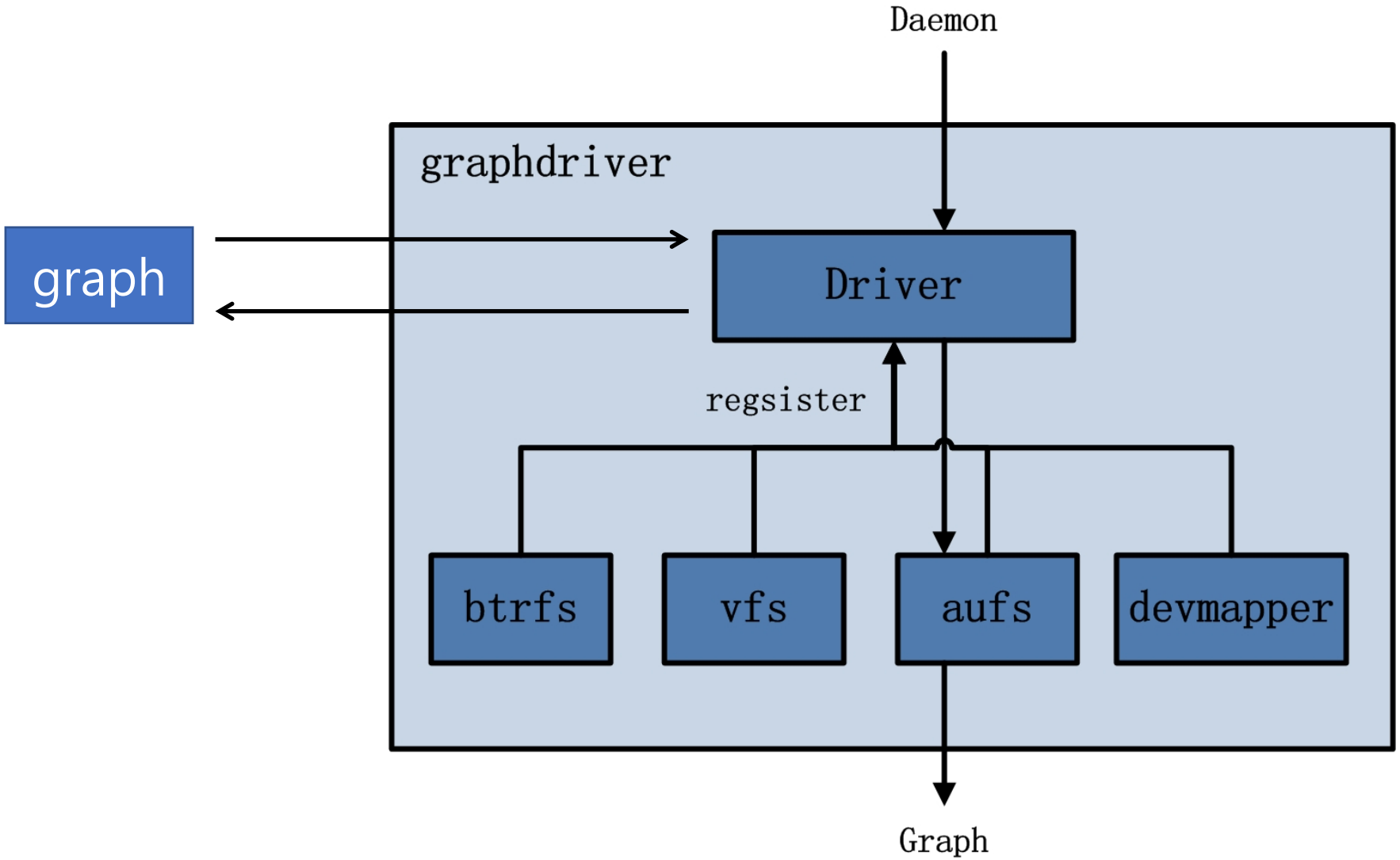


Driver

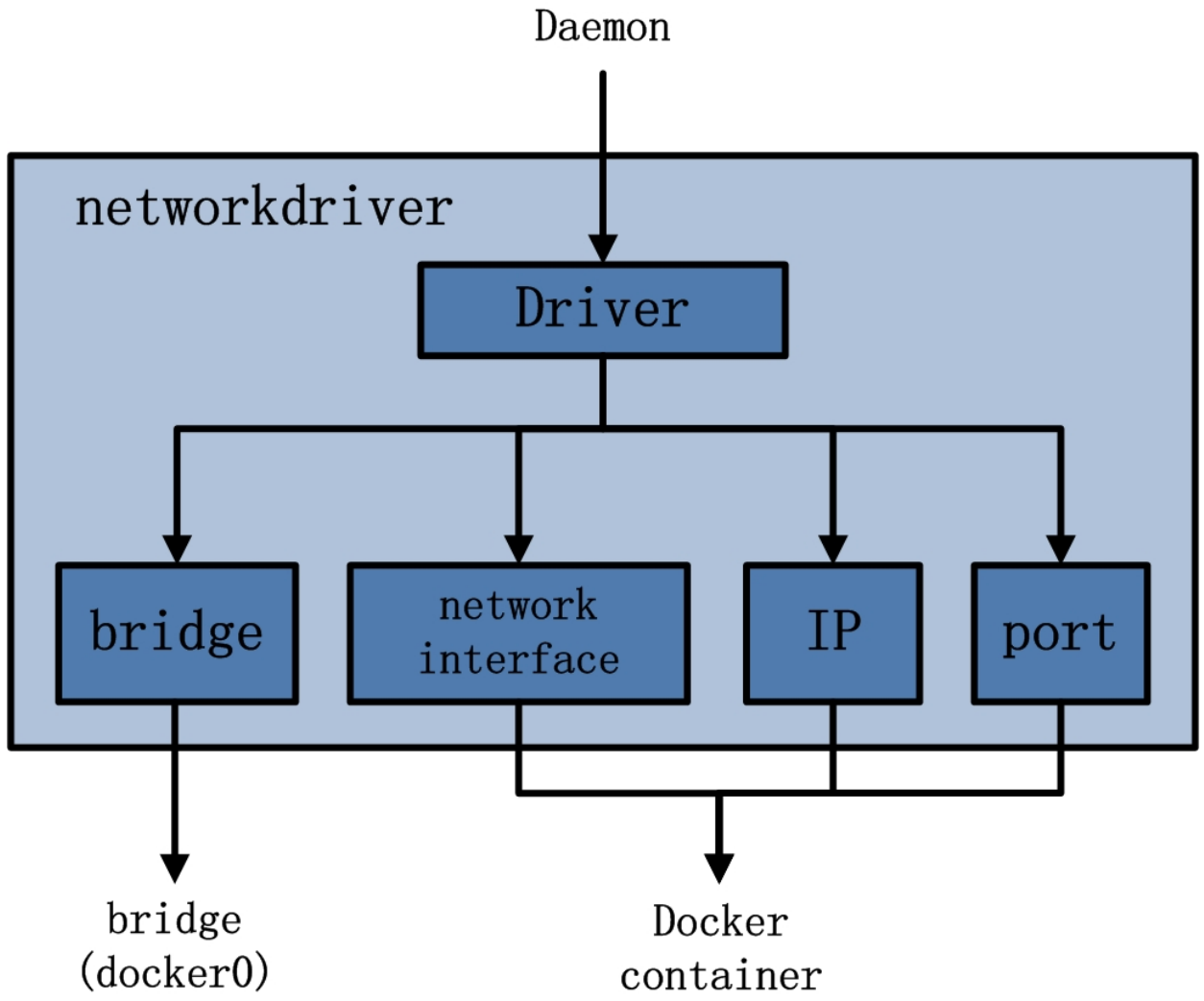
To separate the management of the Docker container from the Daemon Docker internal business logic



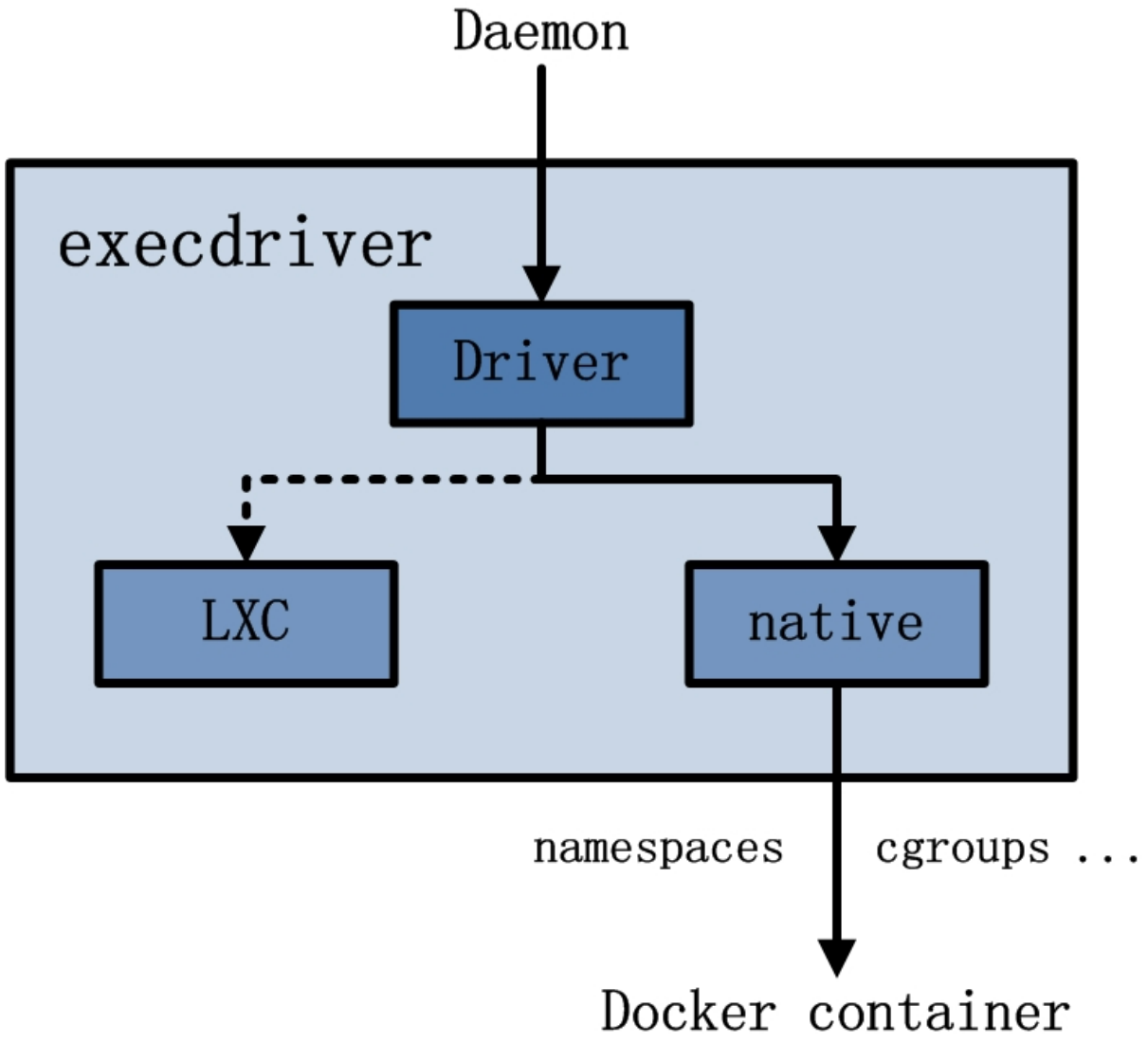
Driver : Graph driver



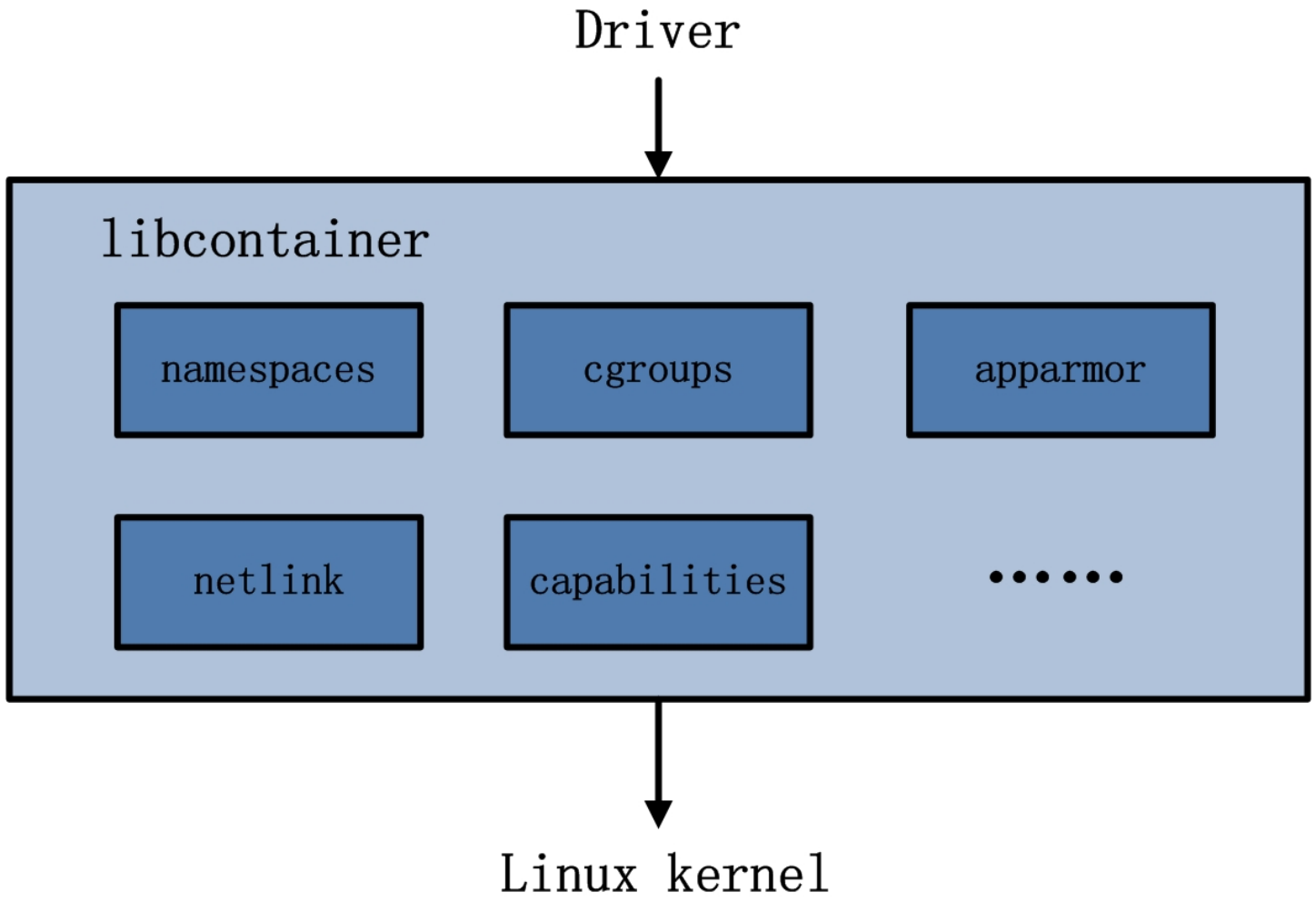
Driver：networkdriver



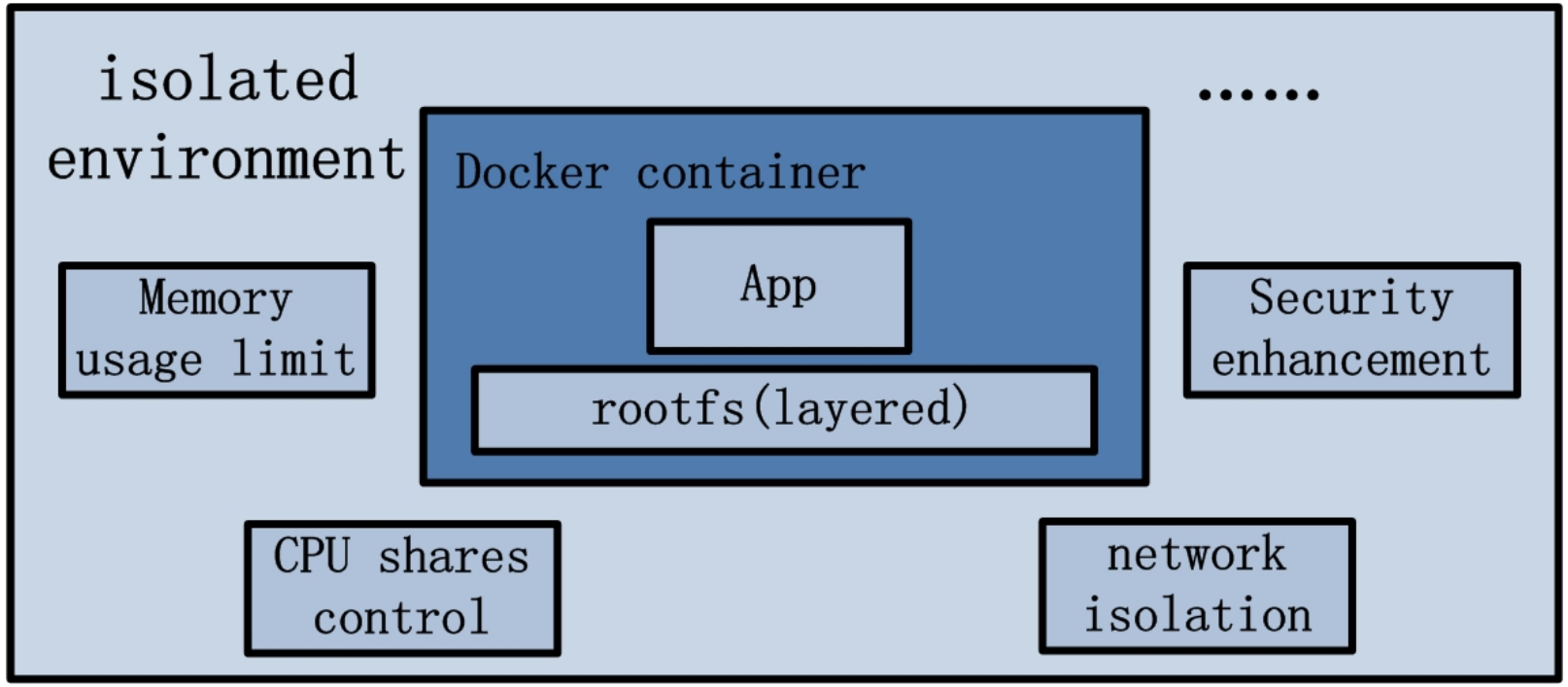
Driver : execdriver



libcontainer

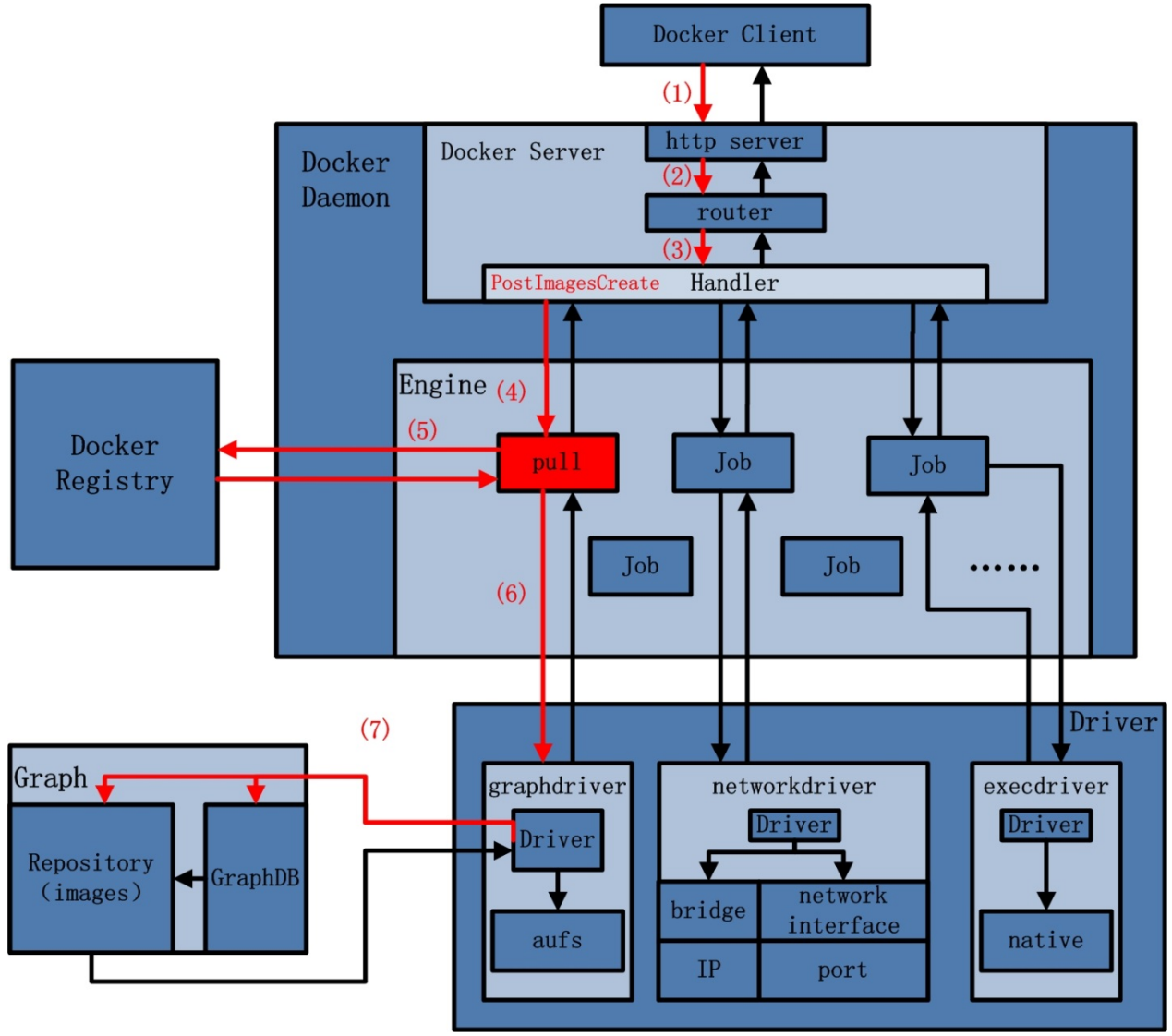


Container docker



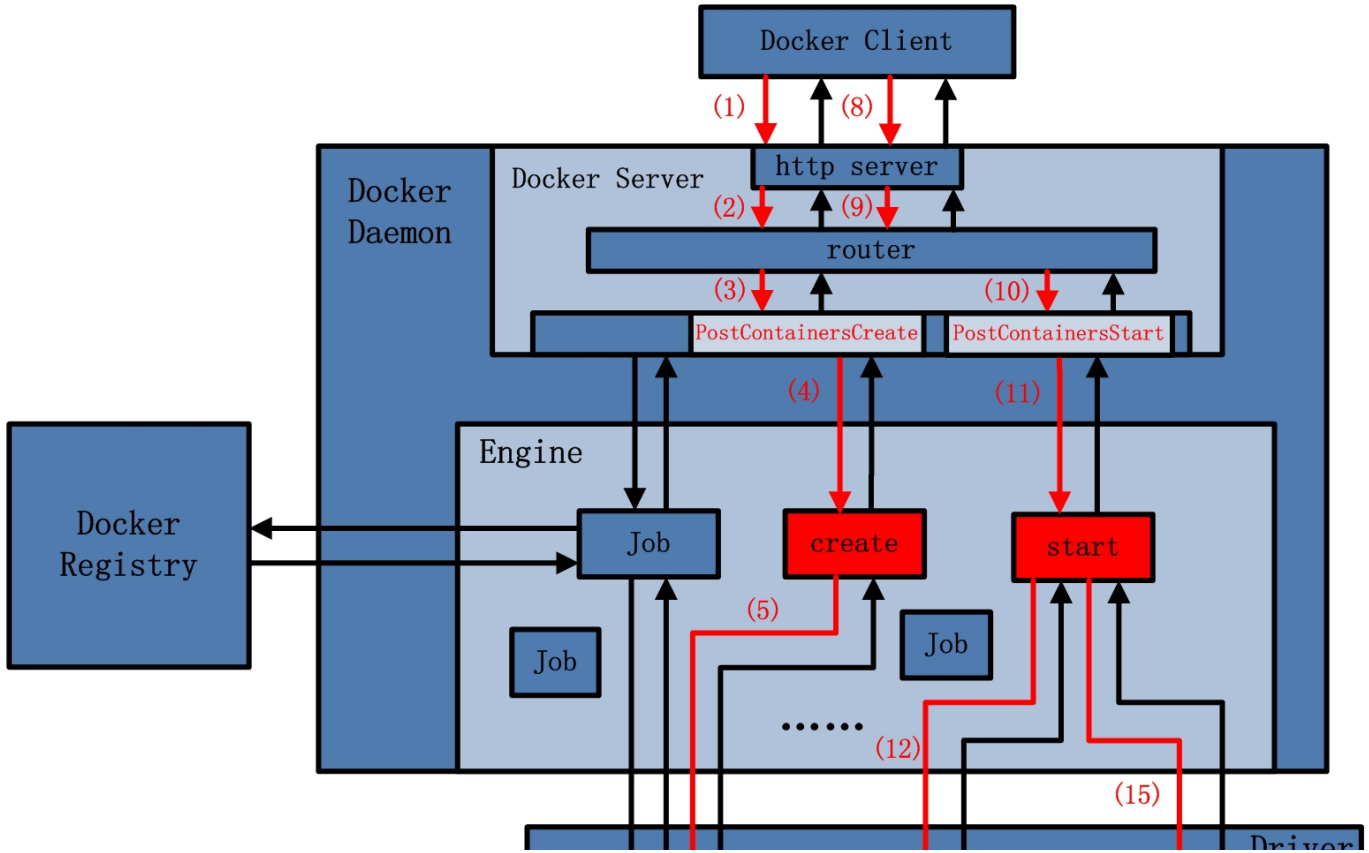
Case study : docker pull

```
koock@koock-HP-ProBook-4540s:~$ sudo docker pull tensorflow/tensorflow:latest-devel-py3
latest-devel-py3: Pulling from tensorflow/tensorflow

75c416ea735c: Already exists
c6ff40b6d658: Already exists
a7050fc1f338: Already exists
f0ffb5cf6ba9: Already exists
be232718519c: Already exists
77becfc78153: Downloading 2.698 MB/284.5 MB
334ea417dbc0: Download complete
dfe6f2e9fb6e: Downloading 4.849 MB/143.8 MB
d3daccf79205: Waiting
c6d796463f9e: Waiting
b58a11e4044d: Waiting
87dfa06c91a7: Waiting
01cf86f86903: Waiting
1ce5cd7e3775: Waiting
2c31a119bac2: Waiting
```

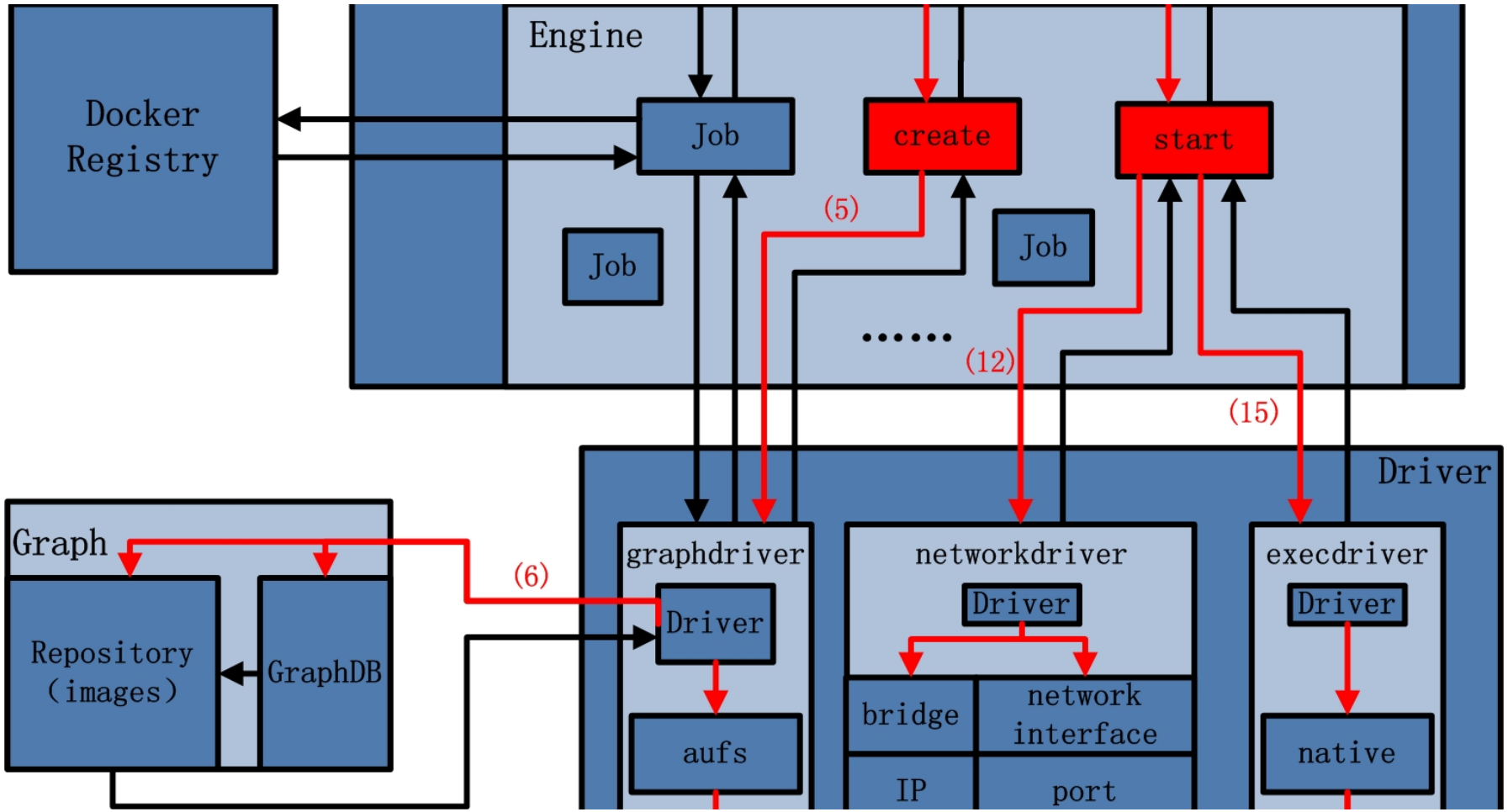


Case study : docker run

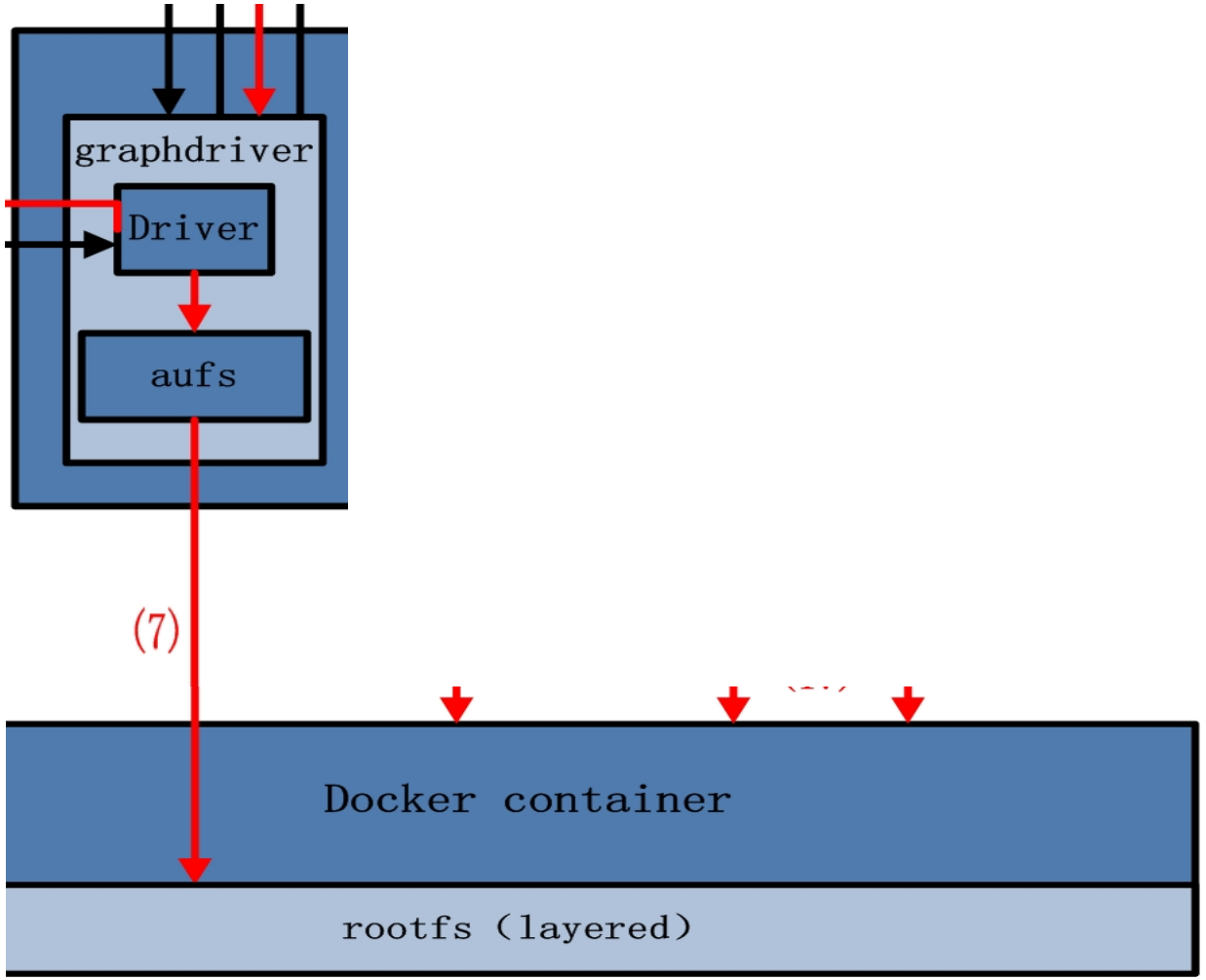
```
koock@koock-HP-ProBook-4540s:~$ sudo  docker run -it tensorflow/tensorflow:lates
t-devel-py3
root@abf5c5d467f2:~# python3
Python 3.5.2 (default, Nov 17 2016, 17:05:23)
[GCC 5.4.0 20160609] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> import tensorflow as tf
>>> message = tf.constant("hello world")
>>> sess = tf.Session()
2017-07-04 10:01:40.615107: W tensorflow/core/platform/cpu_feature_guard.cc:45]
The TensorFlow library wasn't compiled to use SSE4.1 instructions, but these are
 available on your machine and could speed up CPU computations.
2017-07-04 10:01:40.615241: W tensorflow/core/platform/cpu_feature_guard.cc:45]
The TensorFlow library wasn't compiled to use SSE4.2 instructions, but these are
 available on your machine and could speed up CPU computations.
2017-07-04 10:01:40.615297: W tensorflow/core/platform/cpu_feature_guard.cc:45]
The TensorFlow library wasn't compiled to use AVX instructions, but these are av
ailable on your machine and could speed up CPU computations.
>>> print(sess.run(message))
b'hello world'
>>>
```



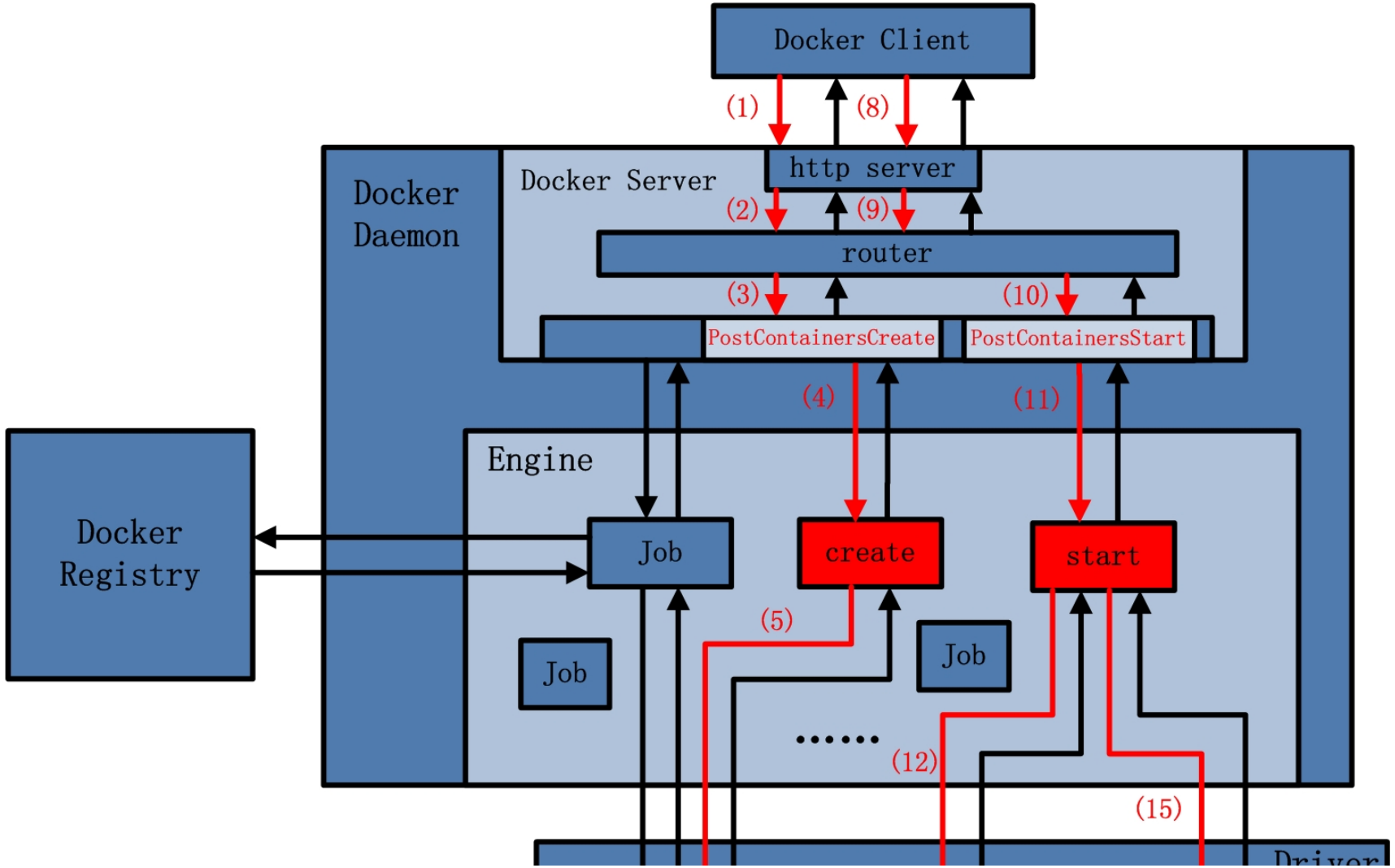
Case study : docker run



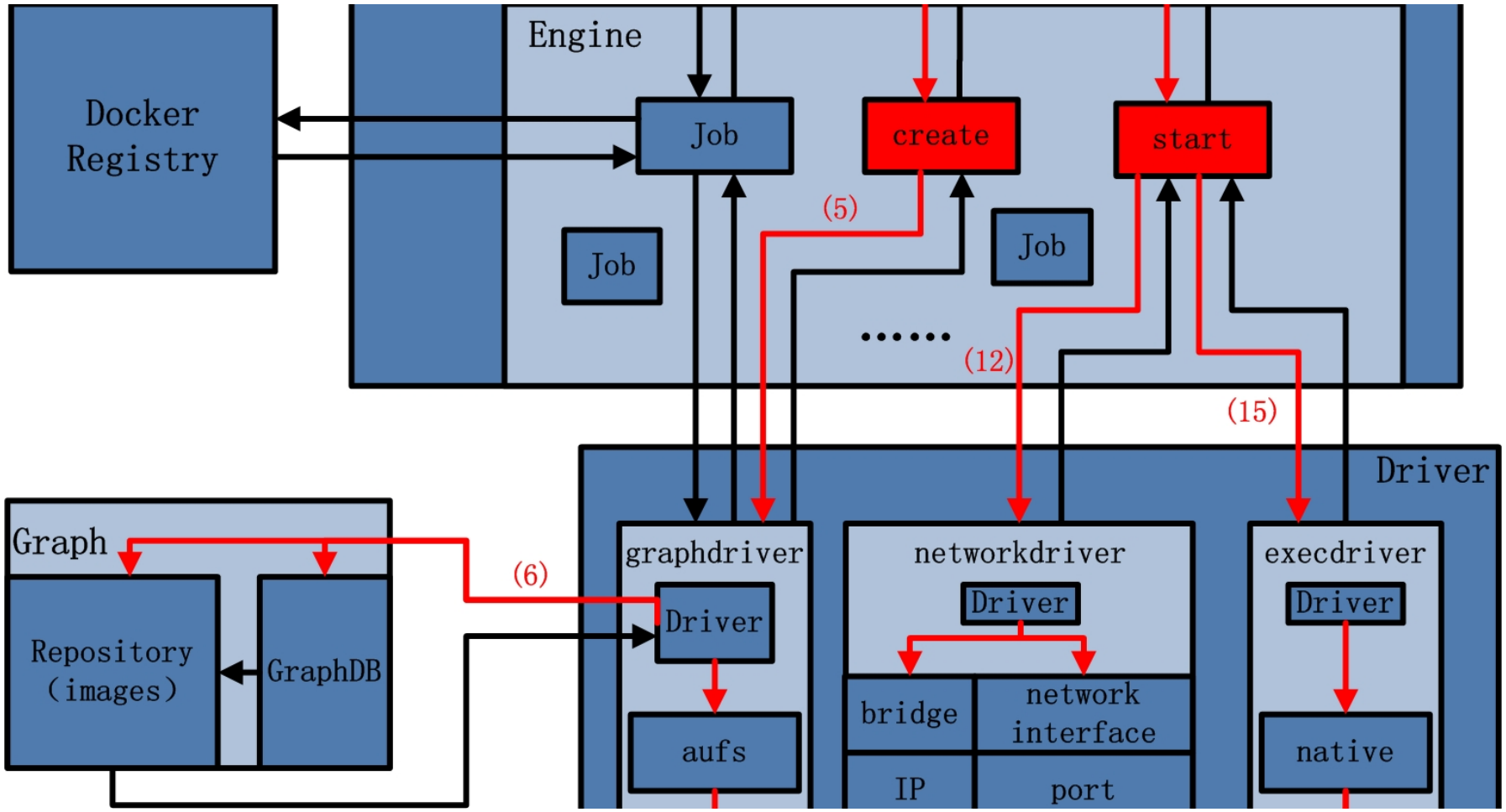
Case study : docker run



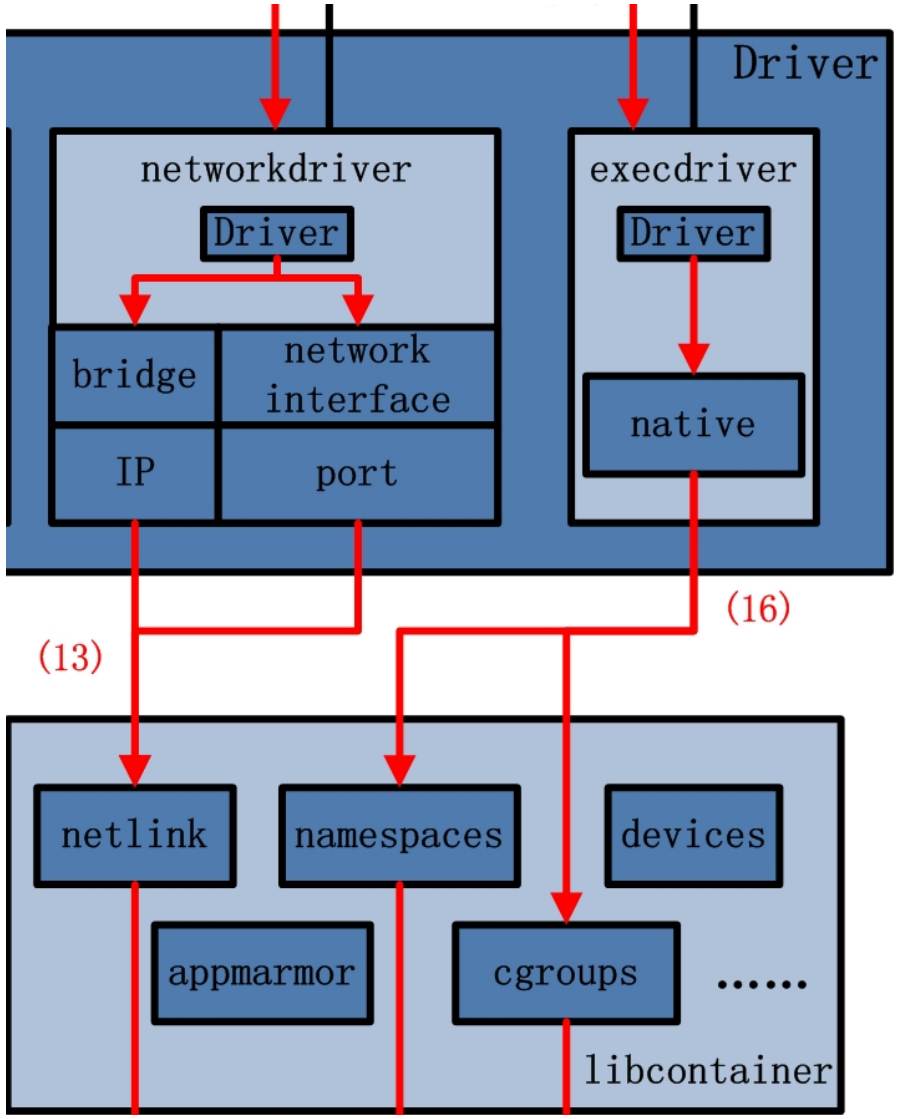
Case study : docker run



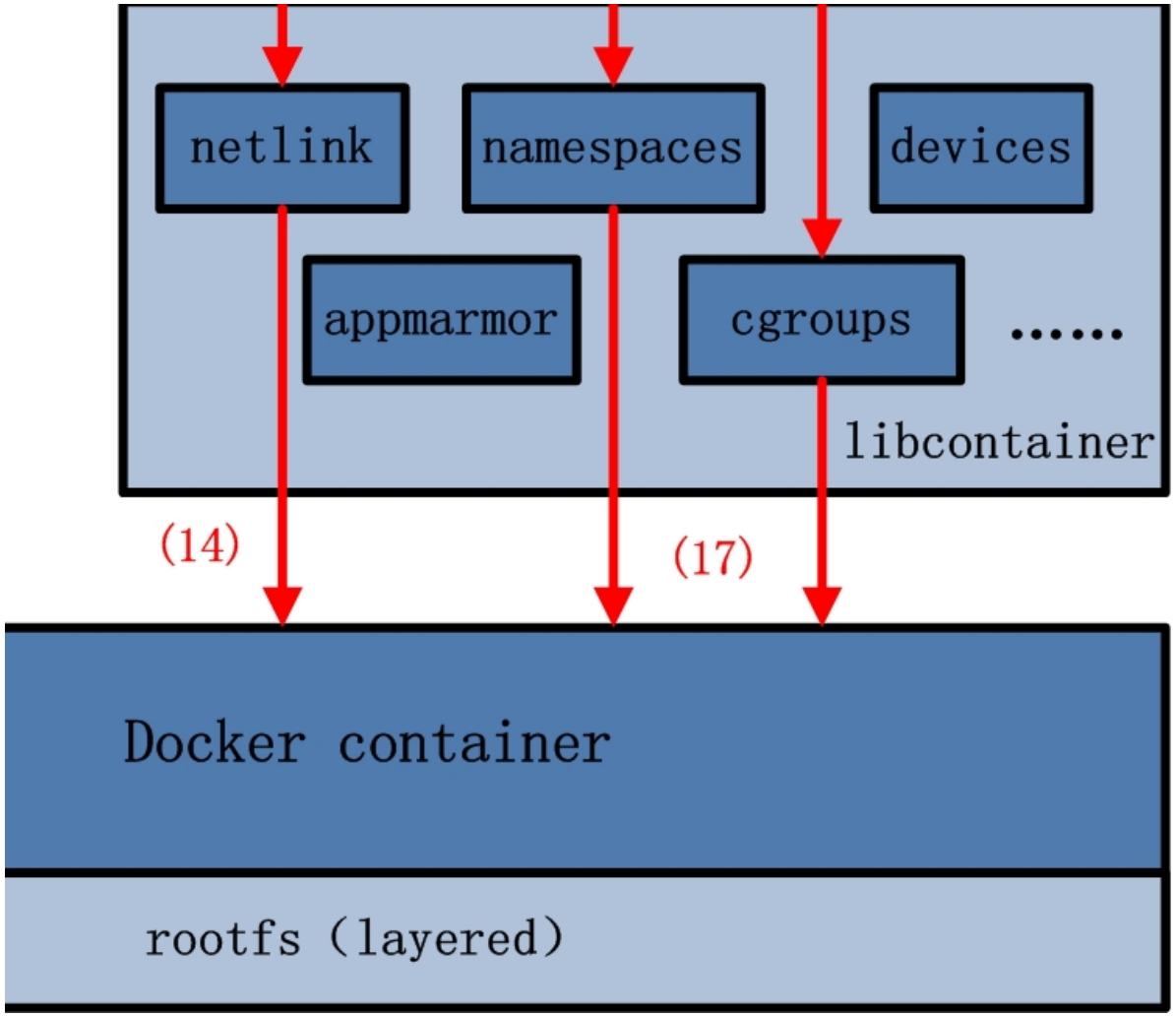
Case study : docker run



Case study : docker run



Case study : docker run



dockerfile

- FROM
- LABEL
- ADD
- COPY
- VOLUME
- WORKDIR
- USER
- ENV
- ARG
- EXPOSE
- RUN
- CMD
- ENTRYPOINT
- ONBUILD

dockerfile

- FROM
- LABEL
- ADD
- COPY
- VOLUME
- WORKDIR
- USER
- ENV
- ARG
- EXPOSE
- RUN
- CMD
- ENTRYPOINT
- ONBUILD

빌드할 이미지의 base image

```
FROM [--platform=<platform>] <image> [AS <name>]
```

Or

```
FROM [--platform=<platform>] <image>[:<tag>] [AS <name>]
```

Or

```
FROM [--platform=<platform>] <image>[@<digest>] [AS <name>]
```

dockerfile

- FROM
- LABEL
- ADD
- COPY
- VOLUME
- WORKDIR
- USER
- ENV
- ARG
- EXPOSE
- RUN
- CMD
- ENTRYPOINT
- ONBUILD

빌드할 이미지를 설명하기 위한 라벨링
만든사람, 버전, 설명 등을 넣음

```
LABEL "com.example.vendor"="ACME Incorporated"
LABEL com.example.label-with-value="foo"
LABEL version="1.0"
LABEL description="This text illustrates \
that label-values can span multiple lines."
```

dockerfile

- FROM
- LABEL
- ADD
- COPY
- VOLUME
- WORKDIR
- USER
- ENV
- ARG
- EXPOSE
- RUN
- CMD
- ENTRYPOINT
- ONBUILD

local의 파일 또는 폴더를 container의 file system에 추가
(속칭 docker container에 말아넣는다(?)는게 이것

COPY는 말그대로 복사만 진행

```
COPY test.txt relativeDir/
```

ADD는 tar를 자동 추출하고 원격 URL 지원

```
ADD test.txt relativeDir/
```

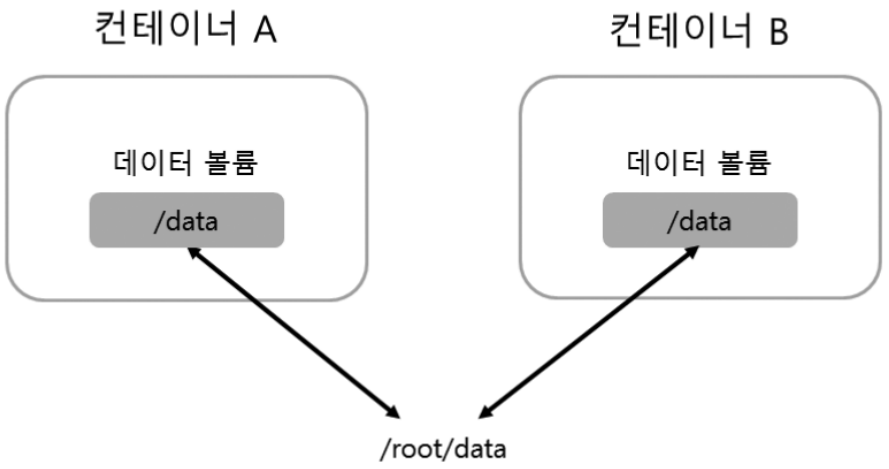
dockerfile

- FROM
- LABEL
- ADD
- COPY
- VOLUME
- WORKDIR
- USER
- ENV
- ARG
- EXPOSE
- RUN
- CMD
- ENTRYPOINT
- ONBUILD

host의 특정경로를 container에 마운트해서 volume을 연결

```
VOLUME ["/data"]
```

```
$ sudo docker run -i -t --name hello-volume -v /data ubuntu /bin/bash
root@019265a6920f:/# cd /data
root@019265a6920f:/data# touch hello
root@019265a6920f:/data# exit
```



dockerfile

- FROM
- LABEL
- ADD
- COPY
- VOLUME
- WORKDIR
- USER
- ENV
- ARG
- EXPOSE
- RUN
- CMD
- ENTRYPOINT
- ONBUILD

빌드를 하는 container의 현재 위치를 지정

RUN cd /path/to/workdir 와 같은 명령

```
WORKDIR /path/to/workdir
```

```
ENV DIRPATH=/path
WORKDIR $DIRPATH/$DIRNAME
RUN pwd
```

dockerfile

- FROM
- LABEL
- ADD
- COPY
- VOLUME
- WORKDIR
- USER
- ENV
- ARG
- EXPOSE
- RUN
- CMD
- ENTRYPOINT
- ONBUILD

현재 작업 유저 설정

```
FROM microsoft/windowsservercore
# Create Windows user in the container
RUN net user /add patrick
# Set it for subsequent commands
USER patrick
```

dockerfile

- FROM
- LABEL
- ADD
- COPY
- VOLUME
- WORKDIR
- USER
- ENV
- ARG
- EXPOSE
- RUN
- CMD
- ENTRYPOINT
- ONBUILD

빌드하거나 실행할 때 필요한 변수

ENV는 컨테이너 내부에서 필요한 환경변수로 빌드할때와 컨테이너를 실행할 때 모두 사용

```
ENV MY_NAME="John Doe"
ENV MY_DOG=Rex\ The\ Dog
ENV MY_CAT=fluffy
```

ARG는 빌드할 때 외부에서 입력받는 변수로 빌드 때만 사용

```
FROM busybox
ARG user1
ARG buildno
# ...
```

dockerfile

- FROM
- LABEL
- ADD
- COPY
- VOLUME
- WORKDIR
- USER
- ENV
- ARG
- EXPOSE
- RUN
- CMD
- ENTRYPOINT
- ONBUILD

컨테이너 내부의 port와 외부 port의 포트포워딩

```
EXPOSE 80/tcp
EXPOSE 80/udp
```

```
FROM debian:stable
RUN apt-get update && apt-get install -y --force-yes apache2
EXPOSE 80 443
```

```
docker run -p 80:80/tcp -p 80:80/udp ...
```

dockerfile

- FROM
- LABEL
- ADD
- COPY
- VOLUME
- WORKDIR
- USER
- ENV
- ARG
- EXPOSE
- RUN
- CMD
- ENTRYPOINT
- ONBUILD

컨테이너에서 명령으로 실행하는 역할

RUN은 컨테이너 빌드 시 차례로 수행, 여러 개 가능

```
FROM microsoft/nanoserver
COPY testfile.txt c:\\
RUN dir c:\
```

```
PS C:\John> docker build -t cmd .
Sending build context to Docker daemon 3.072 kB
Step 1/2 : FROM microsoft/nanoserver
 ---> 22738ff49c6d
Step 2/2 : COPY testfile.txt c:\RUN dir c:
GetFileAttributesEx c:RUN: The system cannot find the file specified.
PS C:\John>
```

CMD는 컨테이너 실행 시 실행, 하나만 가능
docker run 명령 시 인자로 대체, 실행인자 있을 시 실행안함

```
FROM ubuntu
CMD ["/usr/bin/wc","--help"]
```

ENTRYPOINT는 컨테이너 실행 시 실행, 하나만 가능
docker run 명령 시 인자로 대체 불가능, 무조건 실행

```
FROM ubuntu
ENTRYPOINT ["top", "-b"]
CMD ["-c"]
```

ONBUILD는 base image로 build가 될 때 실행되는 명령

```
ONBUILD ADD . /app/src
ONBUILD RUN /usr/local/bin/python-build --dir /app/src
```

```
FROM cuda-python:10.1-3.6
WORKDIR /home/jovyan
USER root
RUN pip install jupyter -U && pip install jupyterlab
ENV DEBIAN_FRONTEND=noninteractive
RUN apt-get update && apt-get install -yq --no-install-recommends apt-transport-https build-essential ₩
        bzip2 ca-certificates curl g++ git gnupg graphviz locales lsb-release openssh-client sudo unzip vim ₩
        wget zip emacs python3-pip python3-dev python3-setuptools libgl1-mesa-glx python3-opencv cmake ₩
        ninja-build && apt-get clean && rm -rf /var/lib/apt/lists/*
RUN curl -s https://packages.cloud.google.com/apt/doc/apt-key.gpg | sudo apt-key add –
RUN echo "deb https://apt.kubernetes.io/ kubernetes-xenial main" | ₩
        sudo tee -a /etc/apt/sources.list.d/kubernetes.list
RUN apt-get update
RUN apt-get install -y kubectl
RUN pip install msrestazure cloudpickle kubeflow-fairing==0.7.2 kfp==0.5 --use-feature=2020-resolver ₩
        kfserving==0.2.2.1 kubeflow-kale dill tensorboard torch==1.6 torchvision==0.7 ₩
        –f https://download.pytorch.org/whl/cu101/torch_stable.html opencv-python jupyterlab && ₩
        jupyter serverextension enable --py jupyterlab --sys-prefix
ARG NB_USER=jovyan
EXPOSE 8888
ENV NB_USER $NB_USER
ENV NB_UID=1000
ENV HOME /home/$NB_USER
ENV NB_PREFIX /
CMD ["sh", "-c", "jupyter lab --notebook-dir=/home/jovyan --ip=0.0.0.0 --no-browser --allow-root --port=8888 ₩
         --LabApp.token='' --LabApp.password='' --LabApp.allow_origin='*' --LabApp.base_url=${NB_PREFIX}"]
```