Enhancing the Workflow

Justin Menga

FULL STACK TECHNOLOGIST

@jmenga   pseudo.co.de

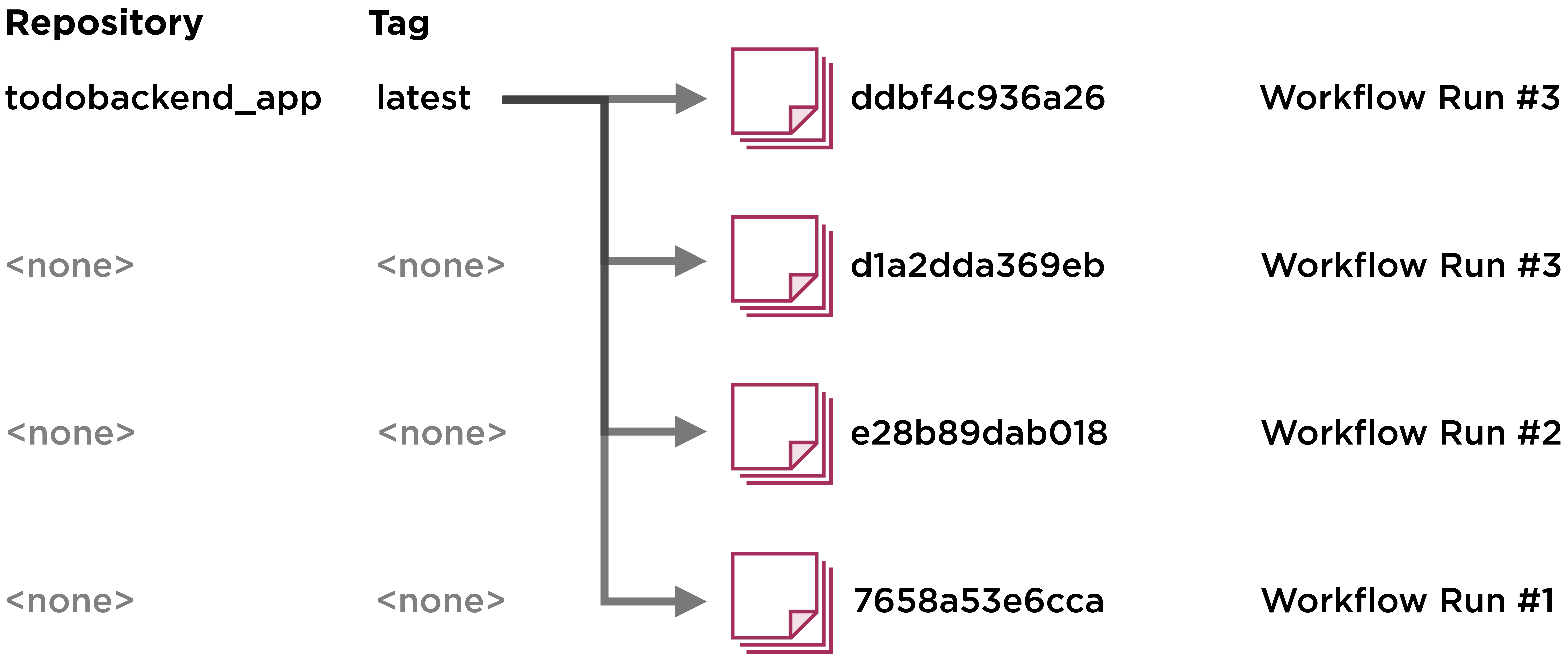
Introduction

Enhancing the Workflow

- Dangling images and volumes
- Improving user feedback
- Making the workflow self-contained
- Producing test reports
- Handling errors
- Ensuring consistency
- Tagging and publishing
- Docker Compose v2 specification

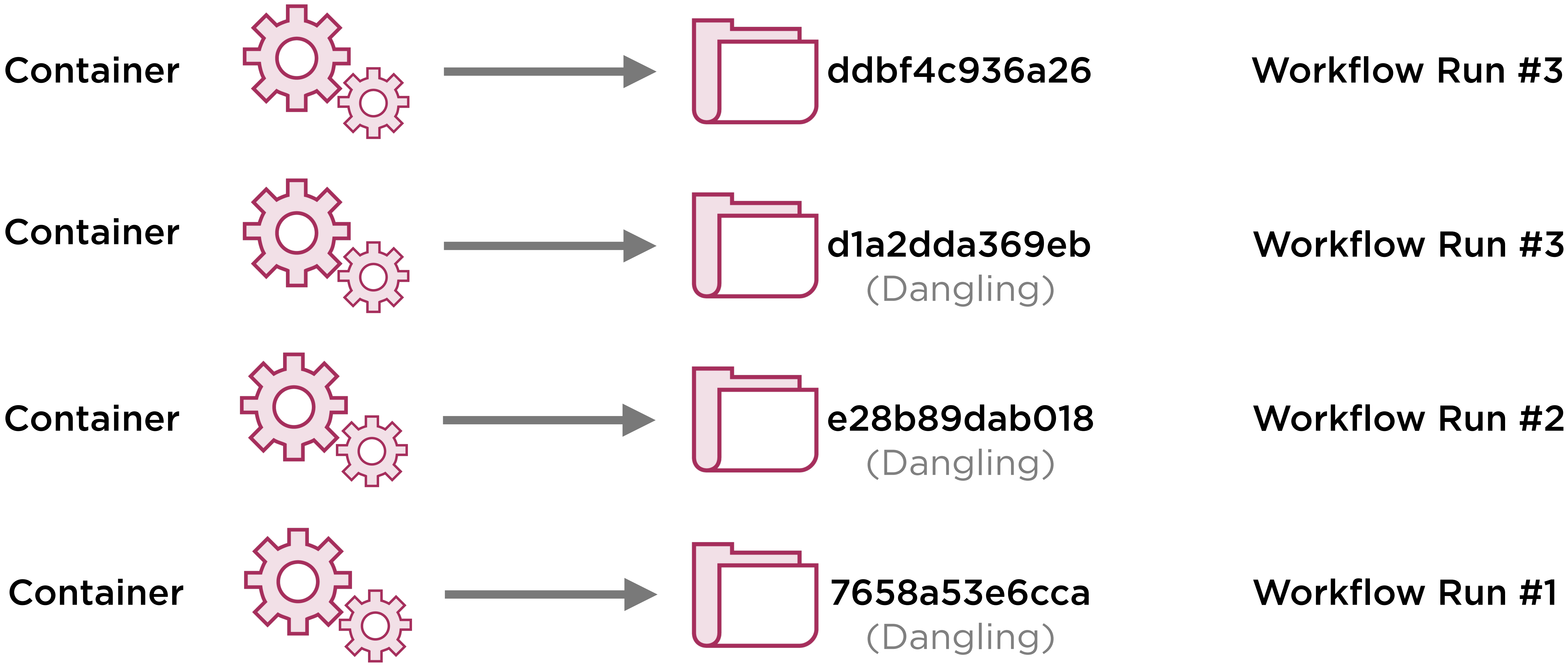
Dangling Images and Volumes

Dangling Images



Repository	Tag		
todobackend_app	latest	ddbf4c936a26	Workflow Run #3
<none>	<none>	d1a2dda369eb	Workflow Run #3
<none>	<none>	e28b89dab018	Workflow Run #2
<none>	<none>	7658a53e6cca	Workflow Run #1

Dangling Volumes



Container → ddbf4c936a26 Workflow Run #3

Container → d1a2dda369eb (Dangling) Workflow Run #3

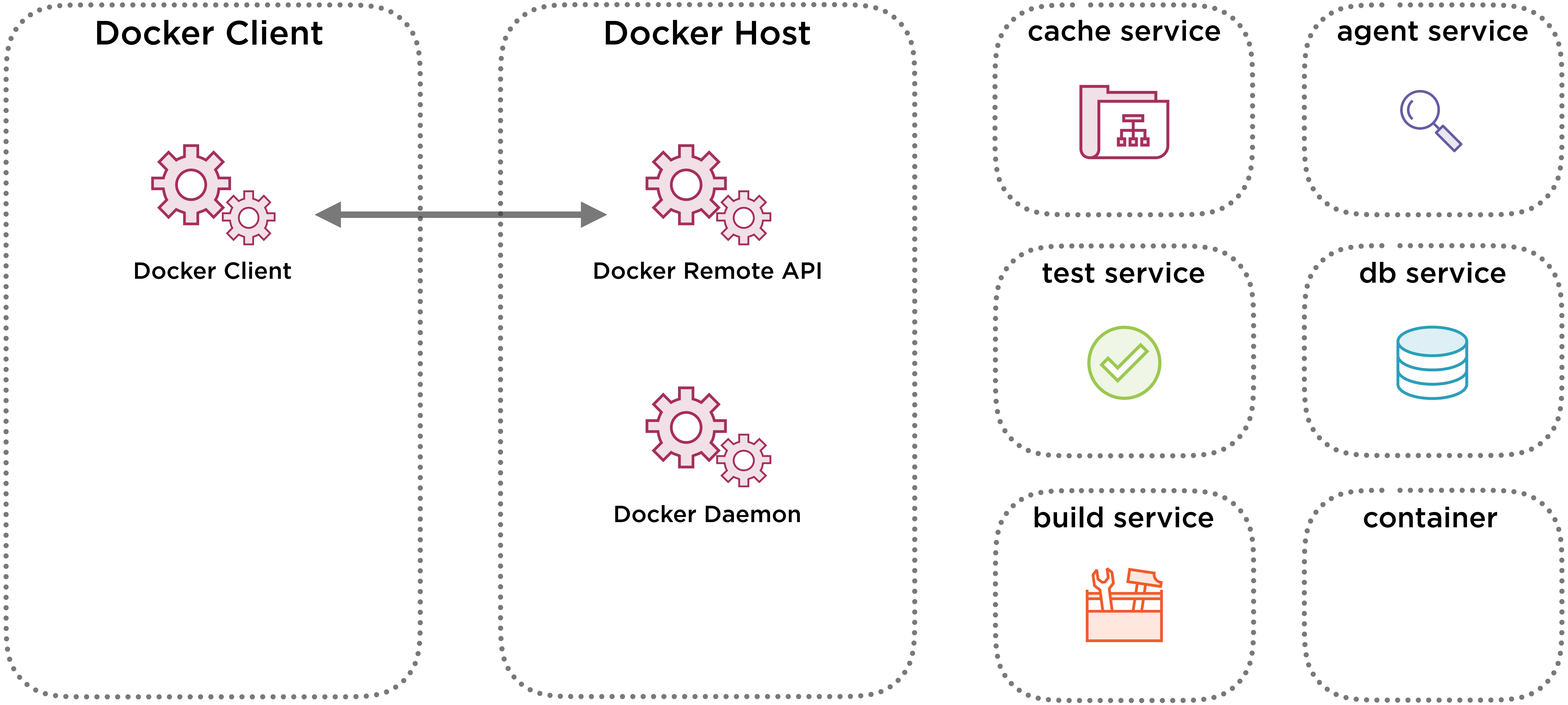
Container → e28b89dab018 (Dangling) Workflow Run #2

Container → 7658a53e6cca (Dangling) Workflow Run #1

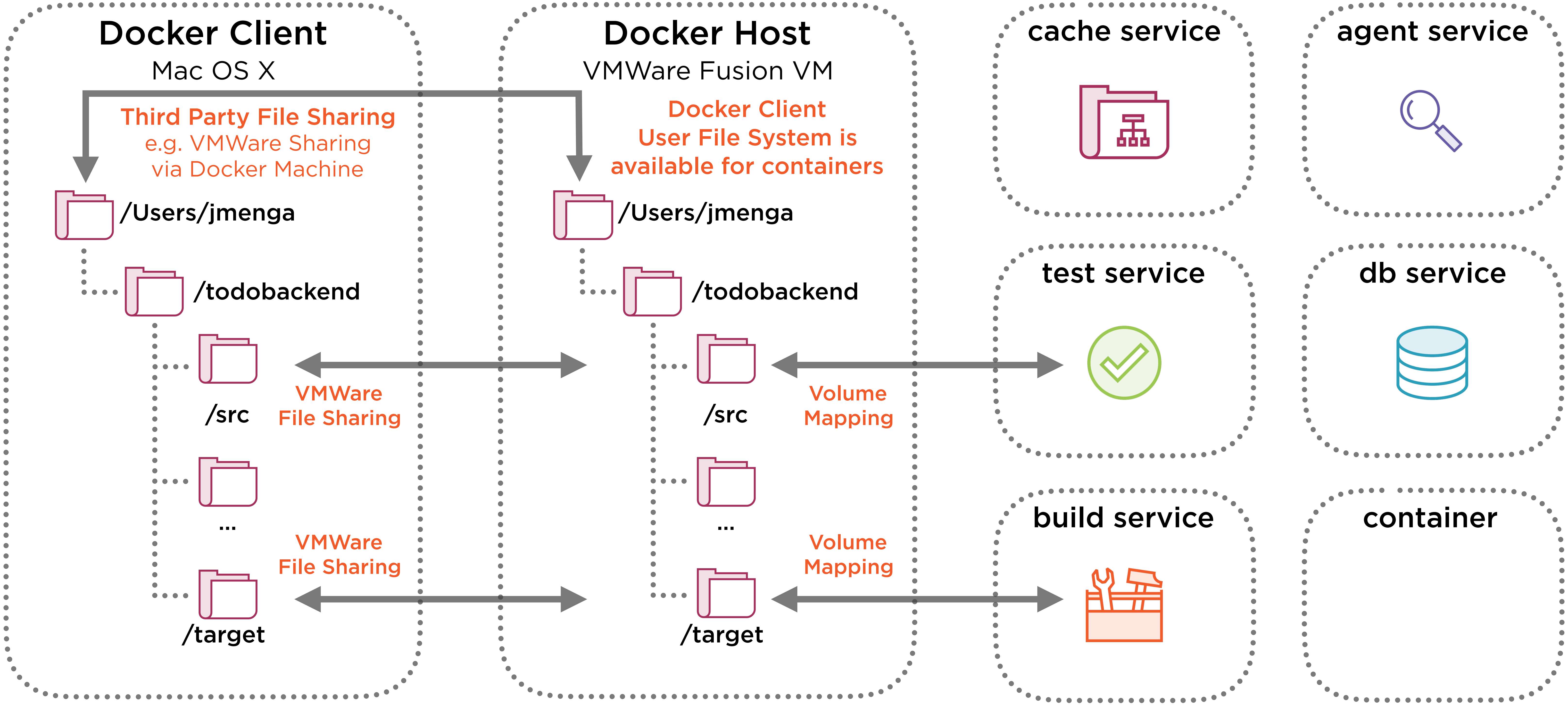
Improving User Feedback

Self Containment

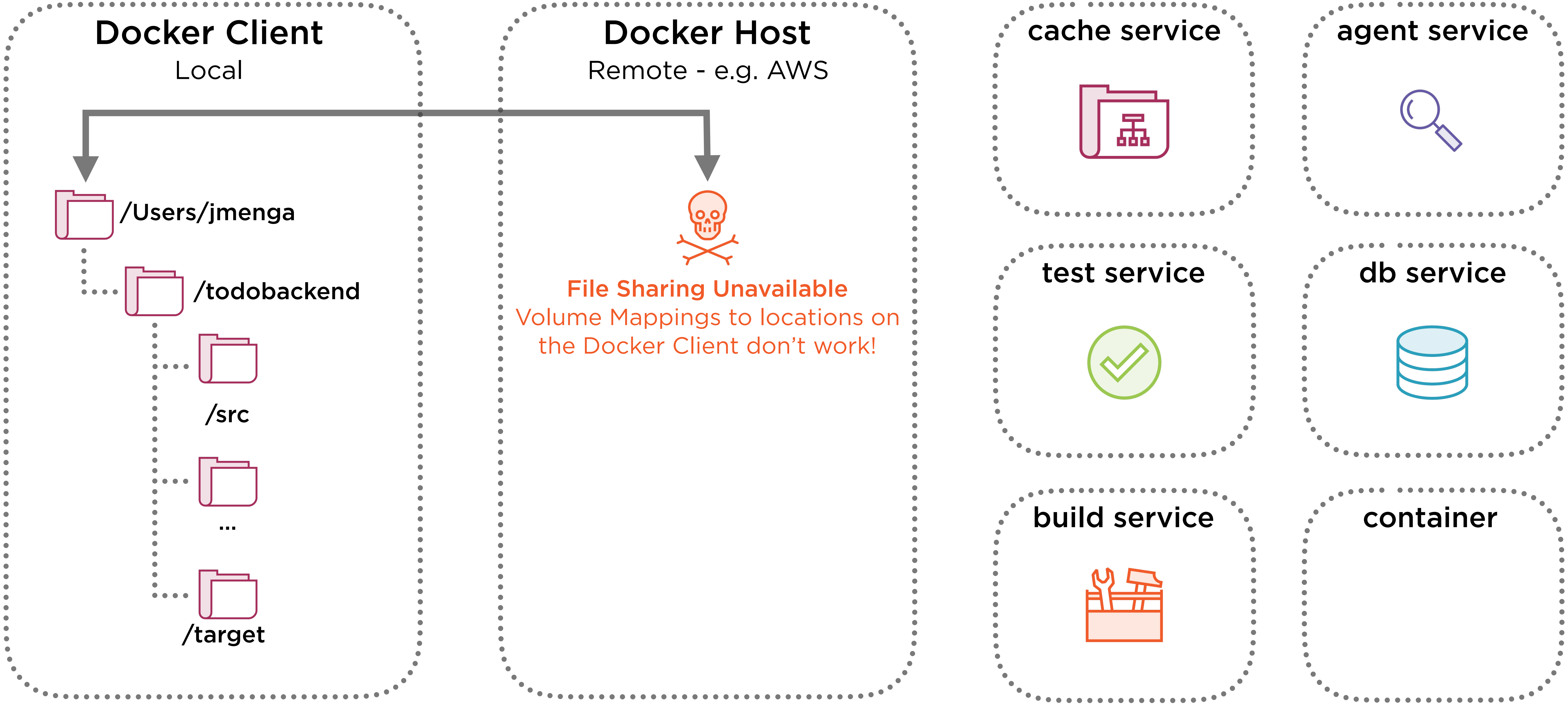
Workflow Systems Architecture



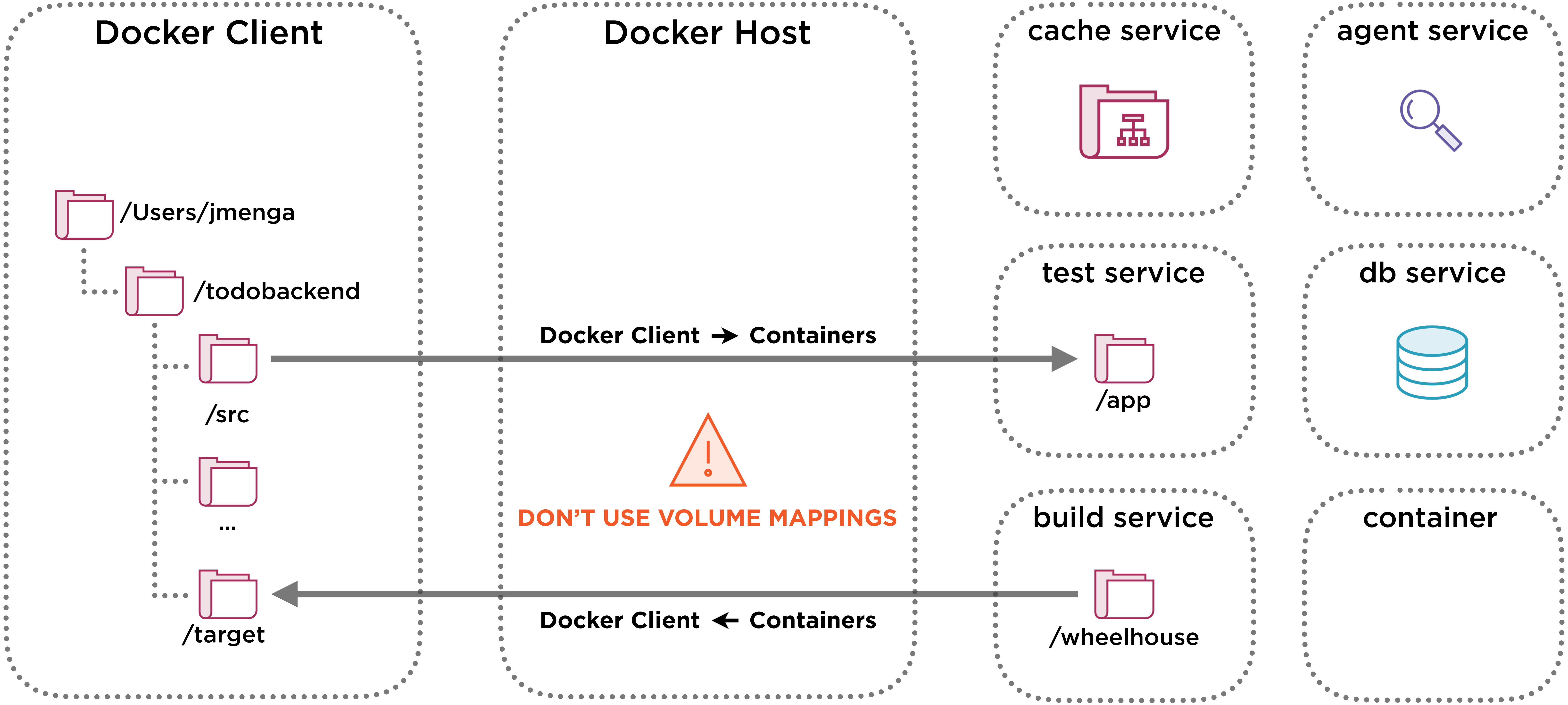
Docker Machine Local Environment



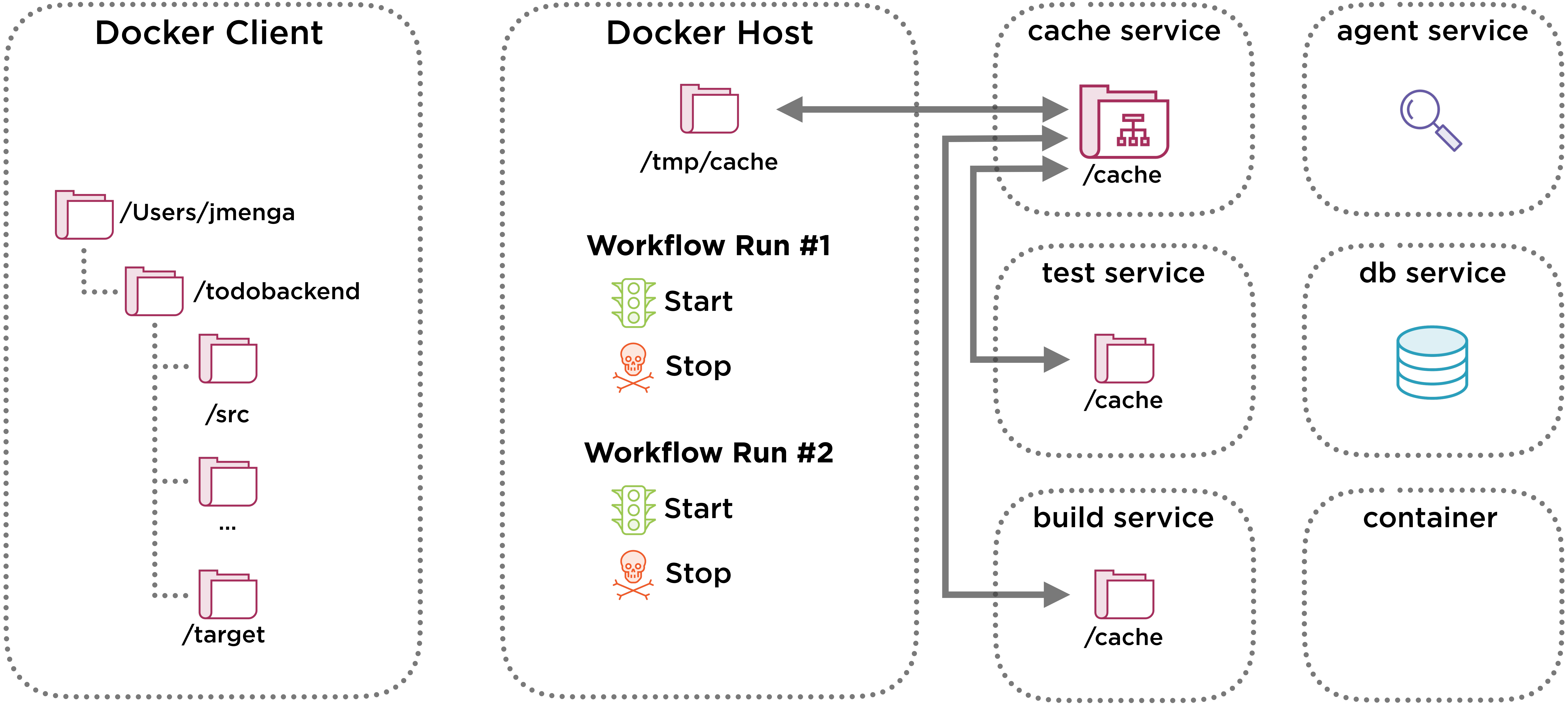
The Problem of Volume Mapping



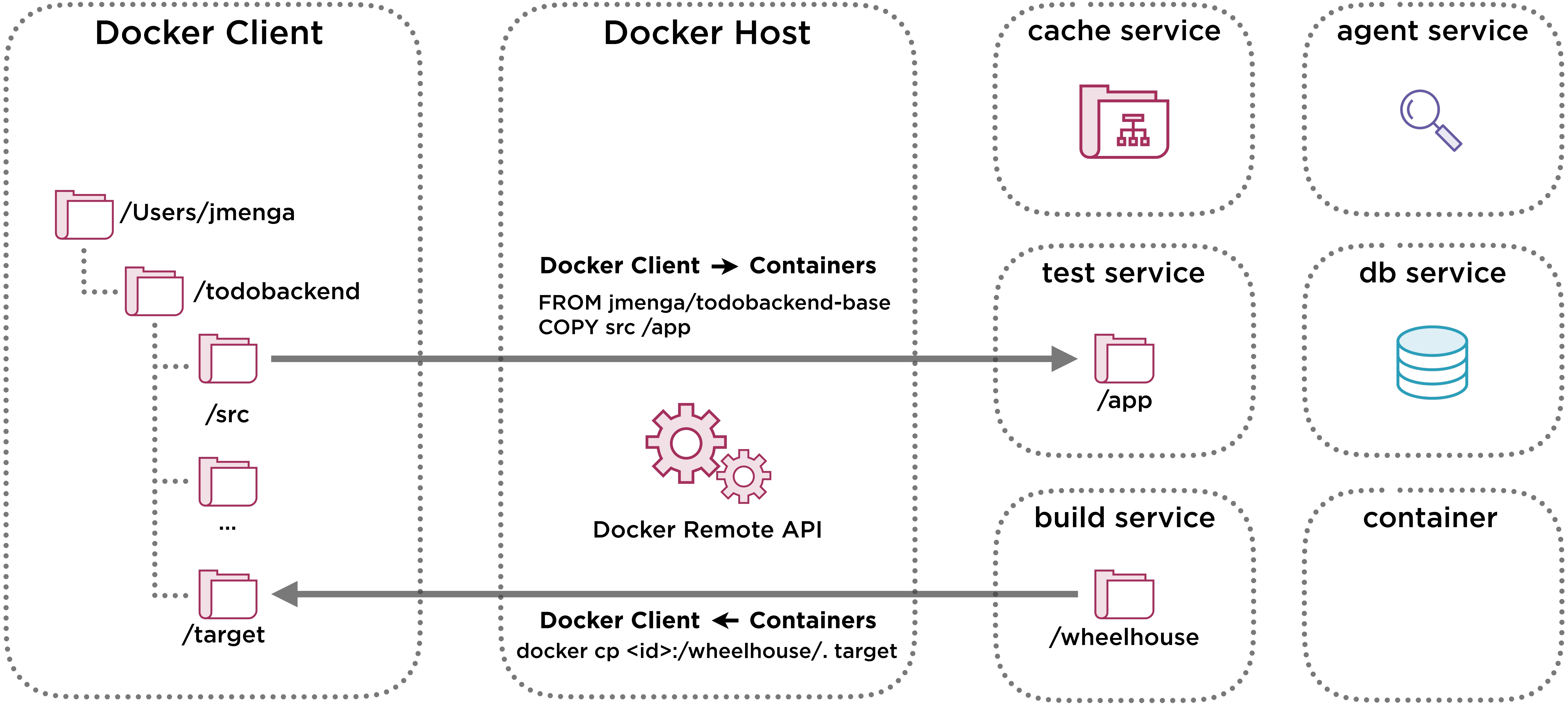
Sharing Files from the Docker Client



Sharing Files with the Docker Host



Sharing Files using the Docker Remote API



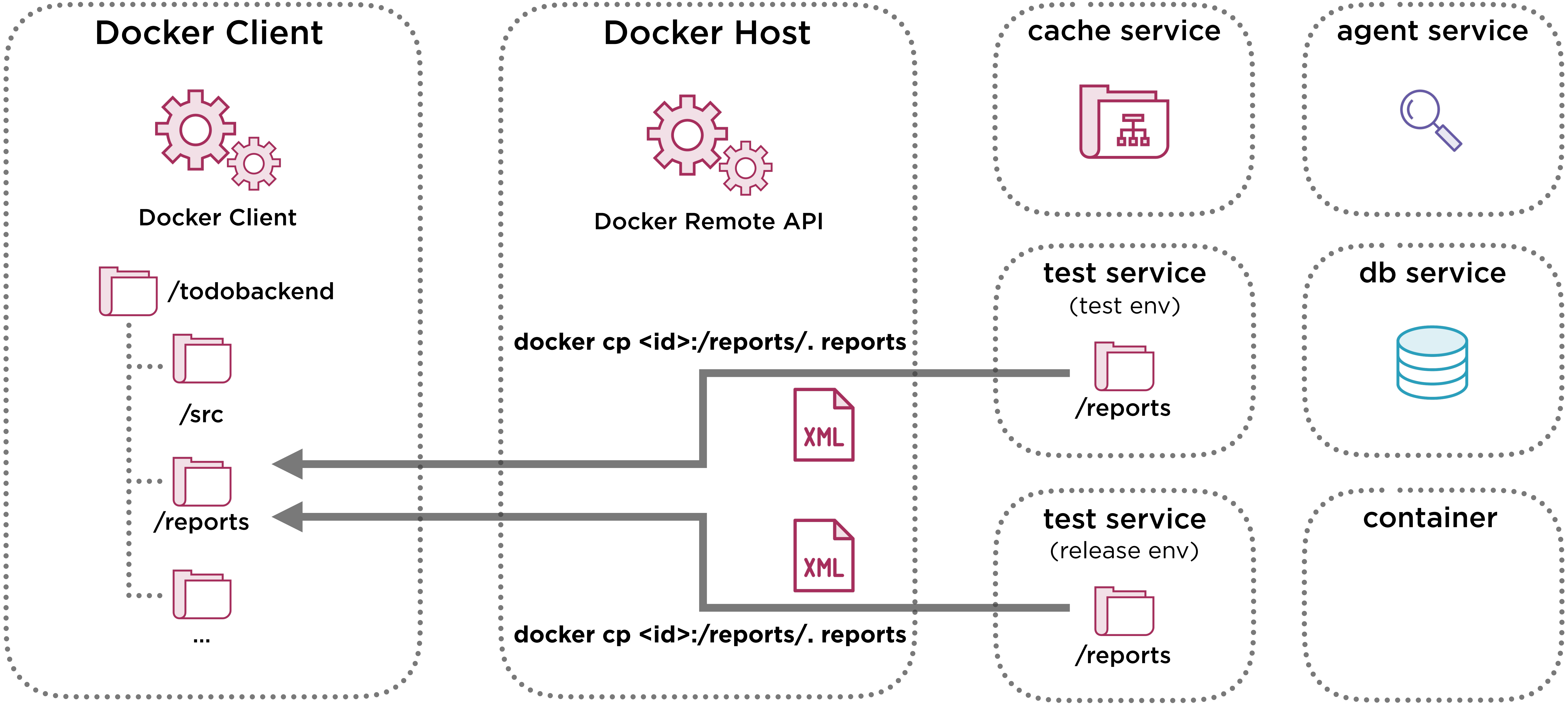
```
FROM jmenga/todobackend-base
COPY src /app
```

```
docker cp <id>:/wheelhouse/. target
```

Ensuring Self Containment

Producing Test Reports

Accessing Test Reports



```
docker cp <id>:/reports/. reports
```

```
docker cp <id>:/reports/. reports
```

Handling Failures and Errors

Make Error Handling

```
$ make test
...
make: my_task exited with code 1
make: *** [test] Error 1

$ echo $?
2
```

◀ Our make task

◀ One of the make rules fails with an error code of 1

◀ This prints the exit code of the last command, in this case the make exit code is always 2, regardless of the rule exit code

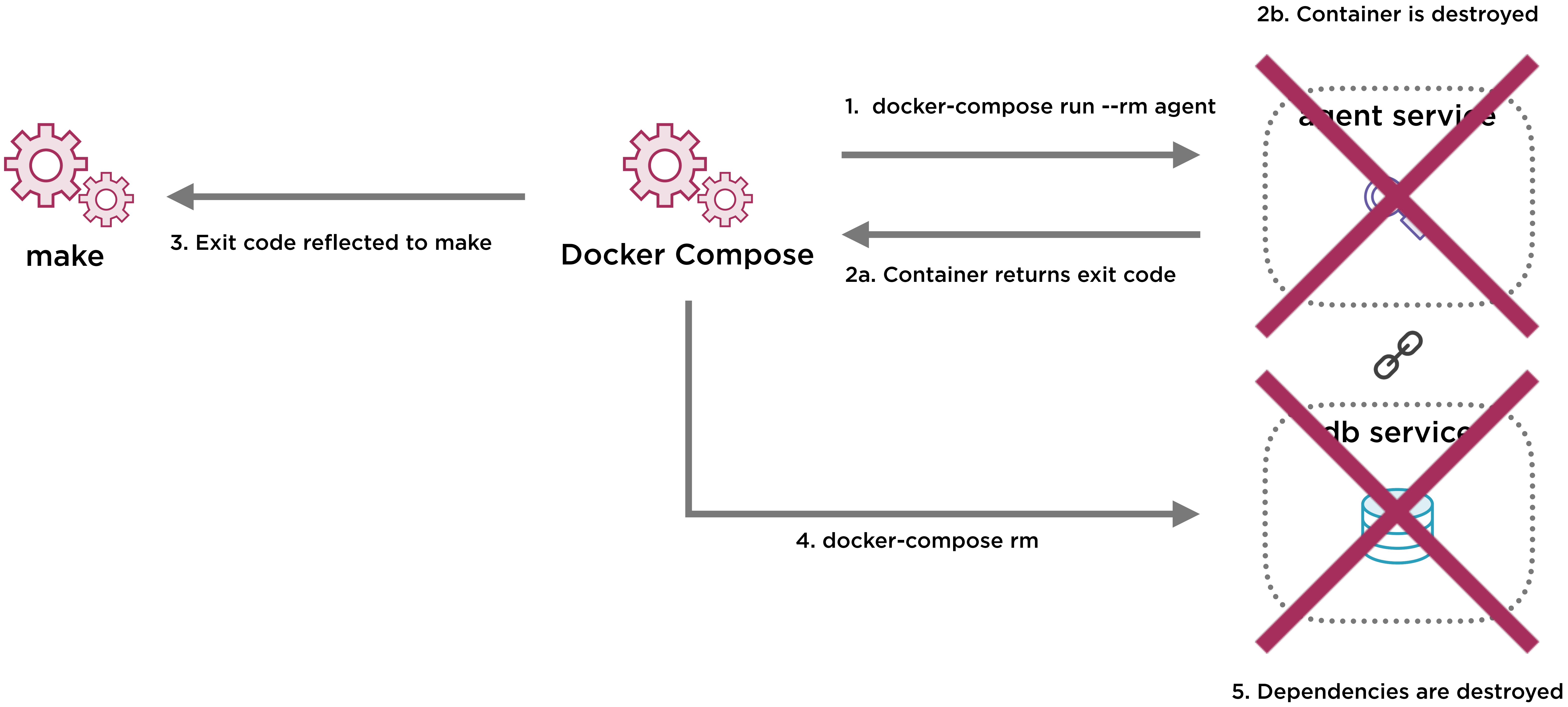
Docker Compose Error Handling

```
$ docker-compose up test
...
test_1 | FAILED (failures=3)
test_1 exited with code 3

$ echo $?
0
```

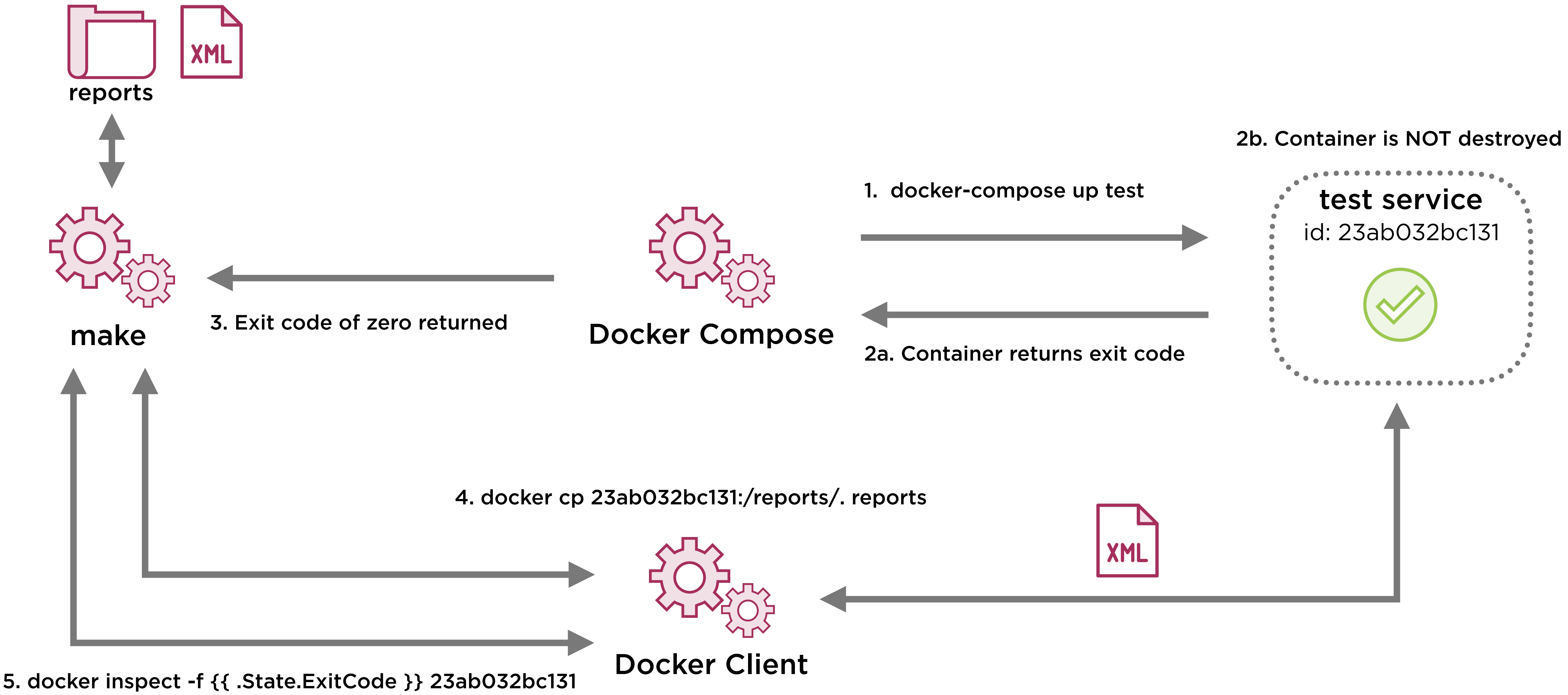
- We “up” the test service
- The test service fails with an error code of 3 (3 tests failed)
- The docker-compose up exit code is 0 (success), even though the test service failed

Docker Compose Strategy for Short Lived Tasks



2b. Container is destroyed
1. docker-compose run --rm agent
agent service
make
3. Exit code reflected to make
Docker Compose
2a. Container returns exit code
db service
4. docker-compose rm
5. Dependencies are destroyed

Docker Compose Strategy If Files Are Needed



1. docker-compose up test

2a. Container returns exit code

2b. Container is NOT destroyed

3. Exit code of zero returned

4. docker cp 23ab032bc131:/reports/. reports

5. docker inspect -f {{ .State.ExitCode }} 23ab032bc131

Ensuring Consistency

Docker Image Consistency Goals

External Images

Always use the latest image

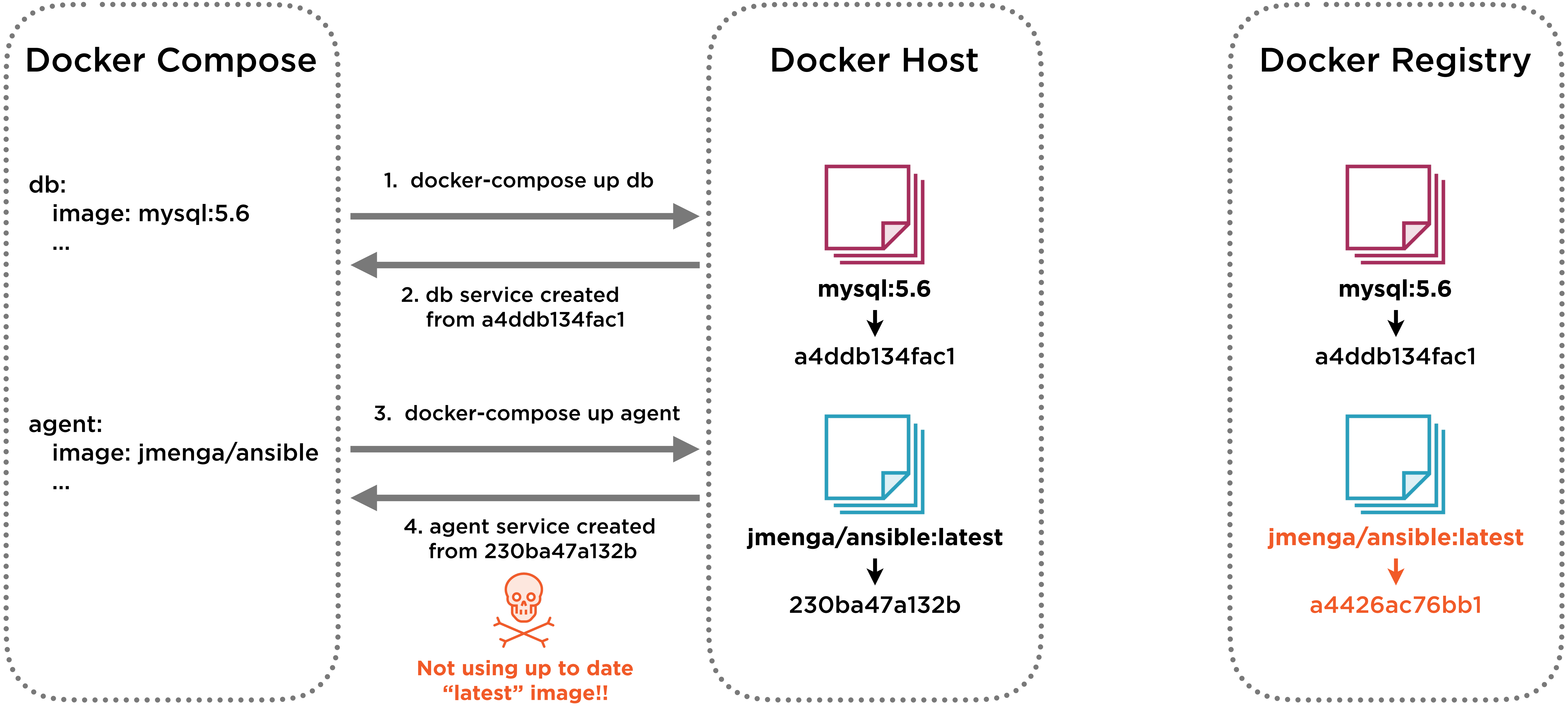
Built Images

Always use the latest parent image

Consistent Images

Once latest images are downloaded, keep image version consistent for the workflow run

Docker Image Caching



Docker Compose

```
db:
   image: mysql:5.6
   ...

agent:
   image: jmenga/ansible
   ...
```

1. docker-compose up db

2. db service created from a4ddb134fac1

3. docker-compose up agent

4. agent service created from 230ba47a132b

Not using up to date “latest” image!!

Docker Host

mysql:5.6

↓

a4ddb134fac1

jmenga/ansible:latest

↓

230ba47a132b

Docker Registry

mysql:5.6

↓

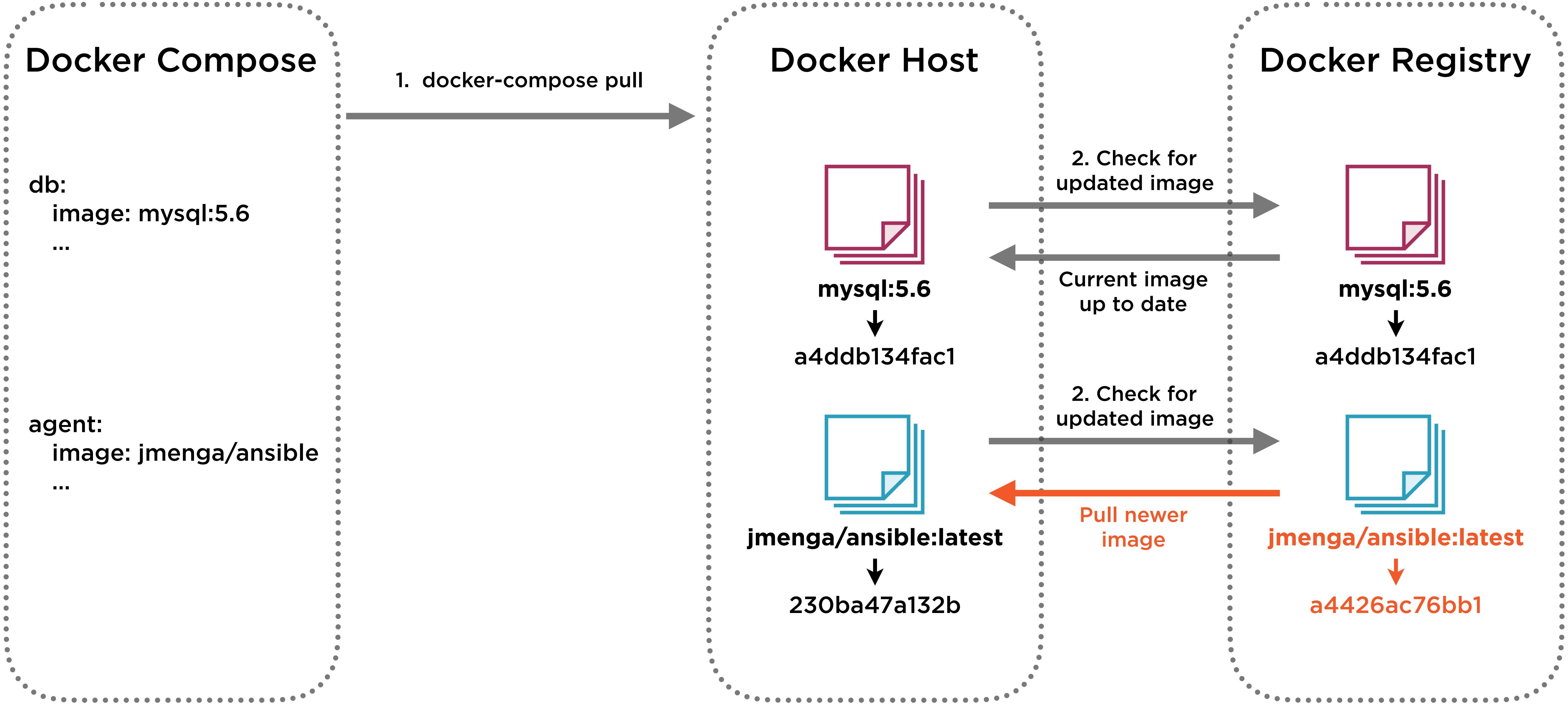
a4ddb134fac1

jmenga/ansible:latest

↓

a4426ac76bb1

Ensuring Docker Images are Up to Date



Docker Compose

```
db:
   image: mysql:5.6
   ...

agent:
   image: jmenga/ansible
   ...
```

1. docker-compose pull

Docker Host

mysql:5.6

a4ddb134fac1

jmenga/ansible:latest

230ba47a132b

2. Check for updated image

Current image up to date

2. Check for updated image

Pull newer image

Docker Registry

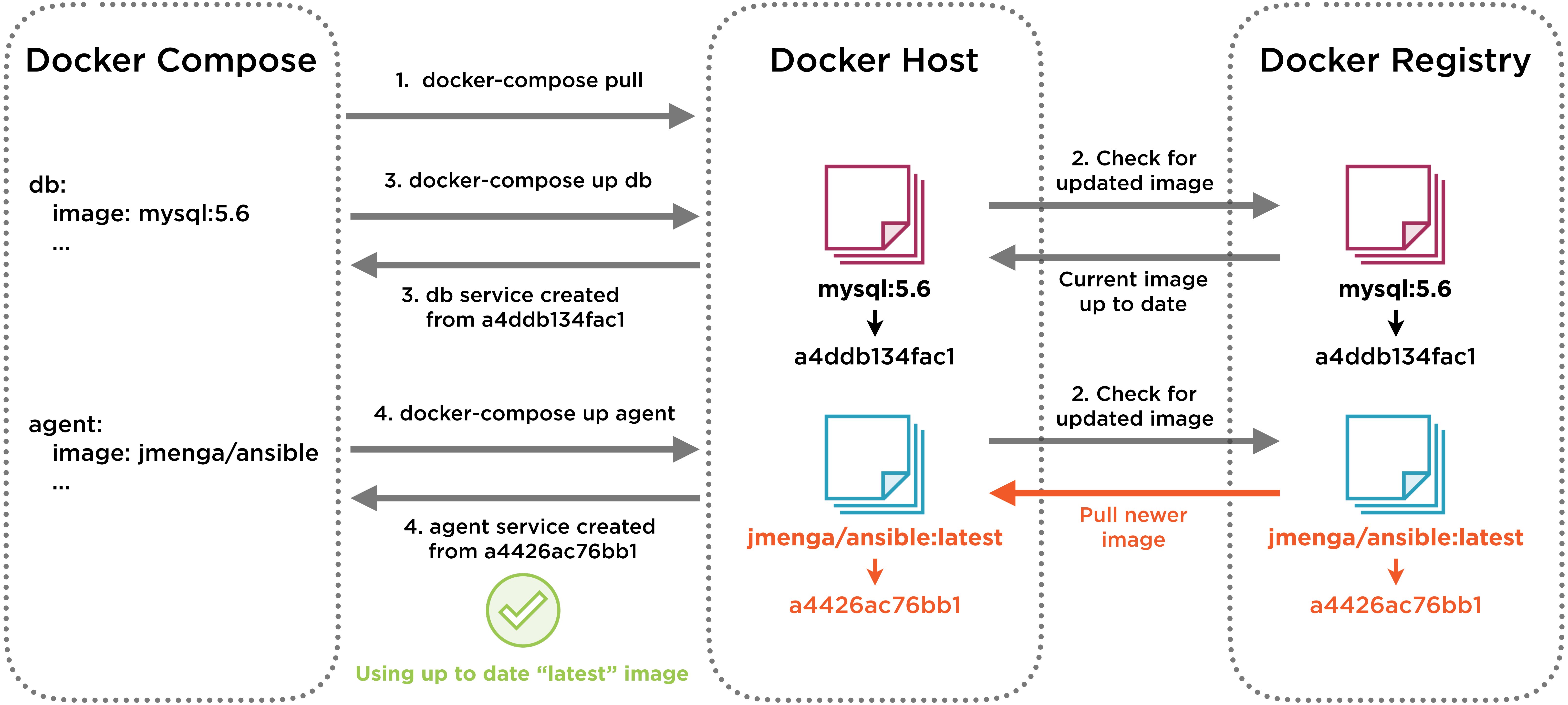
mysql:5.6

a4ddb134fac1

jmenga/ansible:latest

a4426ac76bb1

Ensuring Docker Images are Up to Date



Docker Compose

db:
  image: mysql:5.6
  ...

agent:
  image: jmenga/ansible
  ...

1. docker-compose pull

3. docker-compose up db

3. db service created from a4ddb134fac1

4. docker-compose up agent

4. agent service created from a4426ac76bb1

Using up to date “latest” image

Docker Host

mysql:5.6 ↓ a4ddb134fac1

jmenga/ansible:latest ↓ a4426ac76bb1

2. Check for updated image

Current image up to date

2. Check for updated image

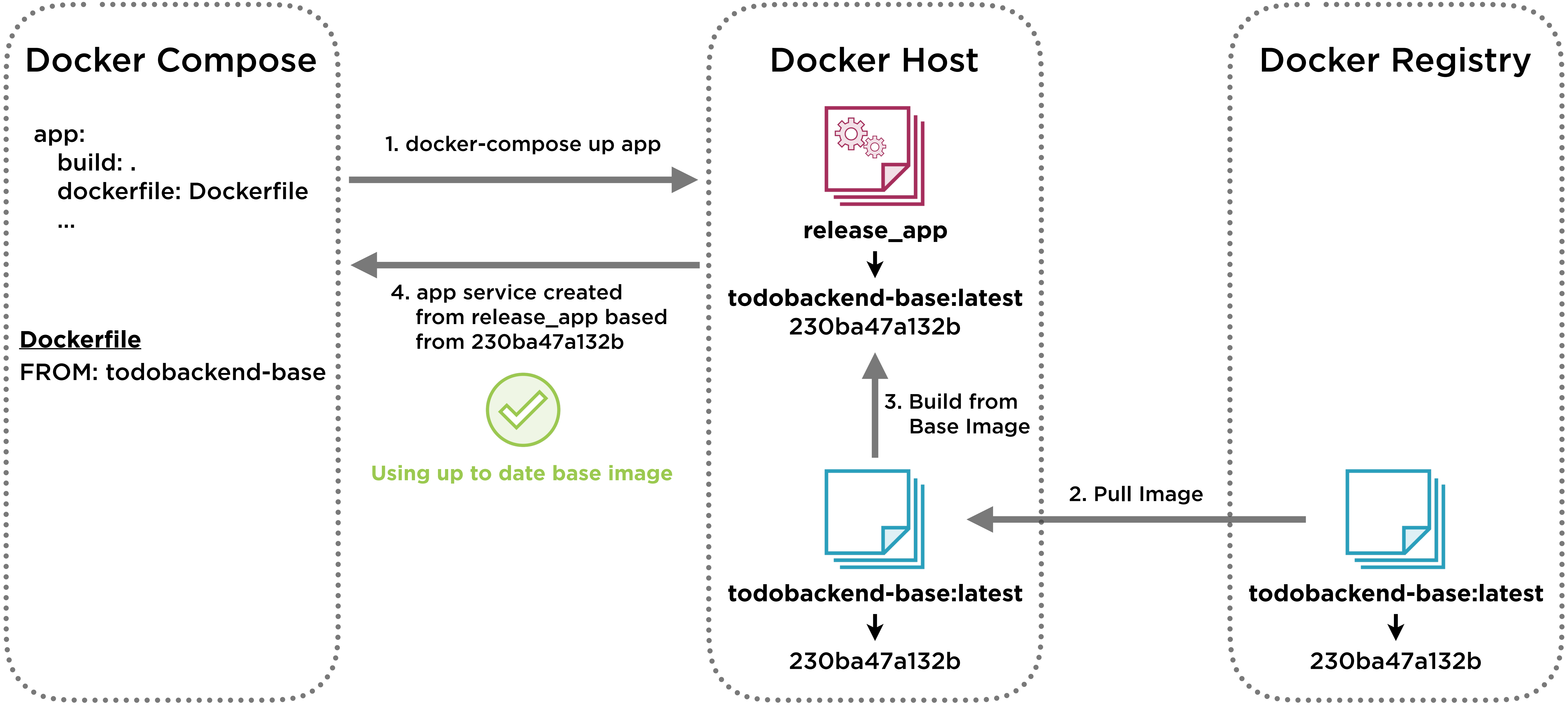
Pull newer image

Docker Registry

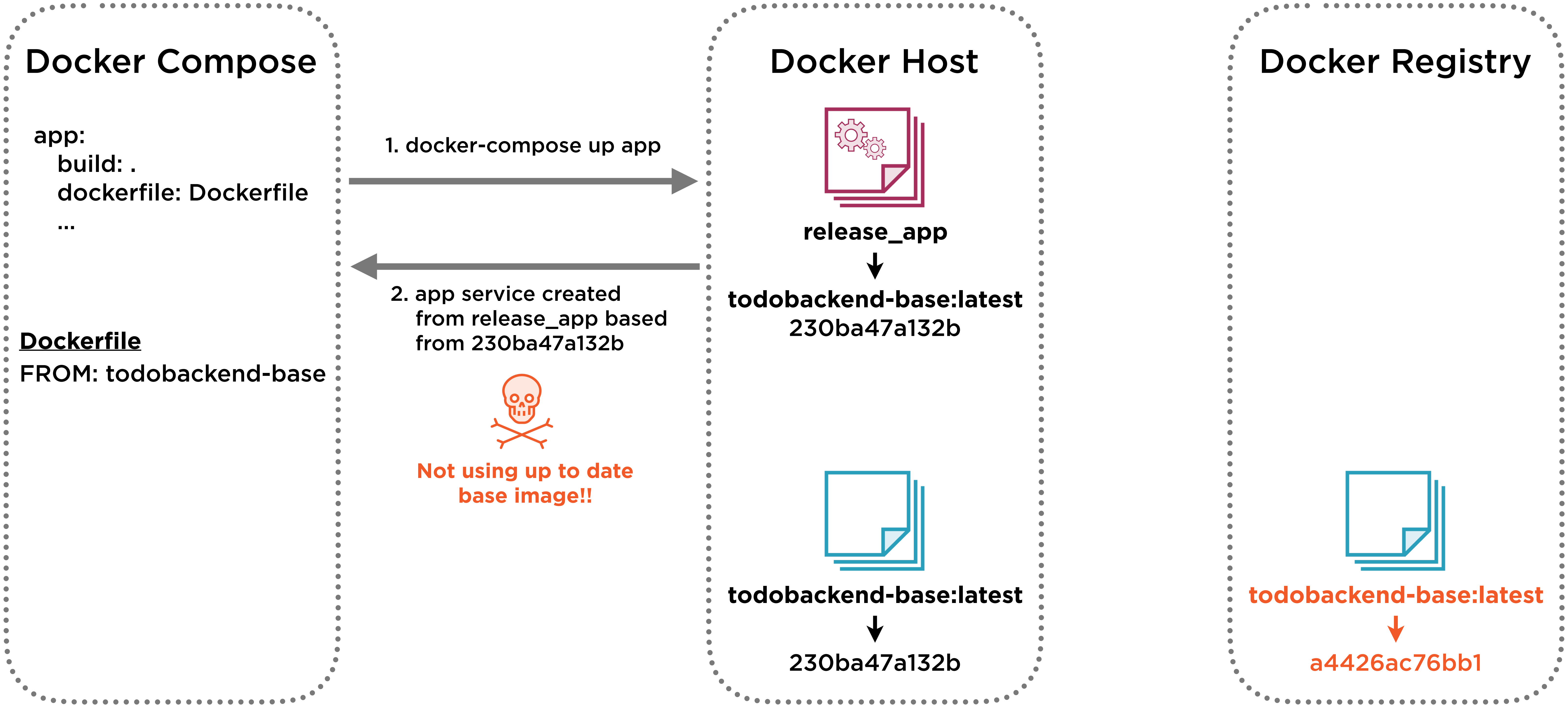
mysql:5.6 ↓ a4ddb134fac1

jmenga/ansible:latest ↓ a4426ac76bb1

Building Up to Date Docker Images



Building Up to Date Docker Images



Docker Compose

```
app:
   build: .
   dockerfile: Dockerfile
   ...
```

Dockerfile

FROM: todobackend-base

1. docker-compose up app

2. app service created
   from release_app based
   from 230ba47a132b

Not using up to date base image!!

Docker Host

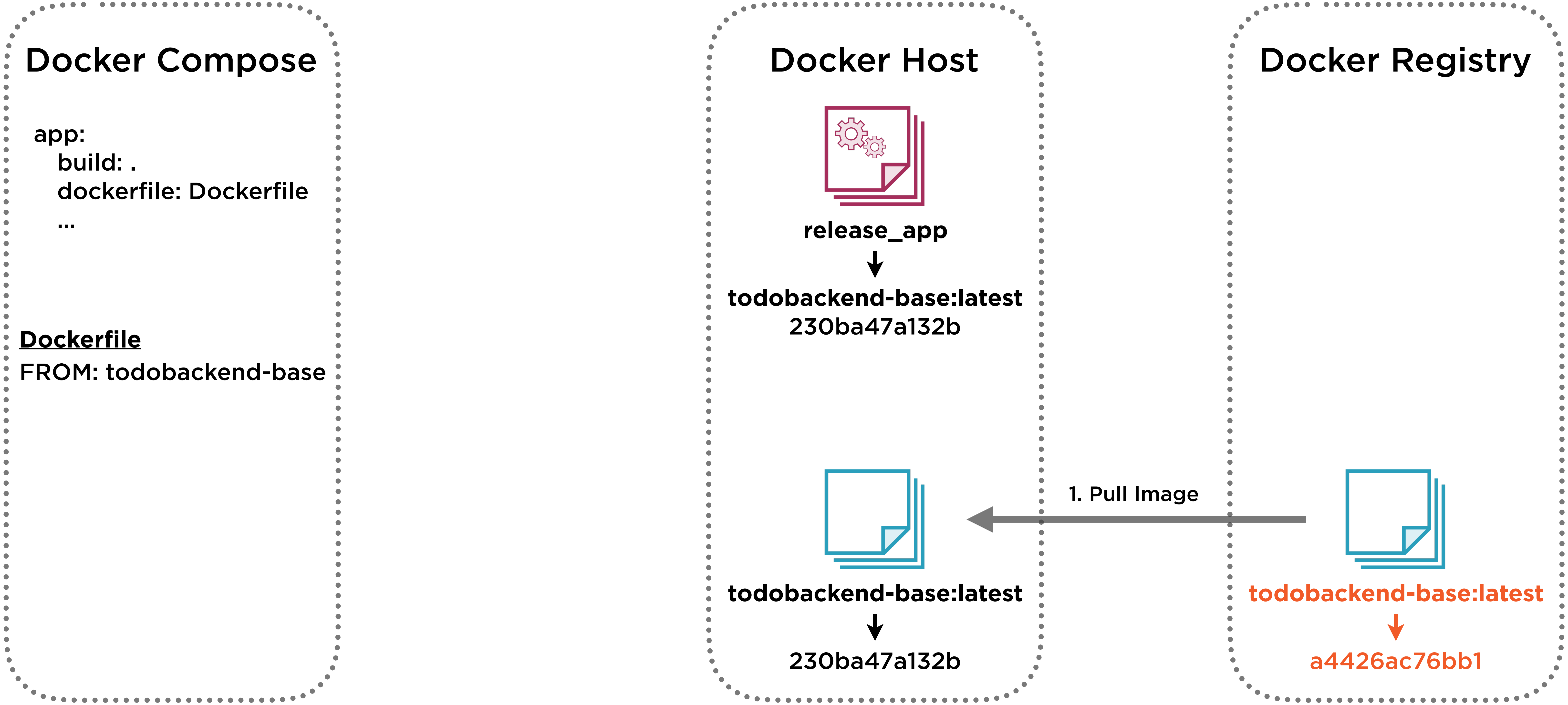
release_app
↓
todobackend-base:latest
230ba47a132b

todobackend-base:latest
↓
230ba47a132b

Docker Registry

todobackend-base:latest
↓
a4426ac76bb1

Building Up to Date Docker Images



Docker Compose

```
app:
   build: .
   dockerfile: Dockerfile
   ...
```

Dockerfile

FROM: todobackend-base

Docker Host

release_app

↓

todobackend-base:latest
230ba47a132b

todobackend-base:latest

↓

230ba47a132b

1. Pull Image

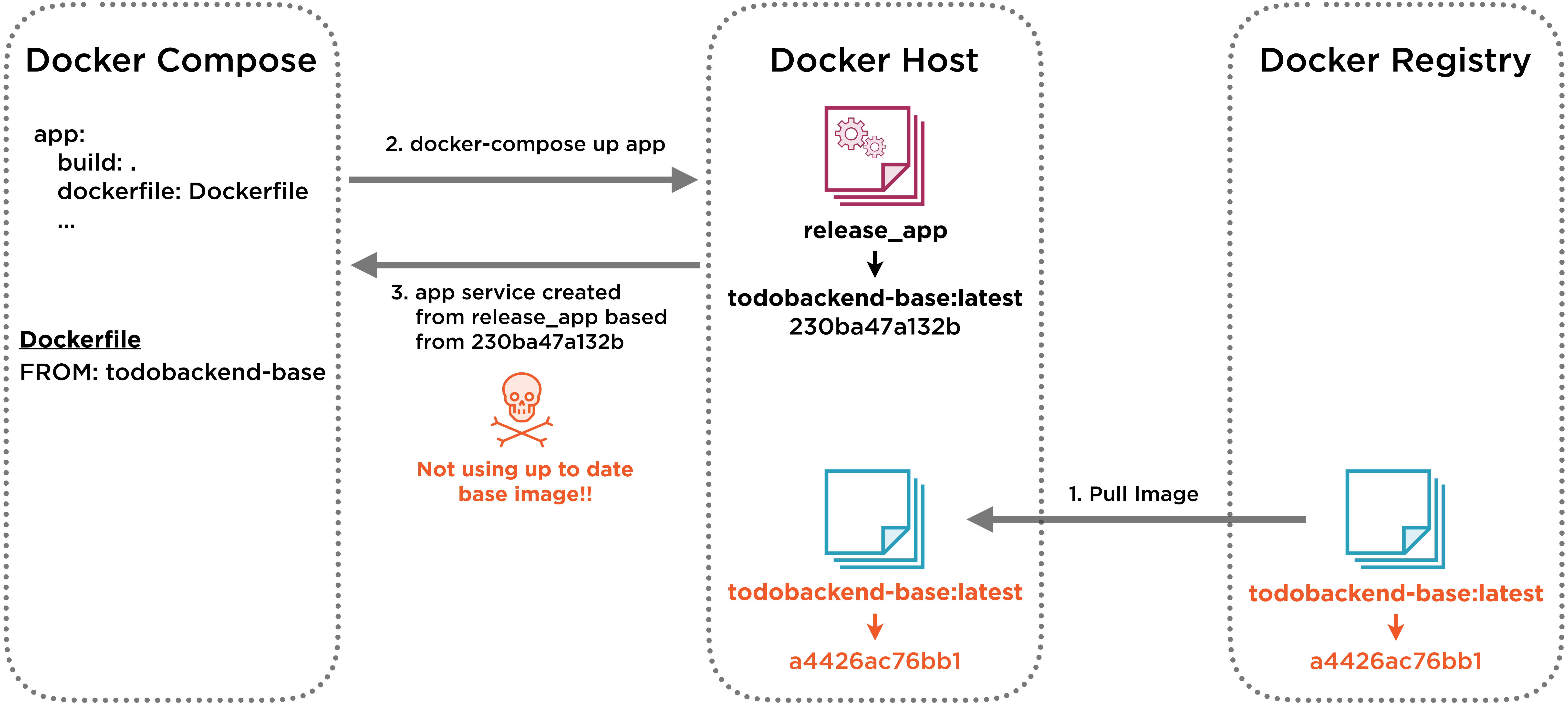
Docker Registry

todobackend-base:latest

↓

a4426ac76bb1

Building Up to Date Docker Images



Docker Compose

```
app:
   build: .
   dockerfile: Dockerfile
   ...
```

Dockerfile

FROM: todobackend-base

2. docker-compose up app

3. app service created from release_app based from 230ba47a132b

Not using up to date base image!!

Docker Host

release_app

↓

todobackend-base:latest

230ba47a132b

todobackend-base:latest

↓

a4426ac76bb1

1. Pull Image

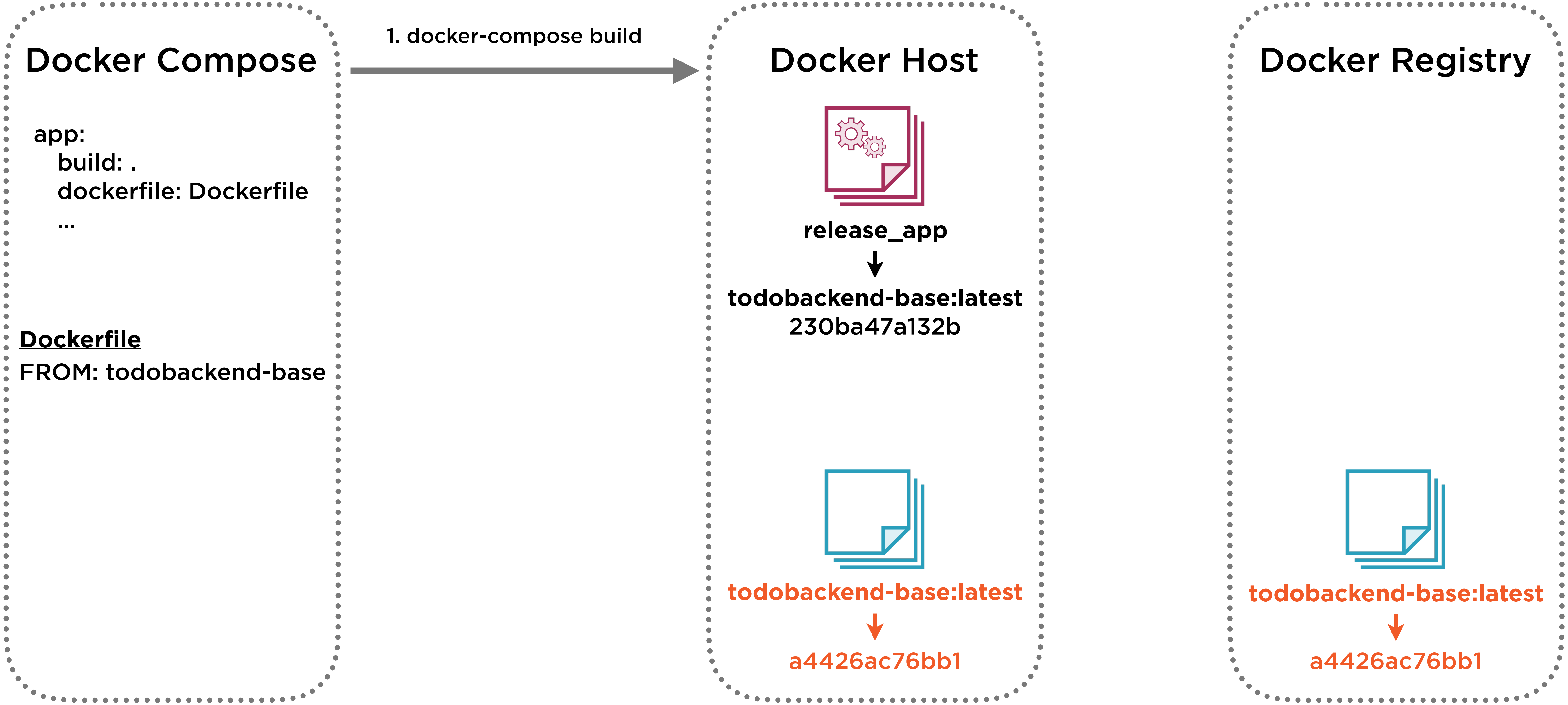
Docker Registry

todobackend-base:latest

↓

a4426ac76bb1

Building Up to Date Docker Images



Docker Compose

```
app:
   build: .
   dockerfile: Dockerfile
   ...
```

Dockerfile

FROM: todobackend-base

1. docker-compose build

Docker Host

release_app

↓

todobackend-base:latest

230ba47a132b

todobackend-base:latest

↓

a4426ac76bb1

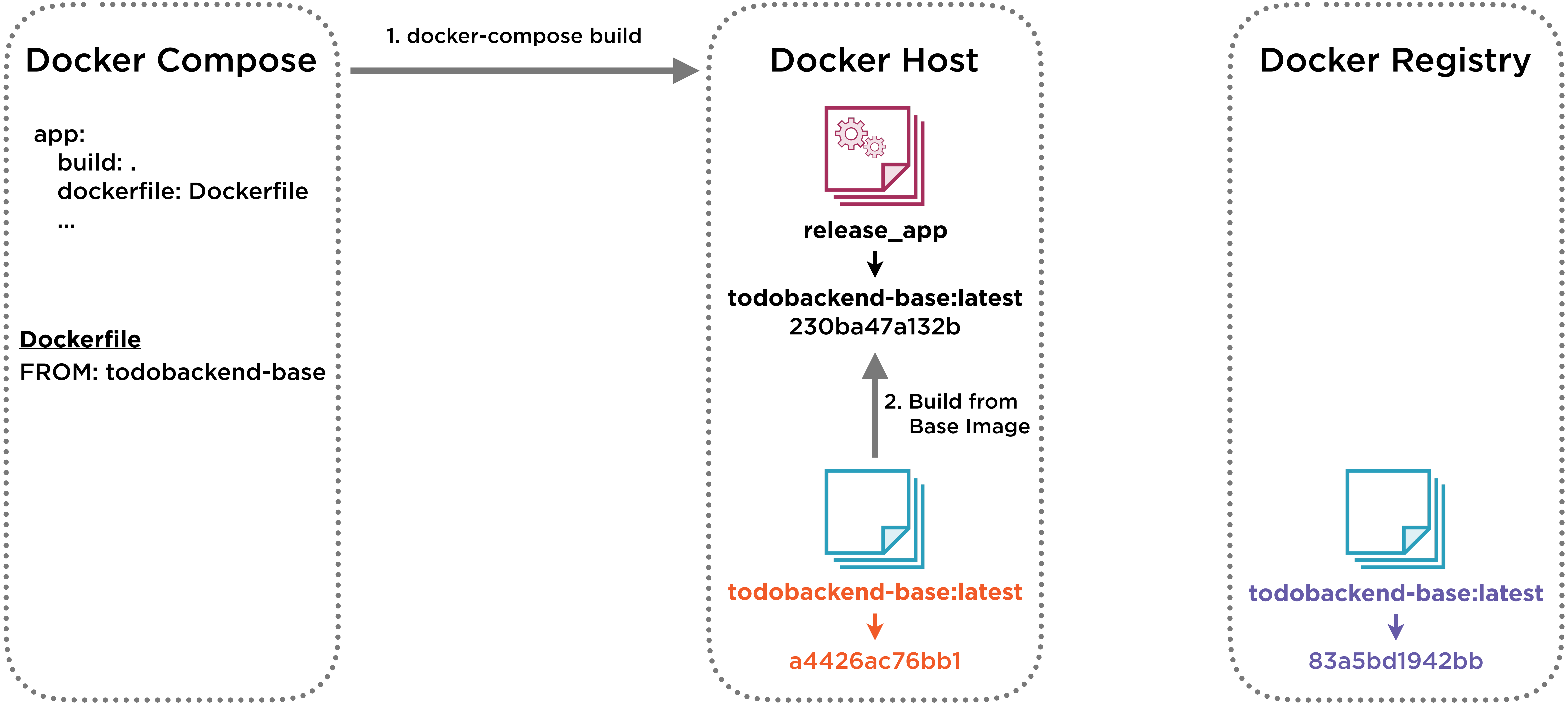
Docker Registry

todobackend-base:latest

↓

a4426ac76bb1

Building Up to Date Docker Images



Docker Compose

```
app:
  build: .
  dockerfile: Dockerfile
  ...
```

Dockerfile

FROM: todobackend-base

1. docker-compose build

Docker Host

release_app

todobackend-base:latest
230ba47a132b

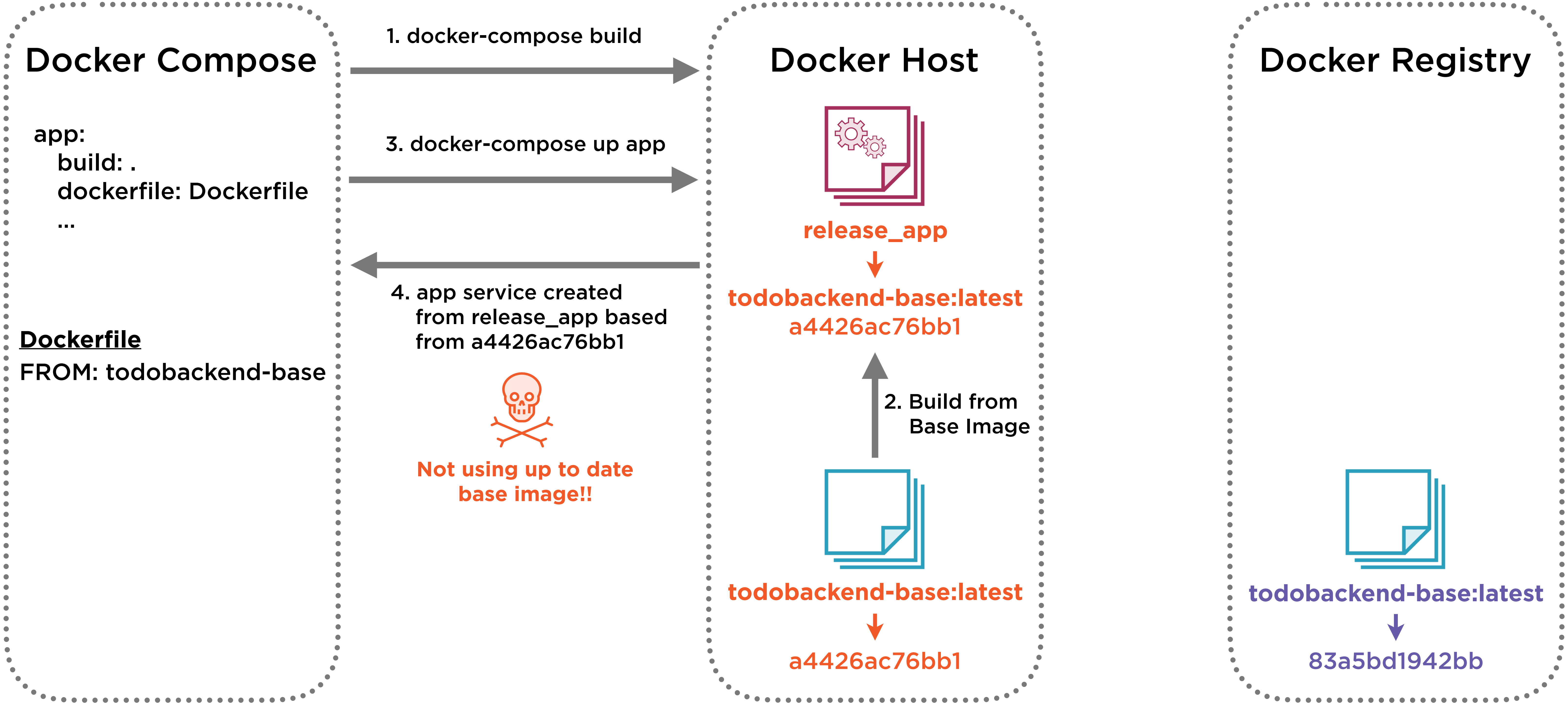
2. Build from
Base Image

todobackend-base:latest
a4426ac76bb1

Docker Registry

todobackend-base:latest
83a5bd1942bb

Building Up to Date Docker Images



Docker Compose

```
app:
   build: .
   dockerfile: Dockerfile
   ...
```

Dockerfile

FROM: todobackend-base

1. docker-compose build

3. docker-compose up app

4. app service created
   from release_app based
   from a4426ac76bb1

Not using up to date base image!!

Docker Host

release_app

todobackend-base:latest
a4426ac76bb1

2. Build from
   Base Image

todobackend-base:latest

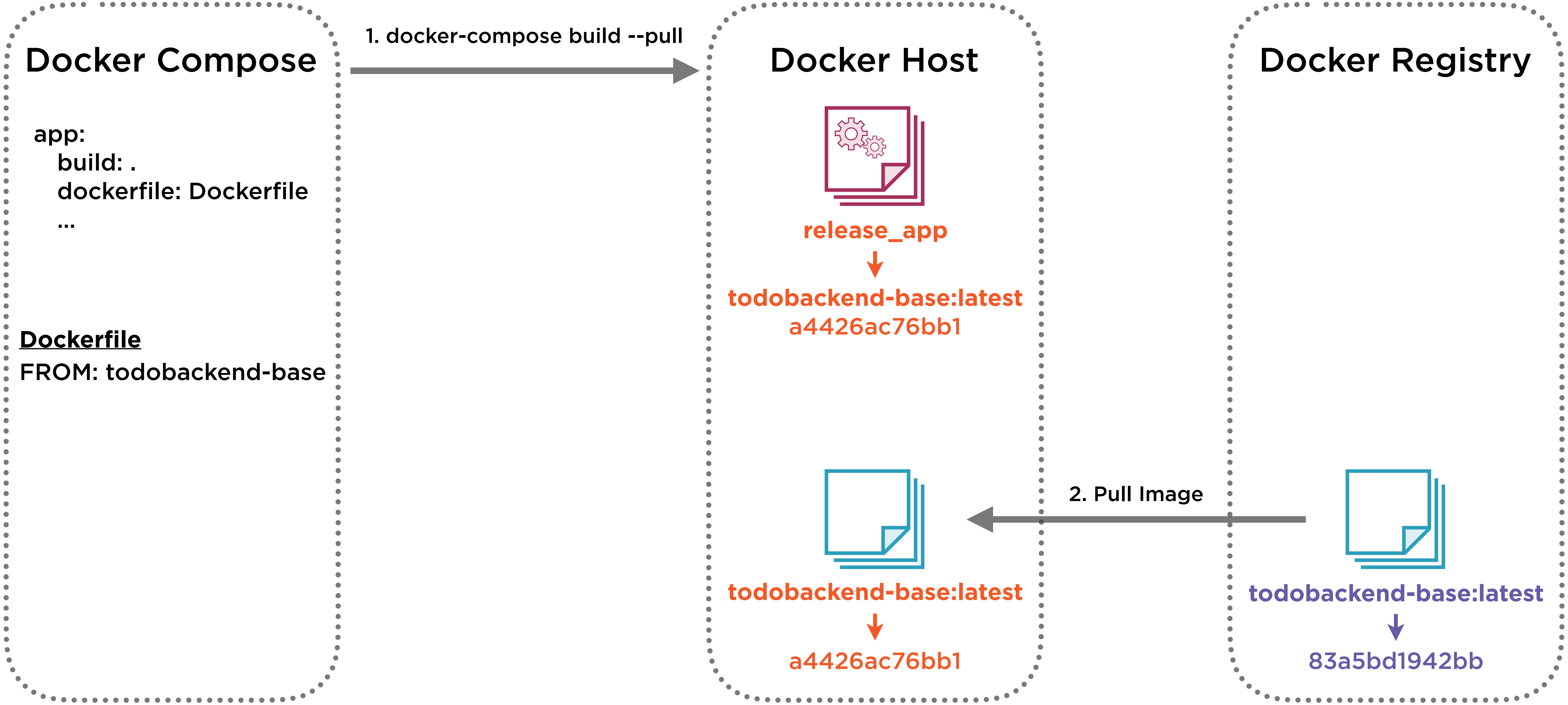
a4426ac76bb1

Docker Registry

todobackend-base:latest

83a5bd1942bb

Building Up to Date Docker Images



Docker Compose

```
app:
  build: .
  dockerfile: Dockerfile
  ...
```

Dockerfile

```
FROM: todobackend-base
```

1. docker-compose build --pull

Docker Host

release_app

todobackend-base:latest
a4426ac76bb1

todobackend-base:latest

a4426ac76bb1

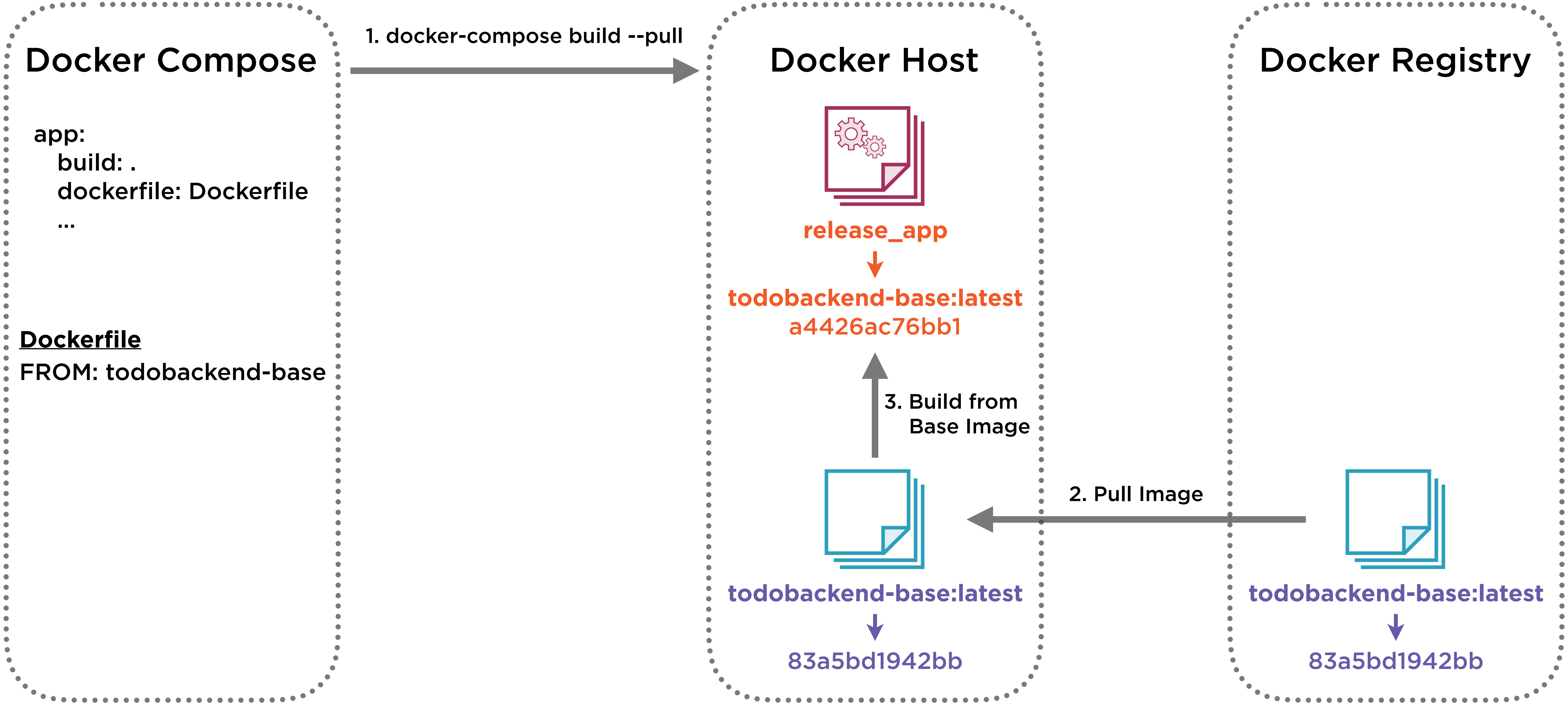
2. Pull Image

Docker Registry

todobackend-base:latest

83a5bd1942bb

Building Up to Date Docker Images



Docker Compose

```
app:
  build: .
  dockerfile: Dockerfile
  ...
```

Dockerfile

```
FROM: todobackend-base
```

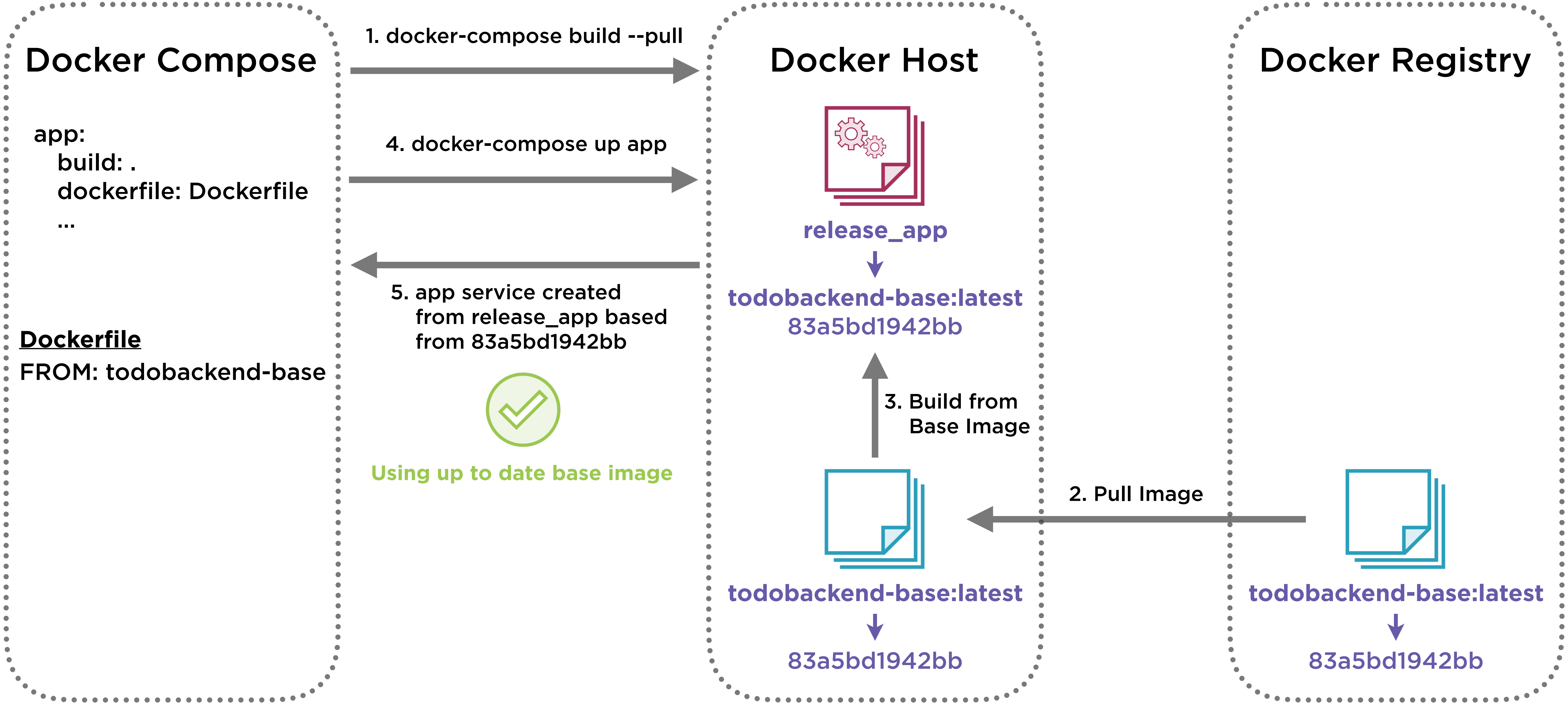
1. docker-compose build --pull

Docker Host

2. Pull Image

Docker Registry

Building Up to Date Docker Images



Docker Compose

```
app:
   build: .
   dockerfile: Dockerfile
   ...
```

Dockerfile

FROM: todobackend-base

1. docker-compose build --pull

4. docker-compose up app

5. app service created
   from release_app based
   from 83a5bd1942bb

Using up to date base image

Docker Host

release_app

todobackend-base:latest
83a5bd1942bb

3. Build from
   Base Image

todobackend-base:latest

83a5bd1942bb

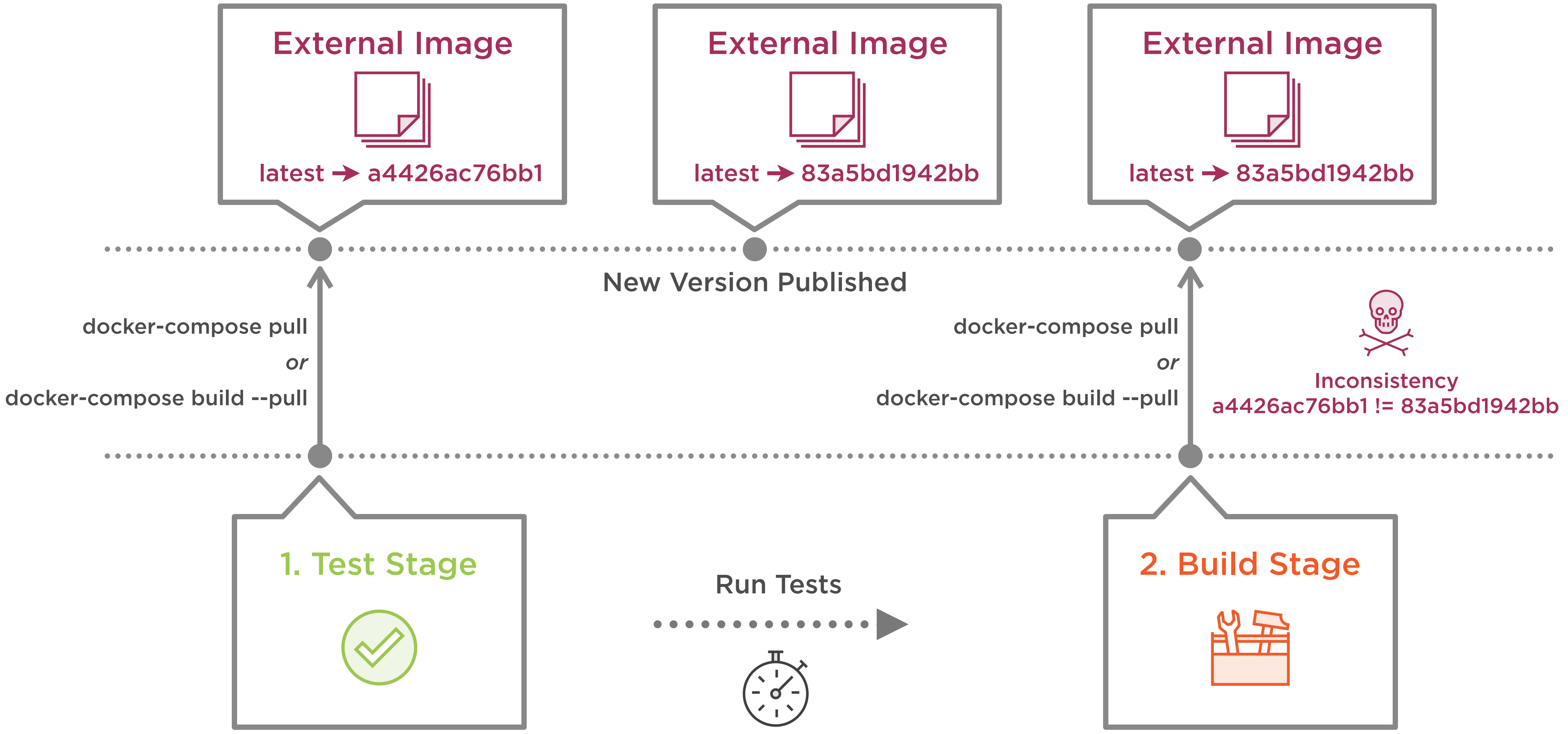
2. Pull Image

Docker Registry

todobackend-base:latest

83a5bd1942bb

Docker Image Consistency



External Image

latest → a4426ac76bb1

External Image

latest → 83a5bd1942bb

External Image

latest → 83a5bd1942bb

New Version Published

docker-compose pull

or

docker-compose build --pull

docker-compose pull

or

docker-compose build --pull

Inconsistency

a4426ac76bb1 != 83a5bd1942bb

1. Test Stage

Run Tests

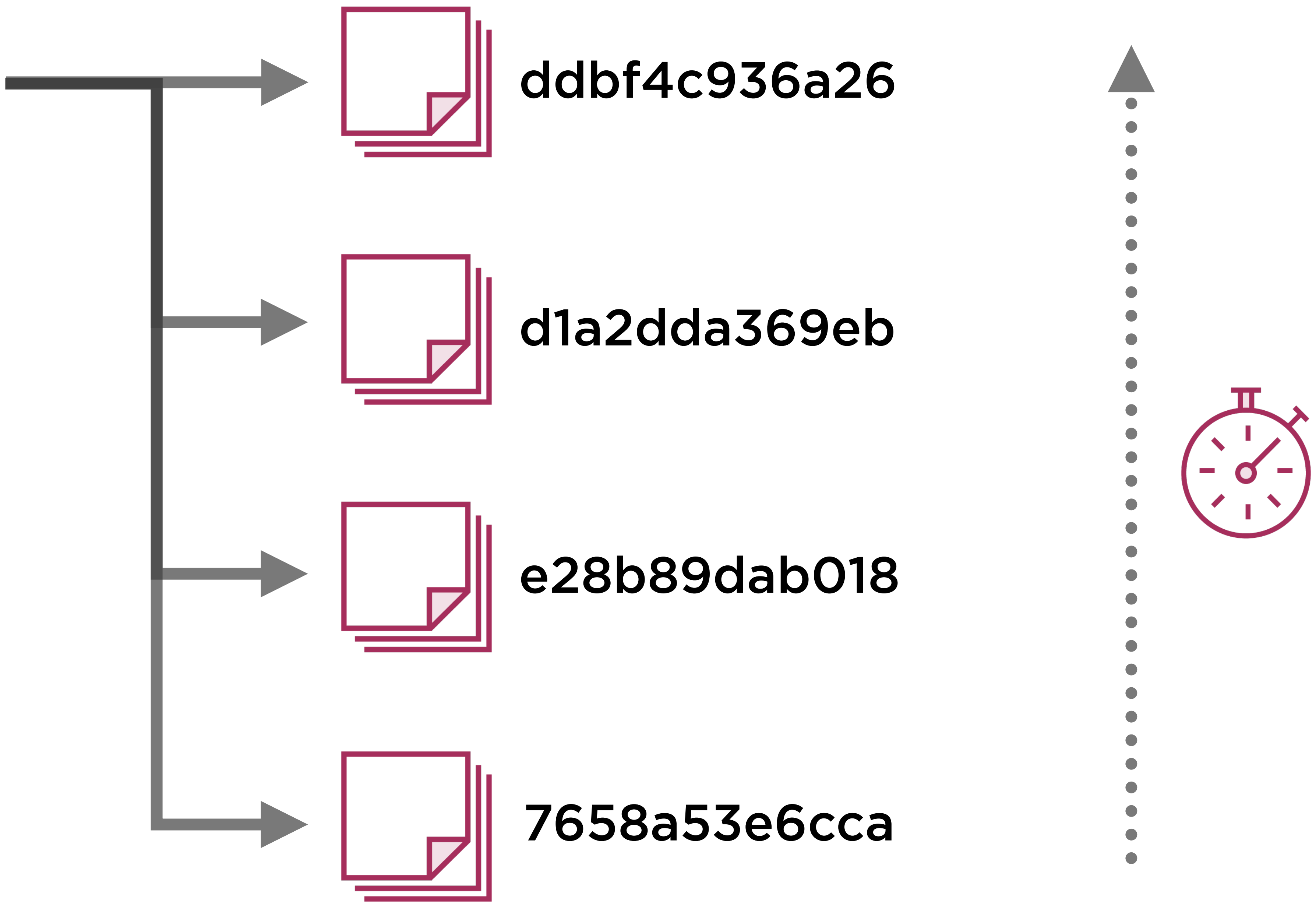
2. Build Stage

Pulling the Latest Images with Consistency

Tagging the Release Image

Logical Tags

Repository	Tag
jmenga/todobackend	latest



ddbf4c936a26

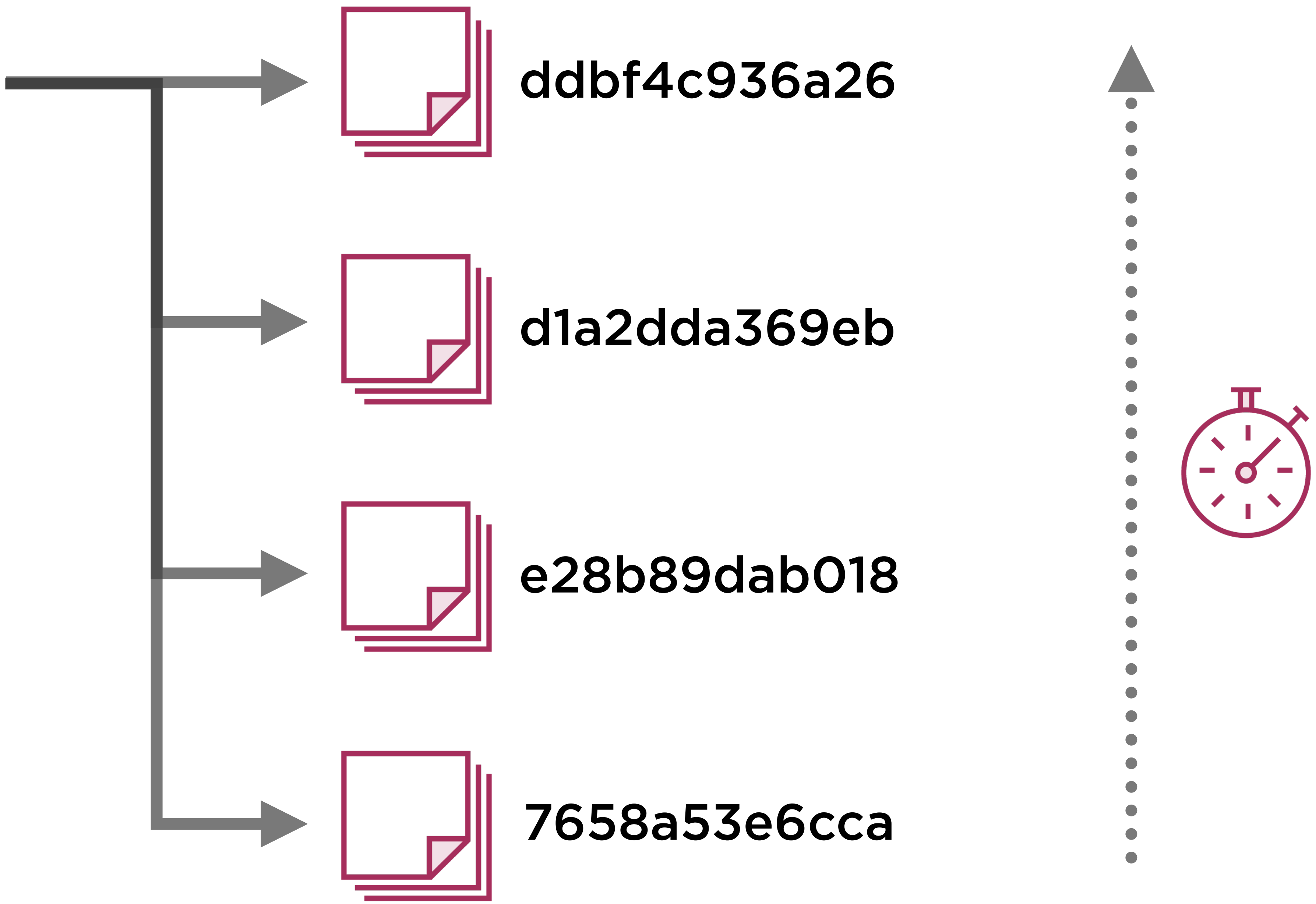
d1a2dda369eb

e28b89dab018

7658a53e6cca

Logical Tags

Repository	Tag
ubuntu	14.04.3



ddbf4c936a26

d1a2dda369eb

e28b89dab018

7658a53e6cca

Build Tags

Repository	Tag	
jmenga/todobackend	0.1.0.201602160847	ddbf4c936a26
jmenga/todobackend	0.1.0.201602151735	d1a2dda369eb
jmenga/todobackend	0.1.0.201602151304	e28b89dab018
jmenga/todobackend	0.1.0.201602151244	7658a53e6cca

Tagging Strategy

make tag

Creates a logical tag that points to the most up-to-date image representing the logical tag

make buildtag

Creates a build tag that points to a specific image created at a specific point in time

make tag

```
$ make tag 0.1 latest
=> Tagging release image with
tags 0.1 latest…
=> Tagging complete

$ docker images
REPOSITORY                TAG
my_org/my_app             0.1
my_org/my_app             latest
…
…
```

- Tag image with one or more tags

- Fully qualified name generated from DOCKER_REGISTRY, ORG_NAME and REPO_NAME environment variables

make buildtag

```
$ make buildtag 0.1 master
=> Tagging release image with
suffix 201602171133 and build
tags 0.1 master...
=> Tagging complete

$ docker images
REPOSITORY   TAG
org/app      0.1.201602171133
org/app      master.201602171133
…
…
```

◀ Tag image with one or more build tags. Suffix generated by BUILD_TAG_EXPRESSION environment variable.

◀ Fully qualified name generated from DOCKER_REGISTRY, ORG_NAME and REPO_NAME environment variables

Publishing the Release Image

Docker Compose v2 Specification

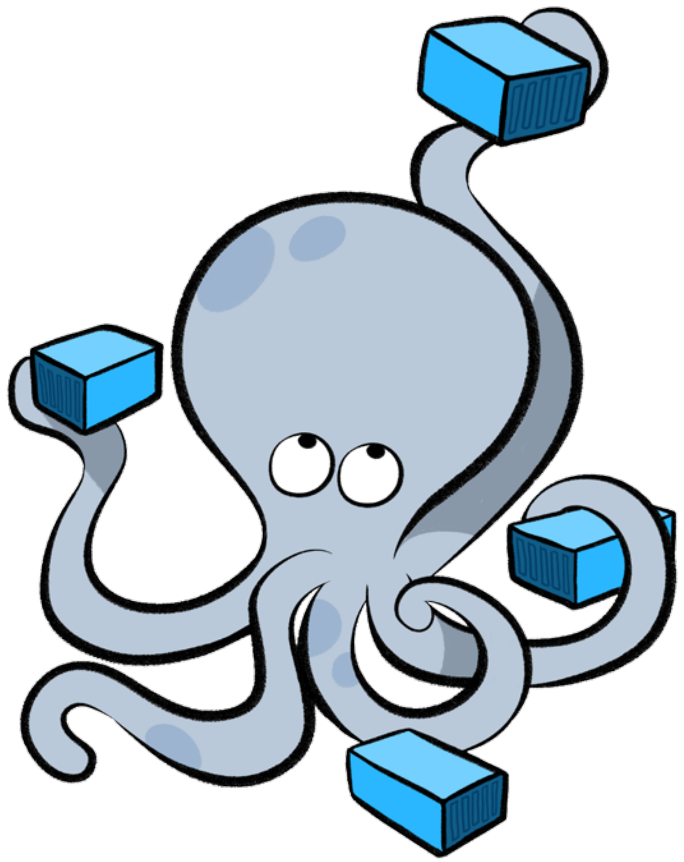
Docker Building Blocks

Services

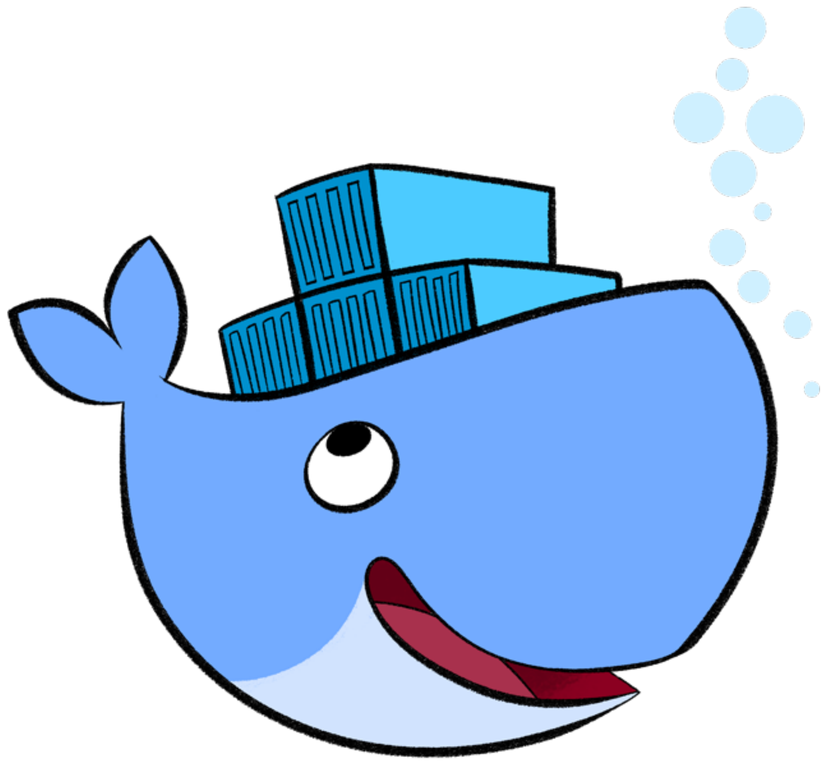
Volumes

Networks

Docker Compose v2 Specification Requirements



Docker Compose 1.6+



Docker Engine 1.10+

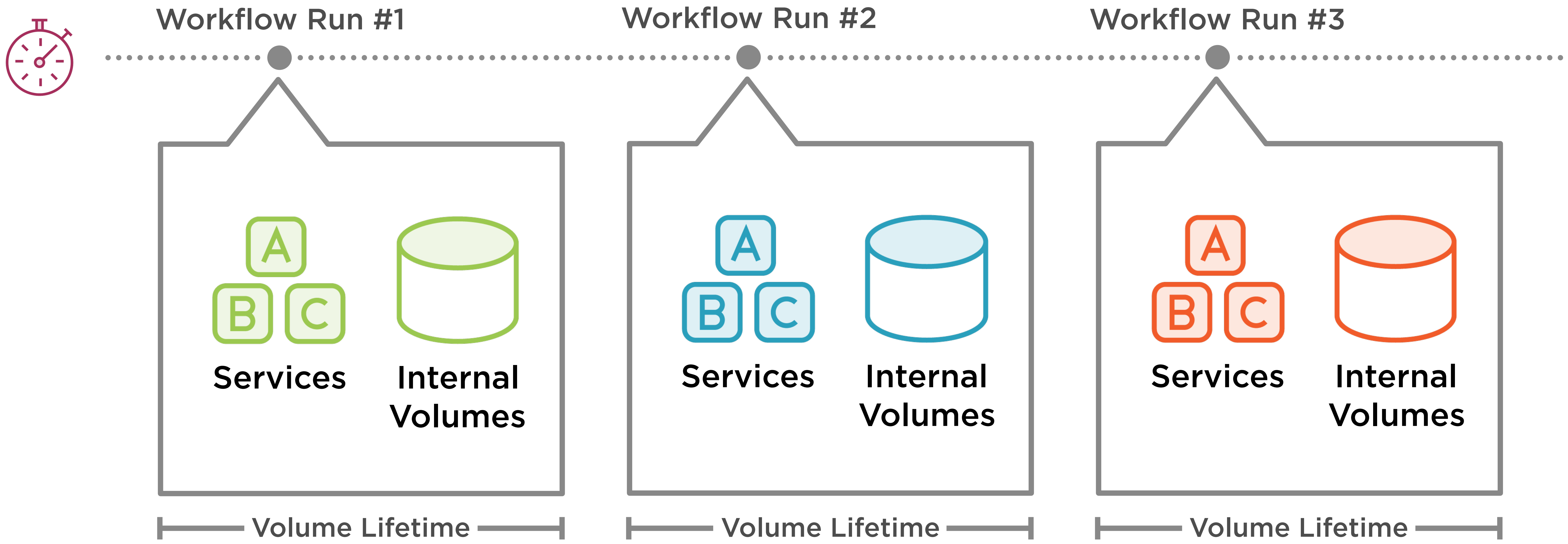
Docker Compose 1.6 Introduction

- https://blog.docker.com/2016/02/compose-1-6/
- https://youtu.be/EReEOMS7gsk

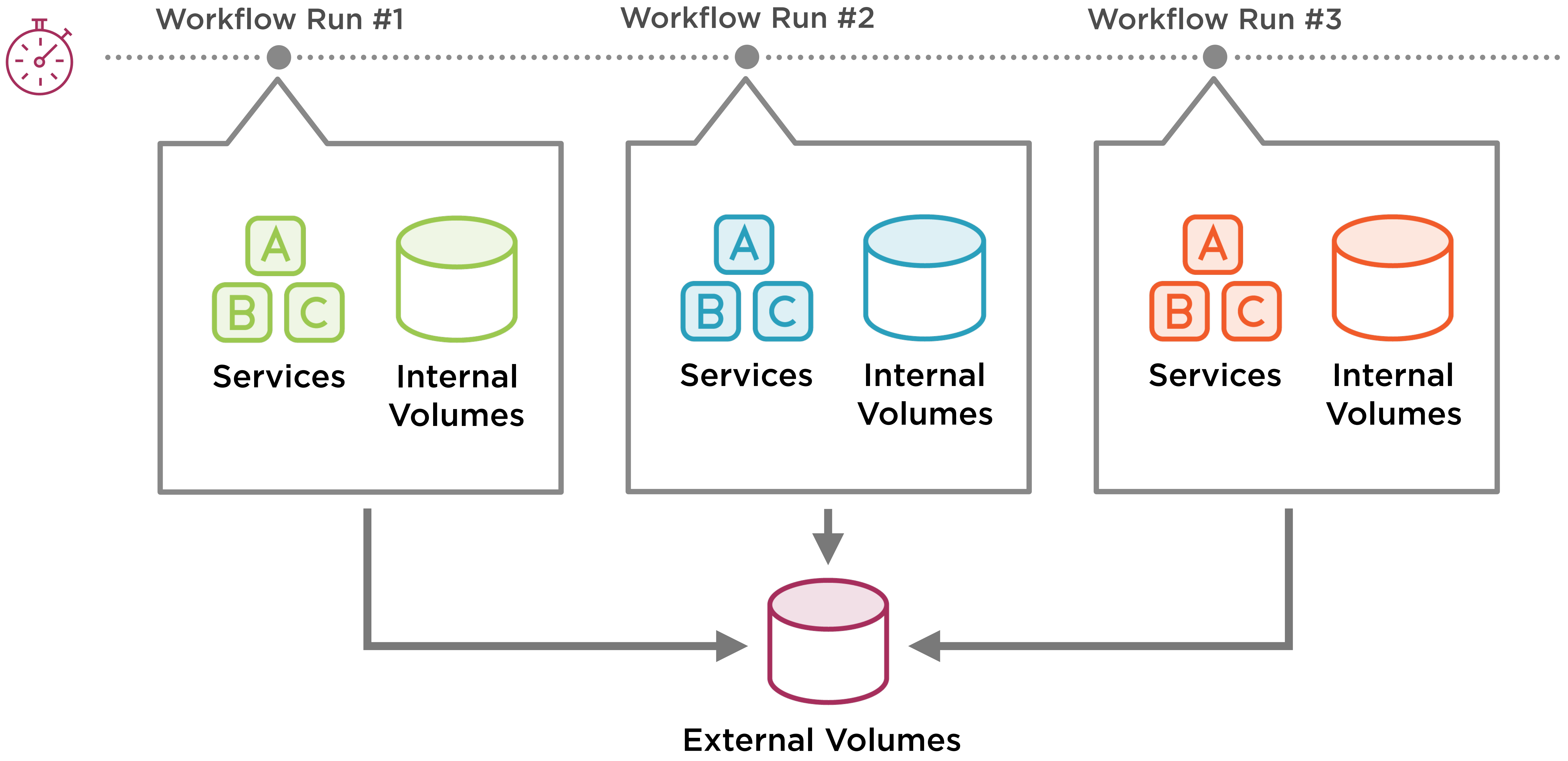
Docker Compose File Reference

- https://docs.docker.com/compose/compose-file/

Docker Compose Internal Volumes

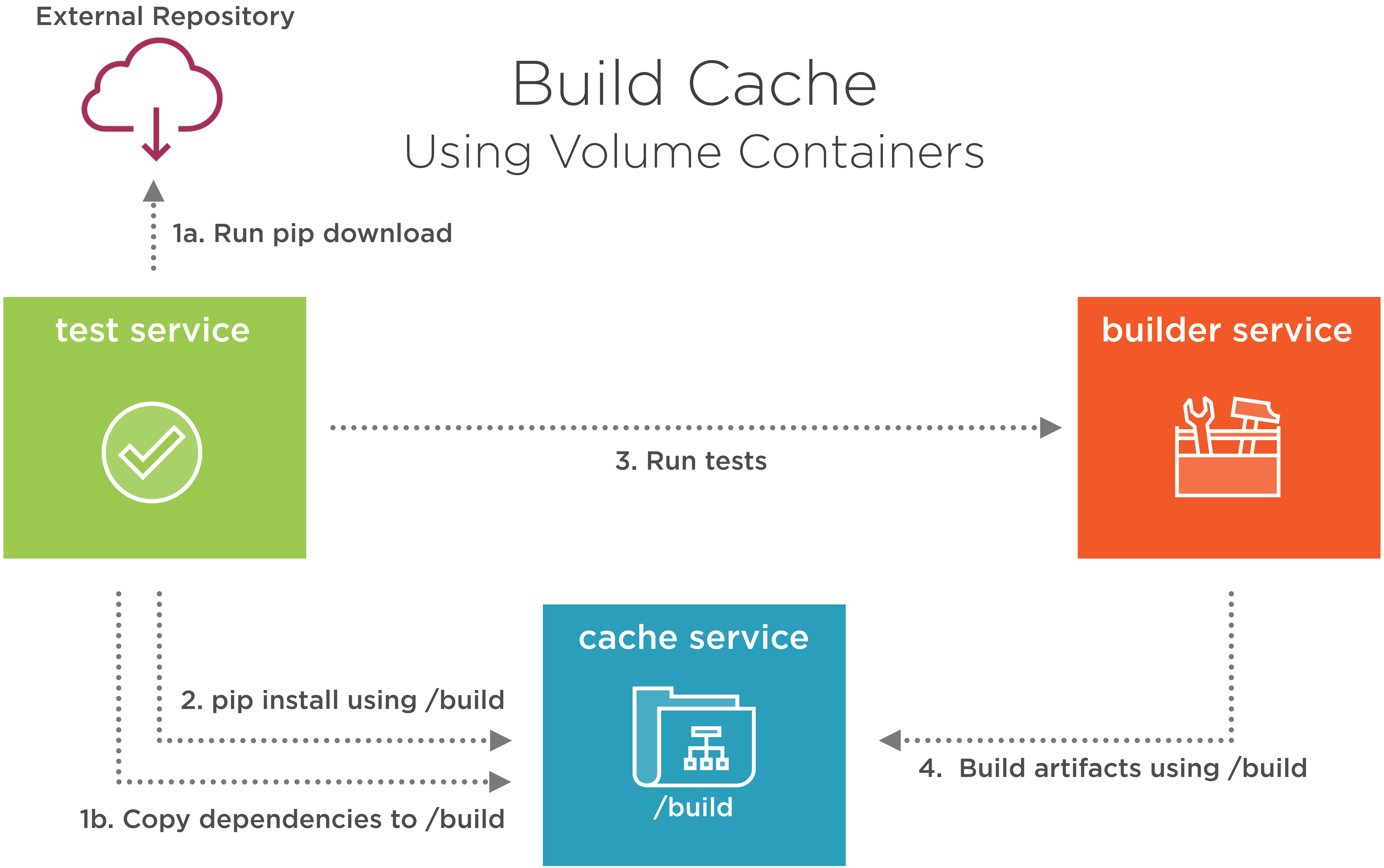


Docker Compose External Volumes



Build Cache

Using Volume Containers



External Repository

1a. Run pip download

test service

builder service

3. Run tests

cache service

/build

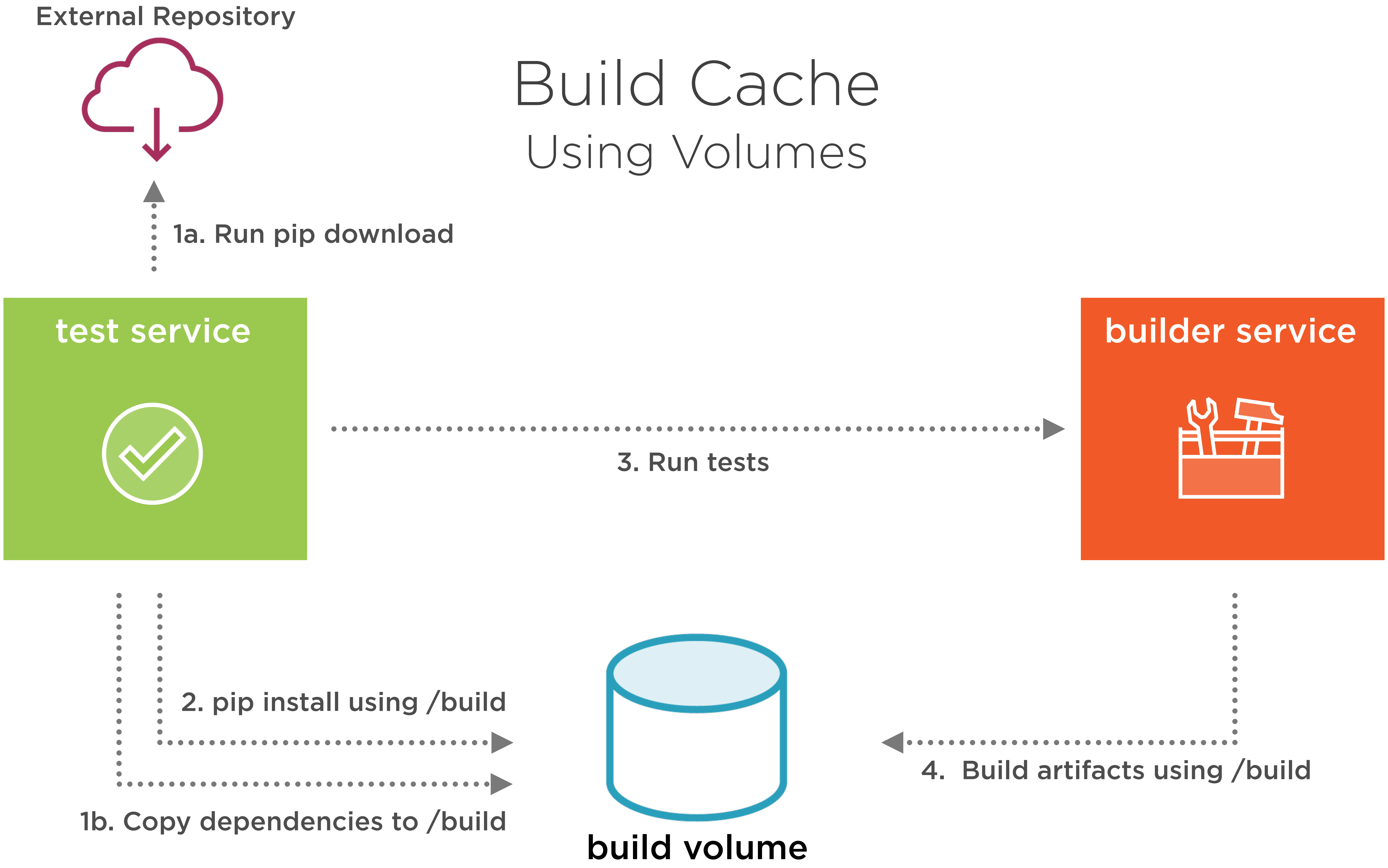
2. pip install using /build

1b. Copy dependencies to /build

4. Build artifacts using /build

Build Cache

Using Volumes



External Repository

1a. Run pip download

test service

builder service

3. Run tests

2. pip install using /build

1b. Copy dependencies to /build

build volume

4. Build artifacts using /build

Summary

Production Ready Workflow

- Improved user feedback
- Publishing test reports
- Handling errors and ensuring consistency
- Self contained
- Tagging and publishing

Docker Compose v2 Specification

- Expresses top-level services, volumes and networks