

도커 이미지 경량화 및 빌드 최적화 기법

Docker & CI/CD 스터디 특강 (2차)

튜터 송영훈



빌드 최적화

이미지 경량화

Dockerfile

도커 이미지를 최적화했을 때 얻을 수 있는 이점



CI/CD 시간 단축



저장소 유지 비용 절감



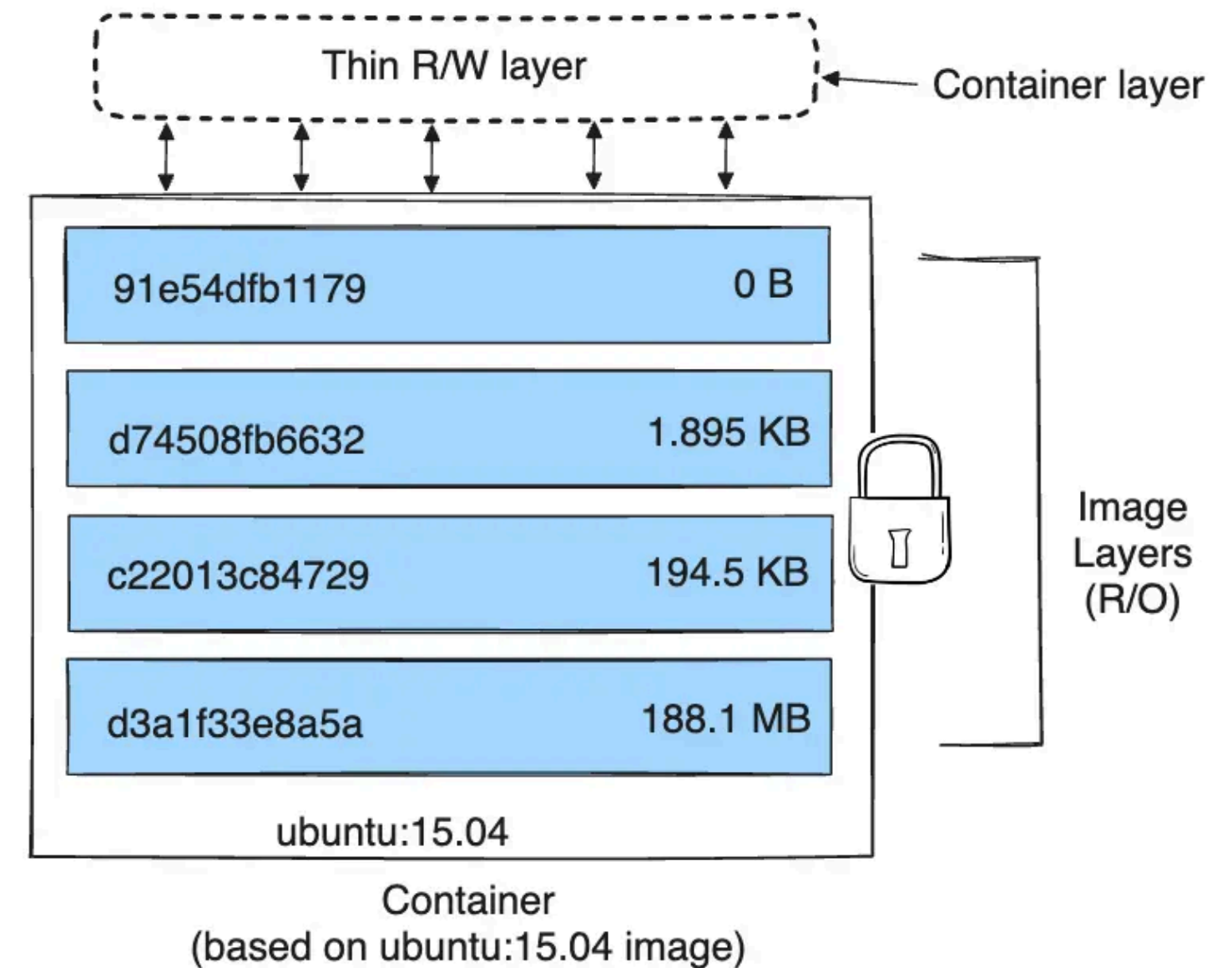
컨테이너가 조금 더 빠르게 동작 할 수 있는 가능성 상승

```
1 FROM python:3.12-slim
2
3 WORKDIR /app
4
5 COPY requirements.txt .
6 RUN pip install -r requirements.txt
7
8 COPY . .
9
10 CMD ["python", "app.py"]
11
```



컨테이너 이미지 레이어란?

```
FROM ubuntu:15.04  
COPY ./app  
RUN make /app  
EXPOSE 8080  
CMD /app/launch
```



왜 레이어 개념이 필요한가요?



이미지의 재사용



빌드 시간 단축

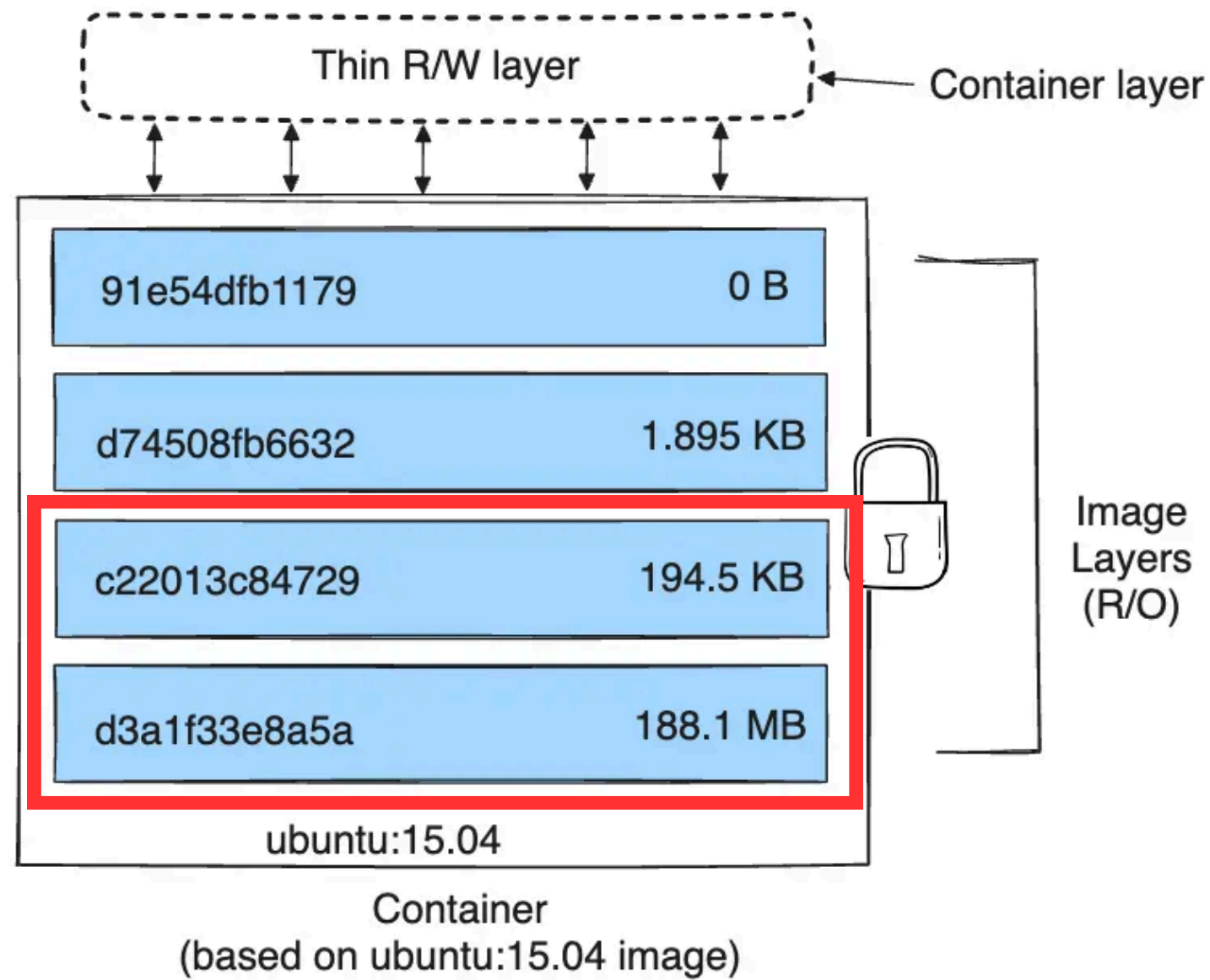


효율적인 배포 (이미지 경량화에 도움)

캐싱(CACHING)이란?

자주 사용하는 데이터나 결과를 임시 저장하여,
필요할 때 빠르게 다시 접근할 수 있도록하는 기법.

도커 이미지 빌드 캐시에 대해 이해하기



새로운 레이어가 추가되는 명령어

FROM	COPY	RUN
ADD	ENV	WORKDIR
EXPOSE / LABEL	USER	VOLUME

캐시를 활용하지 못하는 사례

```
FROM ubuntu:15.04
ENV BUILD_MODE=BLUE
COPY . /app
RUN make /app
CMD /app/launch
```



```
FROM ubuntu:15.04
ENV BUILD_MODE=RED
COPY . /app
RUN make /app
CMD /app/launch
```

일반적으로 생각 할 수 있는 원칙

1. ENV는 사용되기 직전 타이밍에 입력한다.

```
FROM ubuntu:15.04
ENV BUILD_MODE=RED
COPY ./app
RUN make /app
CMD /app/launch
```



```
FROM ubuntu:15.04
COPY ./app
ENV BUILD_MODE=RED
RUN make /app
CMD /app/launch
```

2. 의존 패키지는 미리 따로 설치한다.

```
FROM python:3.12
WORKDIR /app
COPY ..
RUN pip install -r requirements.txt
CMD ["python", "app.py"]
```



```
FROM python:3.12
WORKDIR /app
COPY requirements.txt .
RUN pip install -r requirements.txt
COPY ..
CMD ["python", "app.py"]
```

3. 가능하면 ADD보다 COPY를 쓰자.

도커 이미지를 경량화 하기 위해 고려해야 할 것들

1. 베이스 이미지를 잘 선택하기

2. 불필요한 파일을 이미지에 포함 시키지 않기

경량화 이미지 선택

python:3.12 MULTI-PLATFORM

LANGUAGES & FRAMEWORKS

INDEX DIGEST sha256:b52e97a85736bc3a6bfb081ec3f2a5b828233120b2138080af1761e504b06529

OS/ARCH

linux/amd64

COMPRESSED SIZE ⓘ

363.36 MB

LAST PUSHED

3 days ago by [doijanky](#)

TYPE

Image

VULNERABILITIES

0	1	2	113	2
---	---	---	-----	---

MANIFEST DIGEST

sha256:1b7f52f4a...

python:3.12-slim MULTI-PLATFORM

LANGUAGES & FRAMEWORKS

INDEX DIGEST sha256:032c52613401895aa3d418a4c563d2d05f993bc3ecc065c8f4e2280978acd249

OS/ARCH

linux/amd64

COMPRESSED SIZE ⓘ

44.12 MB

LAST PUSHED

4 days ago by [doijanky](#)

TYPE

Image

VULNERABILITIES

0	0	0	26	0
---	---	---	----	---

MANIFEST DIGEST

sha256:311c6c70...

DOCKERIGNORE 파일을 사용하여 불필요한 파일을 포함 시키지 않기

파일명: dockerignore

__pycache__/

*.py[cod]

*.pyo

*.pyd

venv/

env/

tests/

.coverage

htmlcov/

멀티 스테이지 빌드 활용

빌드 스테이지

FROM maven:3.8.6-eclipse-temurin-17 AS builder

WORKDIR /app

COPY pom.xml .

RUN mvn dependency:go-offline

COPY src ./src

RUN mvn package -DskipTests

실행 스테이지

FROM eclipse-temurin:17-jre-alpine

WORKDIR /app

COPY --from=builder /app/target/*.jar app.jar

CMD ["java", "-jar", "/app/app.jar"]

<https://github.com/wagoodman/dive>

```
shabbirkheriwala — dive mongo:3.6 — dive — dive mongo:3.6 — 130x40

| Layers |
|-----|
| Cmp | Size | Command |
|-----|-----|-----|
| 119 MB | FROM 3106fb756655e55 |
| 745 B | set -xe && echo '#!/bin/sh' > /usr/sbin/poli |
| 0 B | rm -rf /var/lib/apt/lists/* |
| 7 B | mkdir -p /run/systemd && echo 'docker' > /run/system |
| 334 kB | groupadd -r mongodb && useradd -r -g mongodb mongodb |
| 6.0 MB | set -eux; apt-get update; apt-get install -y |
| 3.7 MB | set -ex; savedAptMark="$(apt-mark showmanual |
| 0 B | mkdir /docker-entrypoint-initdb.d |
| 1.7 kB | set -ex; export GNUPGHOME="$(mktemp -d)"; fo |
| 73 B | echo "deb http://$MONGO_REPO/apt/ubuntu xenial/${MON |
| 293 MB | set -x && export DEBIAN_FRONTEND=noninteractive |
| 0 B | mkdir -p /data/db /data/configdb && chown -R mon |
| 13 kB | #(nop) COPY file:1dd28550b0c6cd4baae08342a3beff8f601 |

| Layer Details |
|-----|
Tags: (unavailable)
Id: f0bb84b3e0b7937d54b022aa0e53be651c60434aec8ad4dc382a1270
Digest: sha256:eb81a039824ddd9fd156a18e4a0e67b9362294b6678c9d5b8
Command:
set -xe && echo '#!/bin/sh' > /usr/sbin/policy-rc.d

| Image Details |
|-----|
Image name: mongo:3.6
Total Image size: 422 MB
Potential wasted space: 44 MB
Image efficiency score: 89 %

Count Total Space Path
2 9.0 MB /var/lib/apt/lists/ports.ubuntu.com_ubuntu-
2 7.9 MB /var/lib/apt/lists/ports.ubuntu.com_ubuntu-
2 6.8 MB /var/lib/apt/lists/ports.ubuntu.com_ubuntu-
2 6.3 MB /var/lib/apt/lists/ports.ubuntu.com_ubuntu-
2 4.6 MB /var/lib/apt/lists/ports.ubuntu.com_ubuntu-
2 3.6 MB /var/lib/apt/lists/ports.ubuntu.com_ubuntu-
4 2.0 MB /var/cache/debconf/templates.dat

^C Quit | Tab Switch view | ^F Filter | Space Collapse dir | ^Space Collapse all dir | ^A Added | ^R Removed | ^M Modified | ^U Unmodifie
```


+

Github에서 Gitlab처럼 CI/CD 파이프라인을 구축하는 방법

<https://docs.github.com/ko/actions>

name: Build and Deploy to ECS

on:
push:
branches:
- main

jobs:
build_and_push_to_ecr:
name: Build Docker image and push to ECR
runs-on: ubuntu-latest

steps:
Checkout the repository
- name: Checkout code
uses: actions/checkout@v3

Set up Python environment
- name: Set up Python 3.9
uses: actions/setup-python@v4
with:
python-version: '3.9'

Install Python dependencies (if any)
- name: Install dependencies
run: |
python -m pip install --upgrade pip
if [-f requirements.txt]; then pip install -r requirements.txt; fi

Set up AWS credentials
- name: Configure AWS credentials
uses: aws-actions/configure-aws-credentials@v2
with:
aws-access-key-id: \${ secrets.AWS_ACCESS_KEY_ID }
aws-secret-access-key: \${ secrets.AWS_SECRET_ACCESS_KEY }
aws-region: <your-aws-region>

Log in to Amazon ECR
- name: Log in to Amazon ECR
run: |
aws ecr get-login-password --region <your-aws-region> | docker login --username AWS --password-stdin <aws-account-id>.dkr.ecr.<your-aws-region>.amazonaws.com

Build the Docker image using make build
- name: Build Docker image
run: make build

Tag and push the Docker image to ECR
- name: Tag and Push Docker image to ECR
run: |
IMAGE_TAG=\${ github.sha }
docker tag <your-docker-image>:latest <aws-account-id>.dkr.ecr.<your-aws-region>.amazonaws.com/<your-ecr-repository>:\${IMAGE_TAG}
docker push <aws-account-id>.dkr.ecr.<your-aws-region>.amazonaws.com/<your-ecr-repository>:\${IMAGE_TAG}

Deploy the image (optional step depending on your setup)
This step assumes that you have some deployment mechanism such as ECS, EKS, or Lambda
- name: Deploy to ECS
run: |
aws ecs update-service --cluster <your-ecs-cluster> --service <your-ecs-service> --force-new-deployment



Q&A