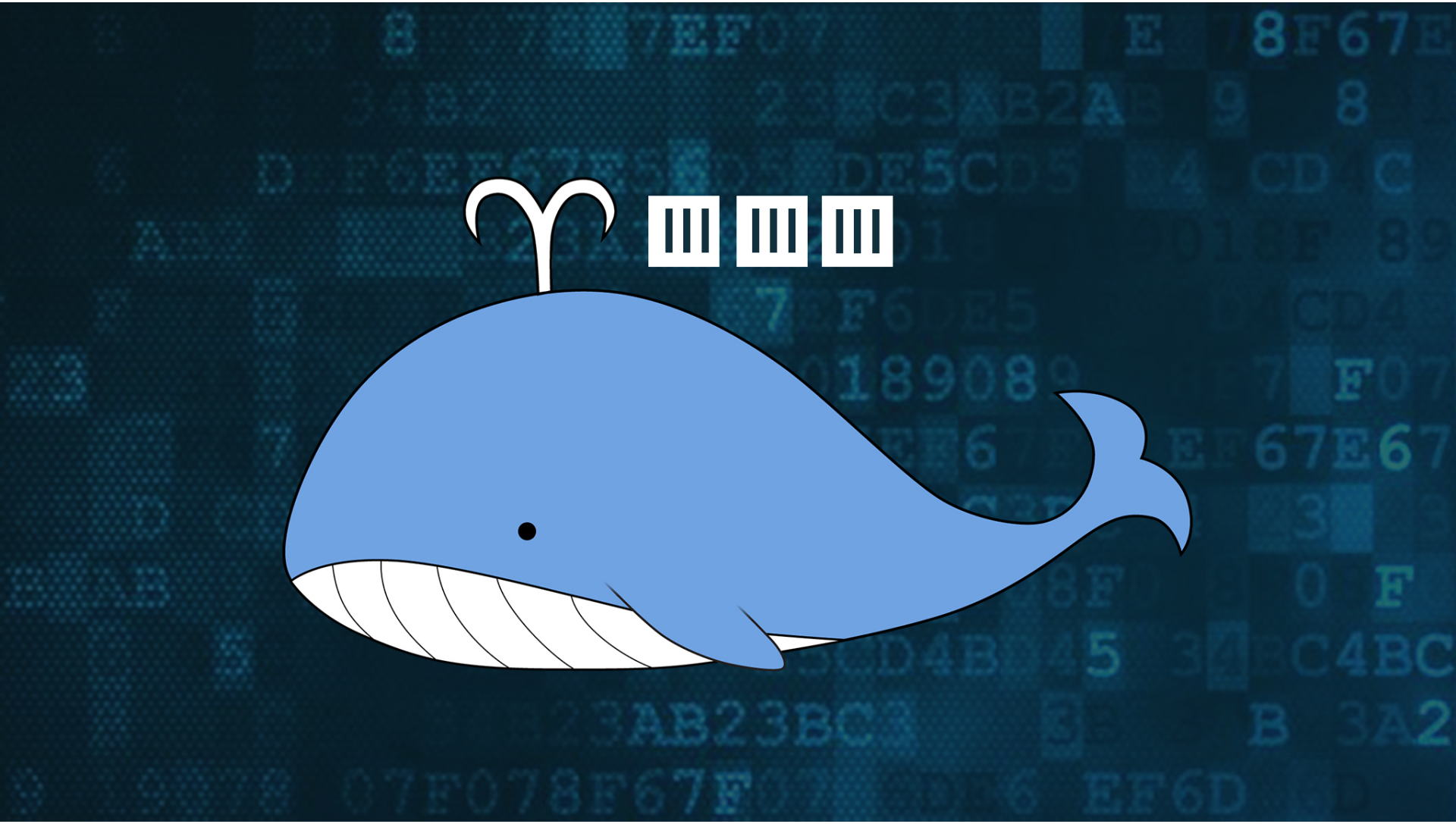


LEVEL UP
www.level-up.one

Docker technology

is one implementation of container based virtualization technologies

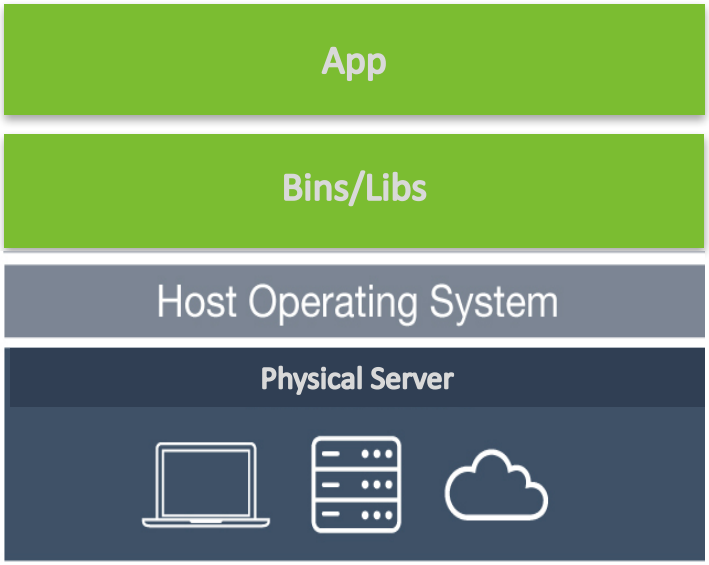
LEVEL UP
www.level-up.one

LEVEL UP
www.level-up.one

Introduction to Virtualization Technologies

LEVEL UP
www.level-up.one

Pre-Virtualization World



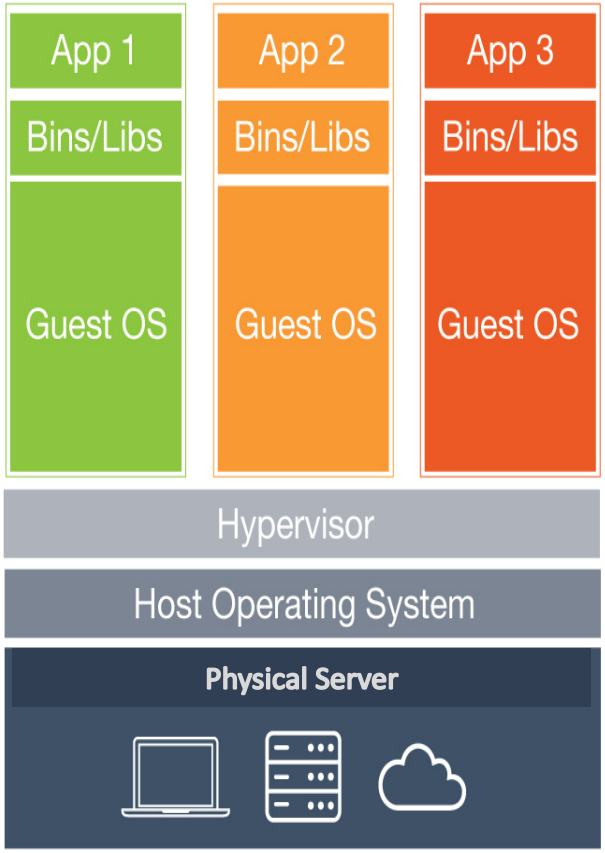
Pre-Virtualization

Problems:

- Huge Cost
- Slow Deployment
- Hard to Migrate

LEVEL UP
www.level-up.one

Hypervisor-based Virtualization



Hypervisor-based Virtualization

Benefits:

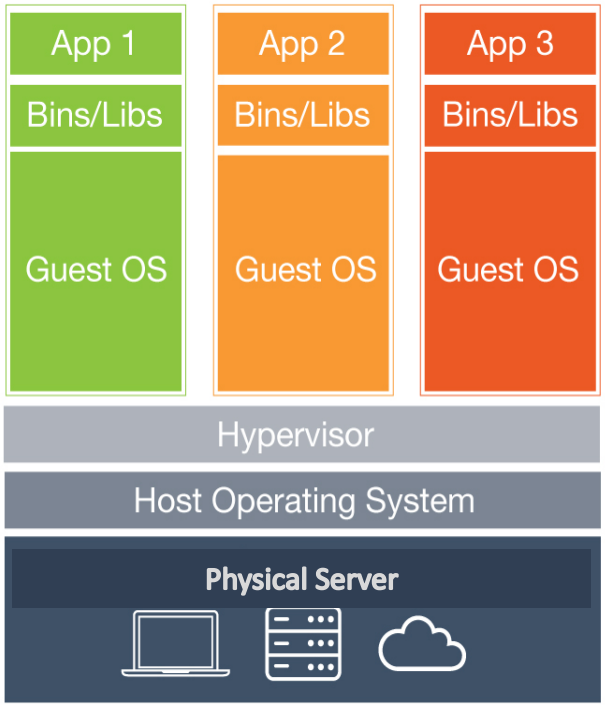
- Cost-Efficient
- Easy to Scale

Limitations:

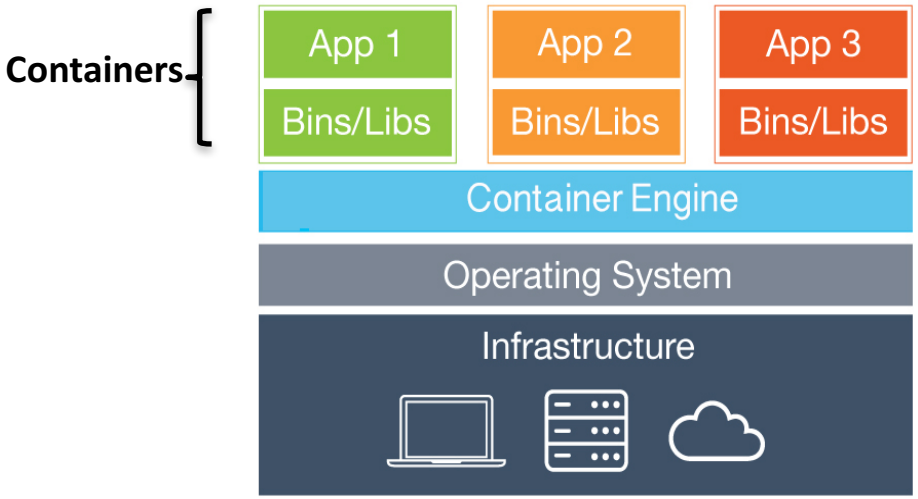
- Kernel Resource Duplication
- Application Portability Issue

LEVEL UP
www.level-up.one

Hypervisor-based *VS* Container-based Virtualization



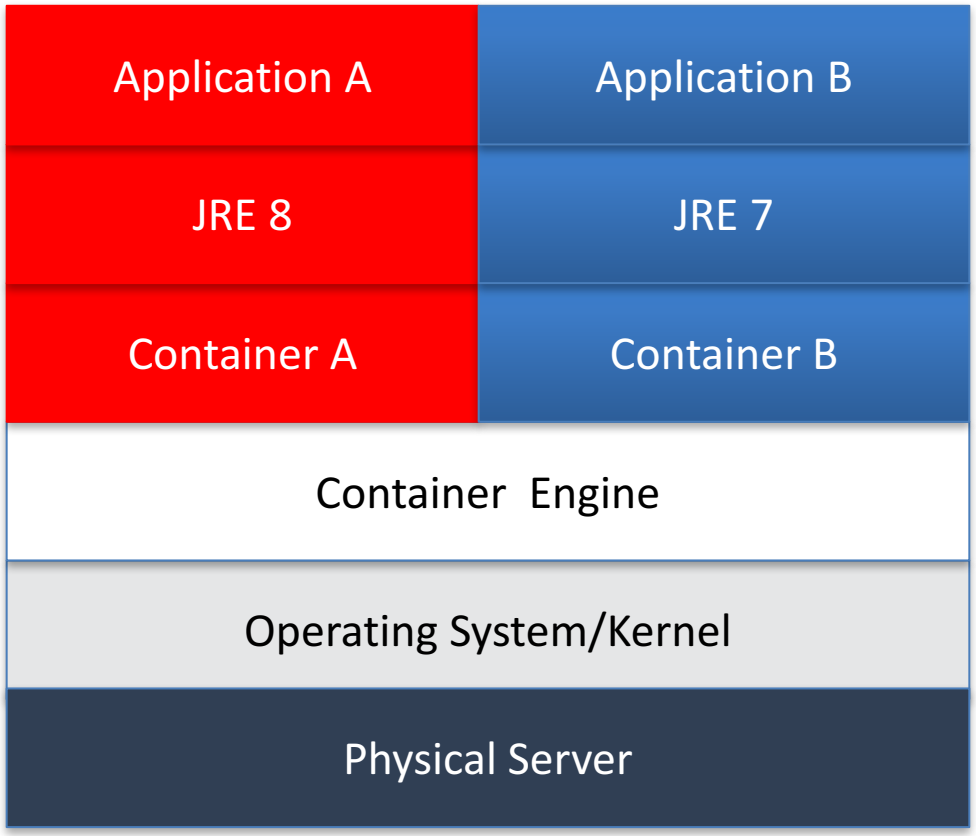
Hypervisor-based Virtualization



Container-based Virtualization

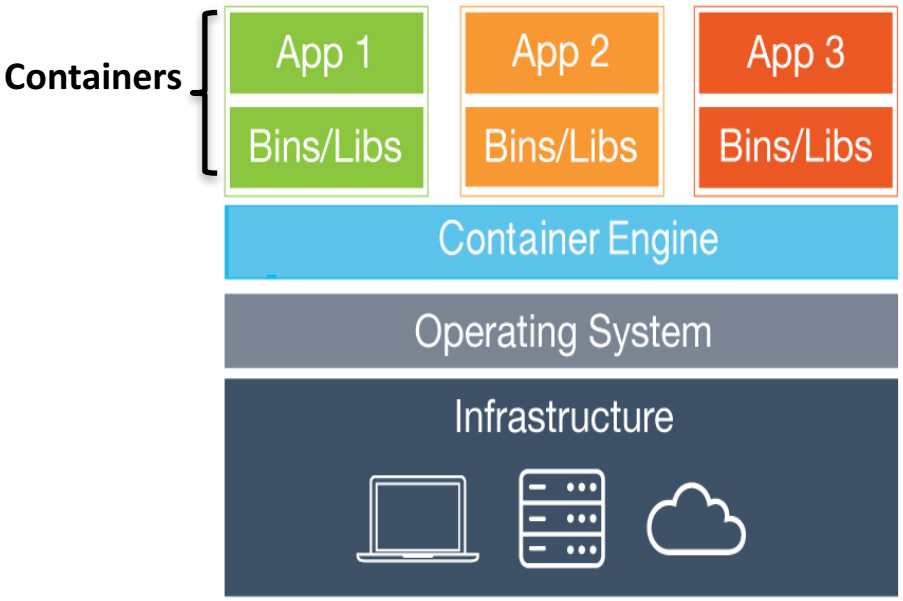
LEVEL UP
www.level-up.one

Runtime Isolation



LEVEL UP
www.level-up.one

Container Virtualization

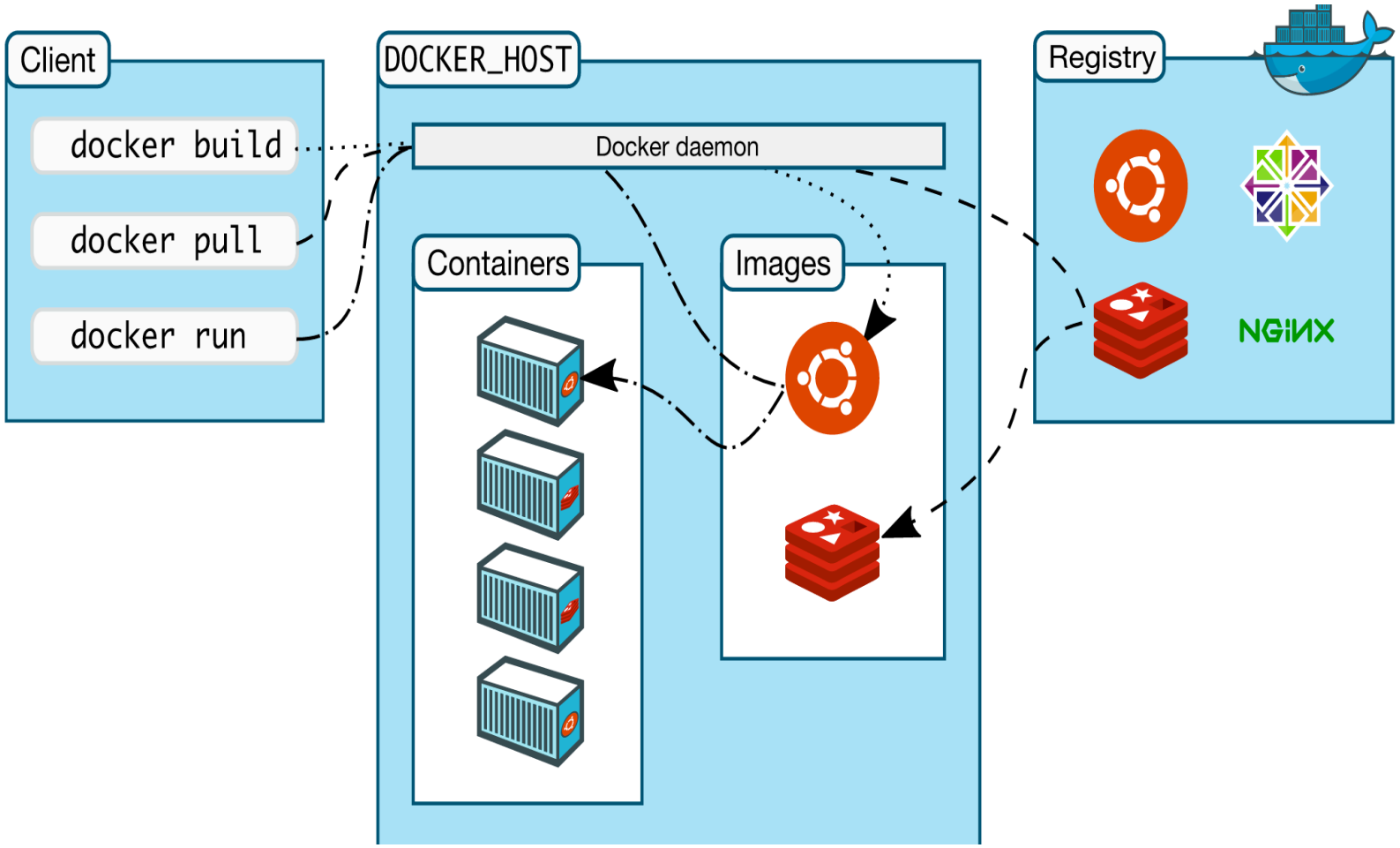


Container Virtualization

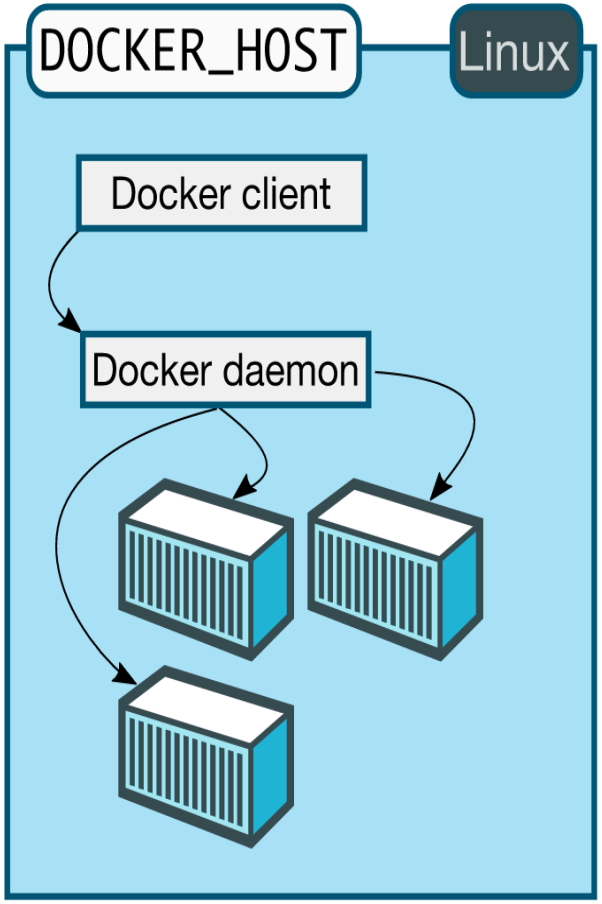
Benefits:

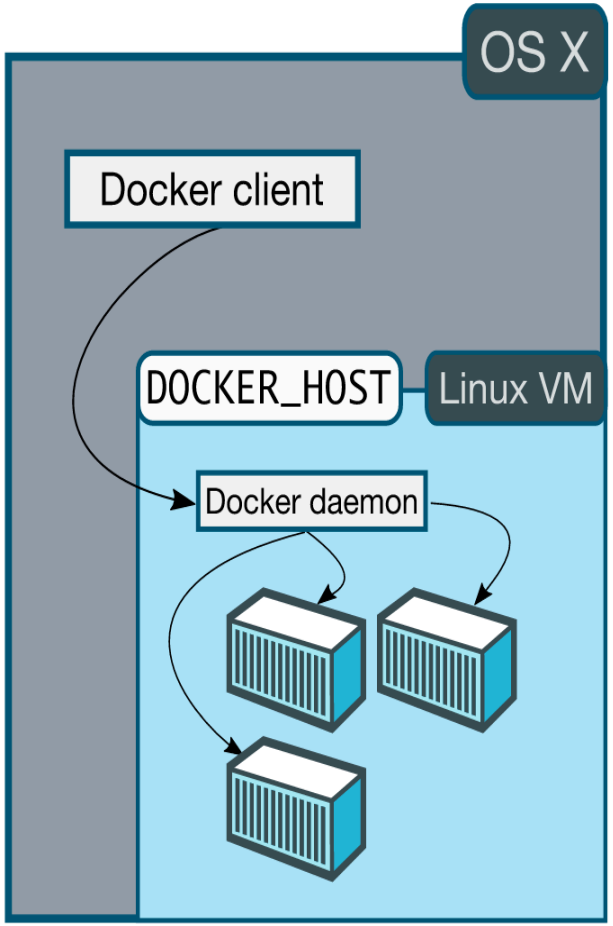
- Cost-Efficient
- Fast Deployment
- Guaranteed Portability

LEVEL UP
www.level-up.one



LEVEL UP
www.level-up.one





LEVEL UP
www.level-up.one

Docker

Client-Server Architecture

LEVEL UP
www.level-up.one

Install Docker for Mac/Windows

LEVEL UP
www.level-up.one

Install Docker Software

This lecture applies to you if:

- **You are using Linux**
- **Or you are using Mac and your Mac version is OS X 10.10.3 or newer**
- **Or you are using Windows and your Windows version is Windows 10 or newer**

Otherwise, you can skip this lecture and follow the installation guide of the next lecture.

LEVEL UP
www.level-up.one

Install Docker Toolbox

LEVEL UP
www.level-up.one

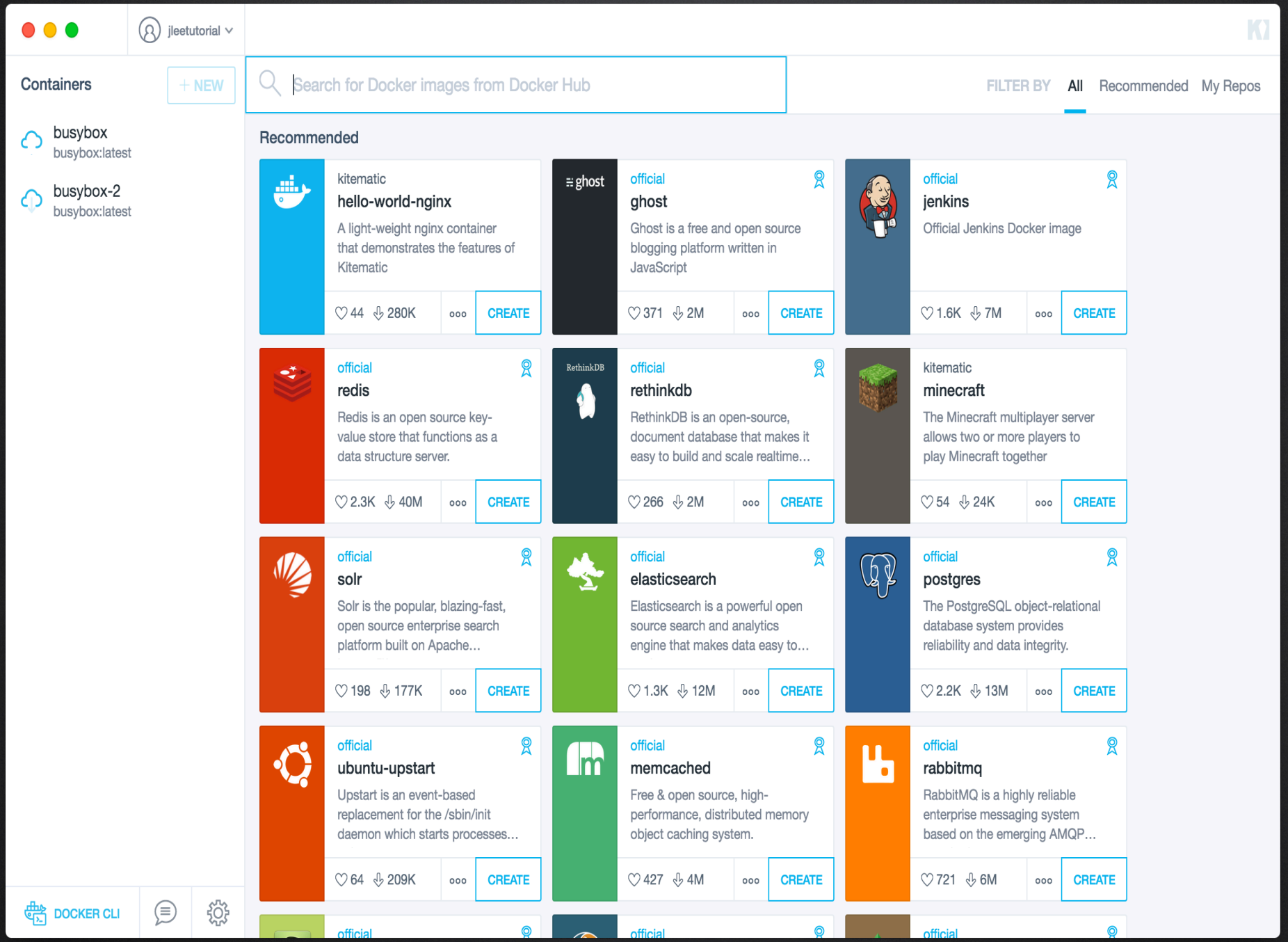
Install Docker Toolbox

This lecture applies to you if:

- **You are using Mac and your Mac version is older than OS X 10.10.3.**
- **Or you are using Windows and your Windows version is older than Windows 10.**
- **Or you want to install Docker Compose, Docker Machine or Kitematic instead of Docker Engine.**

Otherwise, you can skip this lecture and follow the installation guide of the previous lecture.

If you already installed Docker for Mac/Windows, you can skip this lecture for now.


jleetutorial
Containers
+ NEW
busybox
busybox:latest
busybox-2
busybox:latest
DOCKER CLI
Search for Docker images from Docker Hub
FILTER BY
All
Recommended
My Repos
Recommended
kitematic
hello-world-nginx
A light-weight nginx container that demonstrates the features of Kitematic
44 280K
CREATE
official
ghost
Ghost is a free and open source blogging platform written in JavaScript
371 2M
CREATE
official
jenkins
Official Jenkins Docker image
1.6K 7M
CREATE
official
redis
Redis is an open source key-value store that functions as a data structure server.
2.3K 40M
CREATE
official
rethinkdb
RethinkDB is an open-source, document database that makes it easy to build and scale realtime...
266 2M
CREATE
kitematic
minecraft
The Minecraft multiplayer server allows two or more players to play Minecraft together
54 24K
CREATE
official
solr
Solr is the popular, blazing-fast, open source enterprise search platform built on Apache...
198 177K
CREATE
official
elasticsearch
Elasticsearch is a powerful open source search and analytics engine that makes data easy to...
1.3K 12M
CREATE
official
postgres
The PostgreSQL object-relational database system provides reliability and data integrity.
2.2K 13M
CREATE
official
ubuntu-upstart
Upstart is an event-based replacement for the /sbin/init daemon which starts processes...
64 209K
CREATE
official
memcached
Free & open source, high-performance, distributed memory object caching system.
427 4M
CREATE
official
rabbitmq
RabbitMQ is a highly reliable enterprise messaging system based on the emerging AMQP...
721 6M
CREATE

Important
Docker Concepts

LEVEL UP
www.level-up.one

Images

- Images are read only templates used to create containers.
- Images are created with the docker build command, either by us or by other docker users.
- Images are composed of layers of other images.
- Images are stored in a Docker registry.

LEVEL UP
www.level-up.one

Containers

- If an image is a class, then a container is an instance of a class - a runtime object.
- Containers are lightweight and portable encapsulations of an environment in which to run applications.
- Containers are created from images. Inside a container, it has all the binaries and dependencies needed to run the application.

LEVEL UP
www.level-up.one

Registries and Repositories

- A registry is where we store our images.
- You can host your own registry, or you can use Docker's public registry which is called DockerHub.
- Inside a registry, images are stored in repositories.
- Docker repository is a collection of different docker images with the same name, that have different tags, each tag usually represents a different version of the image.

LEVEL UP
www.level-up.one

Why Using Official Images

- **Clear Documentation**
- **Dedicated Team for Reviewing Image Content**
- **Security Update in a Timely Manner**

LEVEL UP
www.level-up.one

Run our First Hello World Docker Container

LEVEL UP
www.level-up.one

Deep Dive into Docker Containers

- **running containers in detached mode**
- **docker ps command**
- **docker container name**
- **docker inspect command**

LEVEL UP
www.level-up.one

	Run Container in Foreground	Run Container in Background
Description	Docker run starts the process in the container and attaches the console to the process's standard input, output, and standard error.	Containers started in detached mode and exit when the root process used to run the container exits.
How to specify?	default mode	-d option
Can the console be used for other commands after the container is started up?	No	Yes

LEVEL UP
www.level-up.one

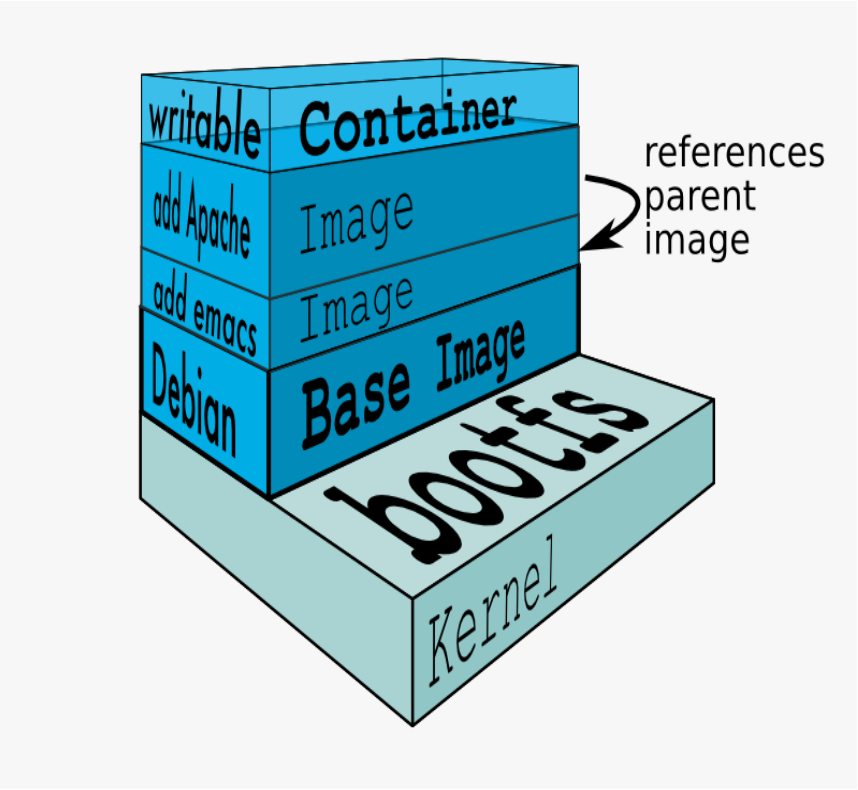
Docker Port Mapping and Docker Logs

LEVEL UP
www.level-up.one

Docker Image Layers

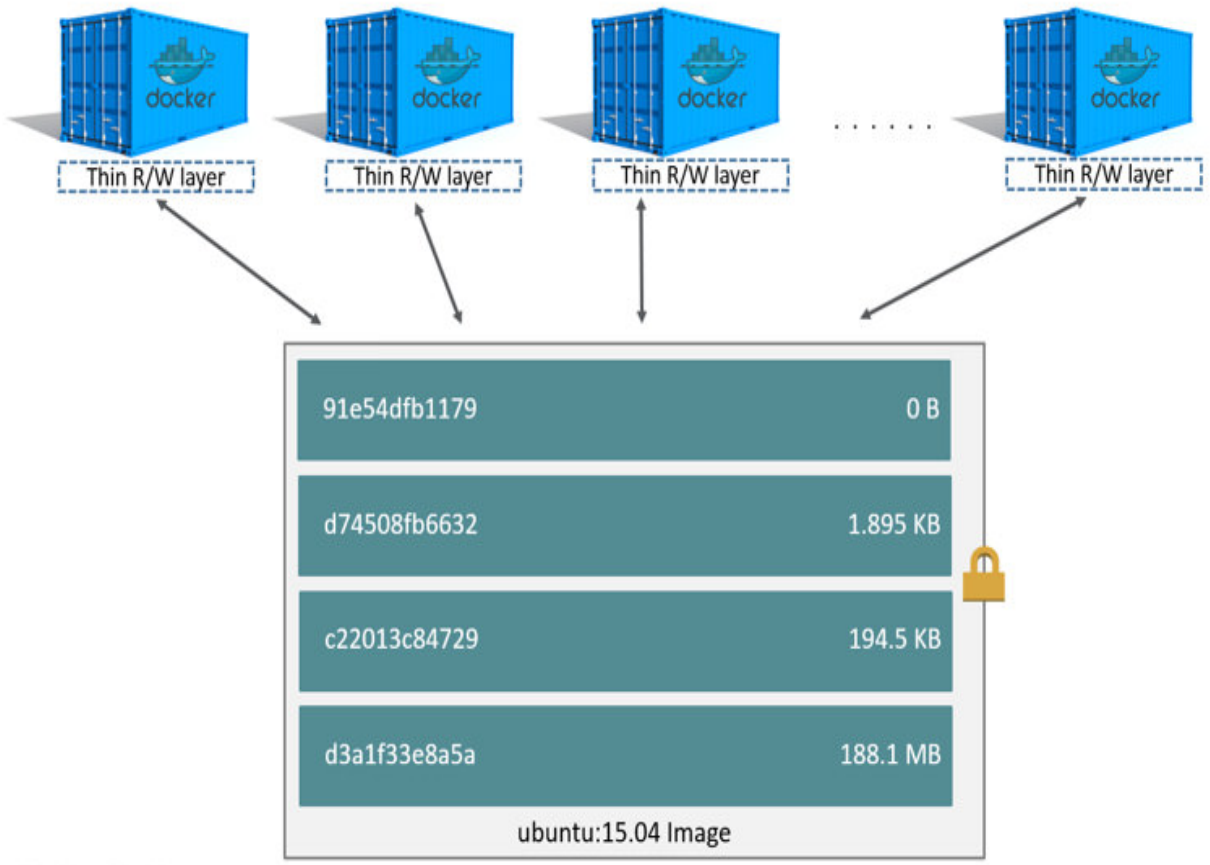
LEVEL UP
www.level-up.one

Image Layers



LEVEL UP
www.level-up.one

Image Layers



LEVEL UP
www.level-up.one

Build Docker Images

Approach 1: committing changes made in a container

LEVEL UP
www.level-up.one

Ways to Build a Docker Image

- Commit changes made in a Docker container.
- Write a Dockerfile.

LEVEL UP
www.level-up.one

Steps

1. Spin up a container from a base image.
2. Install Git package in the container.
3. Commit changes made in the container.

LEVEL UP
www.level-up.one

Docker commit

- Docker commit command would save the changes we made to the Docker container's file system to a new image.

docker commit container_ID repository_name:tag

LEVEL UP
www.level-up.one

Build Docker Images

Approach 2: Writing a Dockerfile

LEVEL UP
www.level-up.one

Dockerfile and Instructions

- A Dockerfile is a text document that contains all the instructions users provide to assemble an image.
- Each instruction will create a new image layer to the image.
- Instructions specify what to do when building the image.

LEVEL UP
www.level-up.one

Docker Build Context

- Docker build command takes the path to the build context as an argument.
- When build starts, docker client would pack all the files in the build context into a tarball then transfer the tarball file to the daemon.
- By default, docker would search for the Dockerfile in the build context path.

LEVEL UP
www.level-up.one

Dockerfile In Depth

LEVEL UP
www.level-up.one

Steps

1. Spin up a container from a base image.
2. Install Git package in the container.
3. Commit changes made in the container.

LEVEL UP
www.level-up.one

Chain RUN Instructions

- Each RUN command will execute the command on the top writable layer of the container, then commit the container as a new image.
- The new image is used for the next step in the Dockerfile. So each RUN instruction will create a new image layer.
- It is recommended to chain the RUN instructions in the Dockerfile to reduce the number of image layers it creates.

LEVEL UP
www.level-up.one

Sort Multi-line Arguments Alphanumerically

- This will help you avoid duplication of packages and make the list much easier to update.

LEVEL UP
www.level-up.one

CMD Instructions

- CMD instruction specifies what command you want to run when the container starts up.
- If we don't specify CMD instruction in the Dockerfile, Docker will use the default command defined in the base image.
- The CMD instruction doesn’t run when building the image, it only runs when the container starts up.
- You can specify the command in either exec form which is preferred or in shell form.

LEVEL UP
www.level-up.one

Docker Cache

- Each time Docker executes an instruction it builds a new image layer.
- The next time, if the instruction doesn't change, Docker will simply reuse the existing layer.

LEVEL UP
www.level-up.one

Docker Cache

- Each time Docker executes an instruction it builds a new image layer.
- The next time, if the instruction doesn't change, Docker will simply reuse the existing layer.

LEVEL UP
www.level-up.one

Dockerfile with Aggressive Caching

FROM ubuntu:14.04 reusing cache

RUN apt-get update reusing cache

RUN apt-get install -y git curl

LEVEL UP
www.level-up.one

Cache Busting

```
FROM ubuntu:14.04

RUN apt-get update && apt-get install -y \
  git \
  curl
```

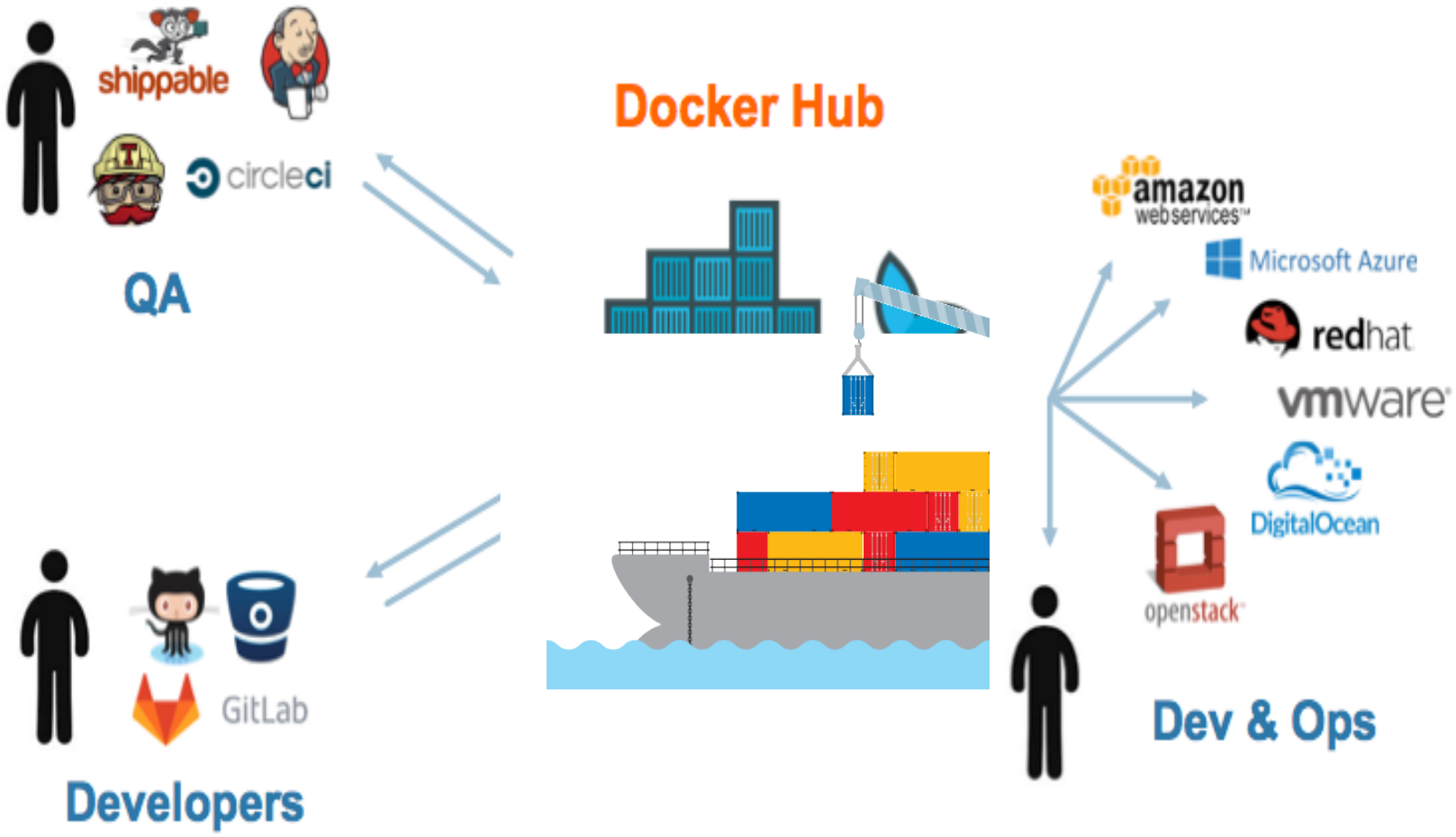
LEVEL UP
www.level-up.one

Cache Busting

- You can also achieve cache-busting by specifying a package version. This is known as version pinning.

```
RUN apt-get update && apt-get install -y \
  package-bar \
  package-baz \
  package-foo=1.3.*
```

LEVEL UP
www.level-up.one



LEVEL UP
www.level-up.one

Containerize a Hello World Web Application

LEVEL UP
www.level-up.one

Text Direction: Dockerize a Hello World Web Application

Check out source code:

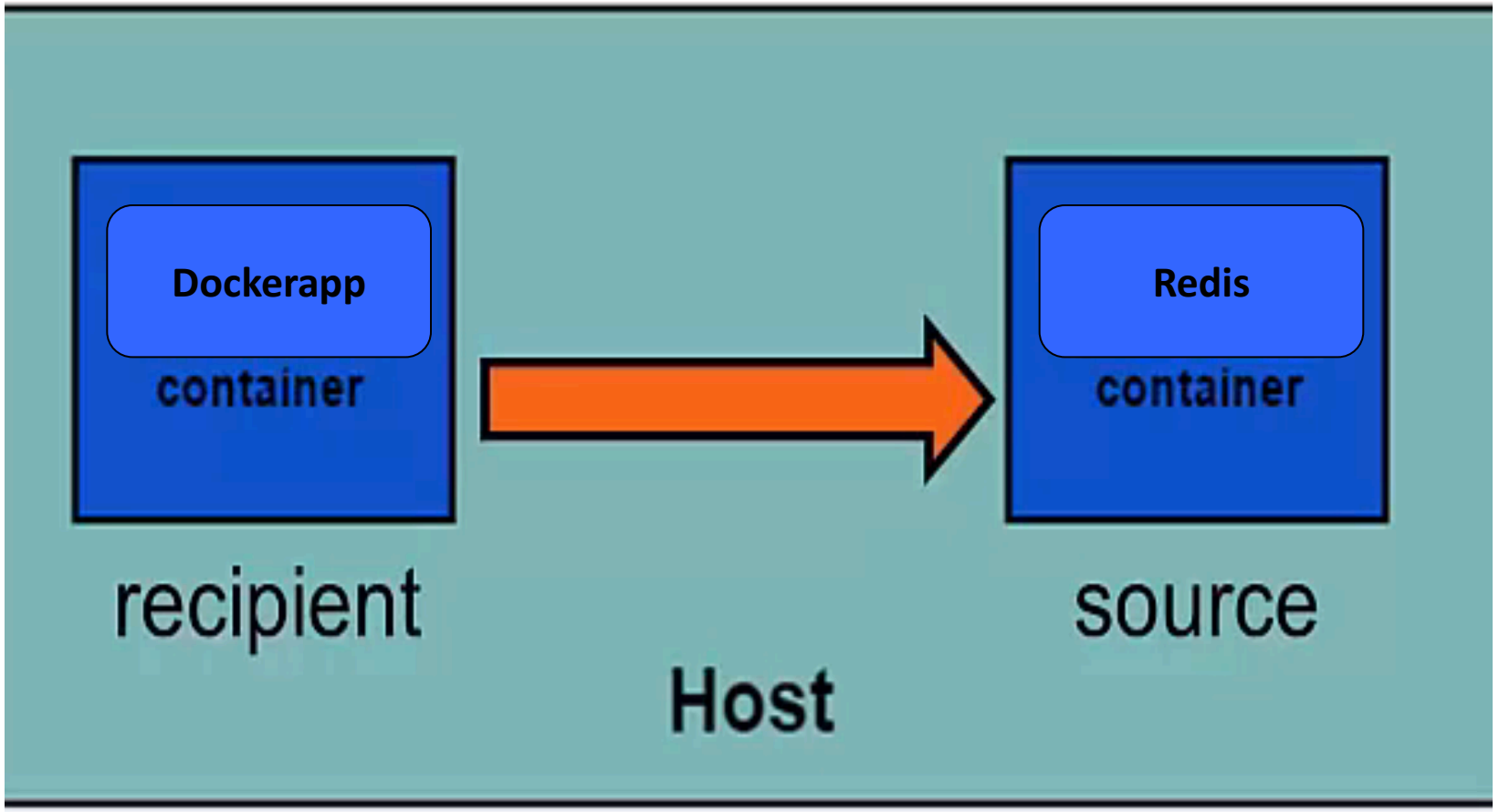
git clone -b v0.1 https://github.com/jleetutorial/dockerapp.git

LEVEL UP
www.level-up.one

Docker Container Links

LEVEL UP
www.level-up.one

Docker Container Links



LEVEL UP
www.level-up.one

How container links work behind the scenes?

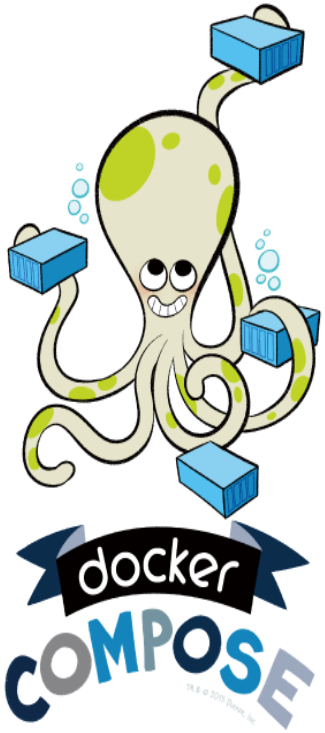
LEVEL UP
www.level-up.one

Benefits of Docker Container Links

- The main use for docker container links is when we build an application with a microservice architecture, we are able to run many independent components in different containers.

- Docker creates a secure tunnel between the containers that doesn't need to expose any ports externally on the container.

LEVEL UP
www.level-up.one



Automate the Docker Workflow with Docker Compose

LEVEL UP
www.level-up.one

Deep Dive into
Docker Compose Workflow

LEVEL UP
www.level-up.one

Why Docker Compose?

Manual linking containers and configuring services become impractical when the number of containers grows.

LEVEL UP
www.level-up.one

Docker Compose

- Docker compose is a very handy tool to quickly get docker environment up and running.

- Docker compose uses yaml files to store the configuration of all the containers, which removes the burden to maintain our scripts for docker orchestration.

LEVEL UP
www.level-up.one

Docker Compose Commands

- **docker compose up starts up all the containers.**
- **docker compose ps checks the status of the containers managed by docker compose.**
- **docker compose logs outputs colored and aggregated logs for the compose-managed containers.**
- **docker compose logs with dash f option outputs appended log when the log grows.**
- **docker compose logs with the container name in the end outputs the logs of a specific container.**
- **docker compose stop stops all the running containers without removing them.**
- **docker compose rm removes all the containers.**
- **docker compose build rebuilds all the images.**

LEVEL UP
www.level-up.one

Introduction to Docker Networking

LEVEL UP
www.level-up.one

Docker Network Types

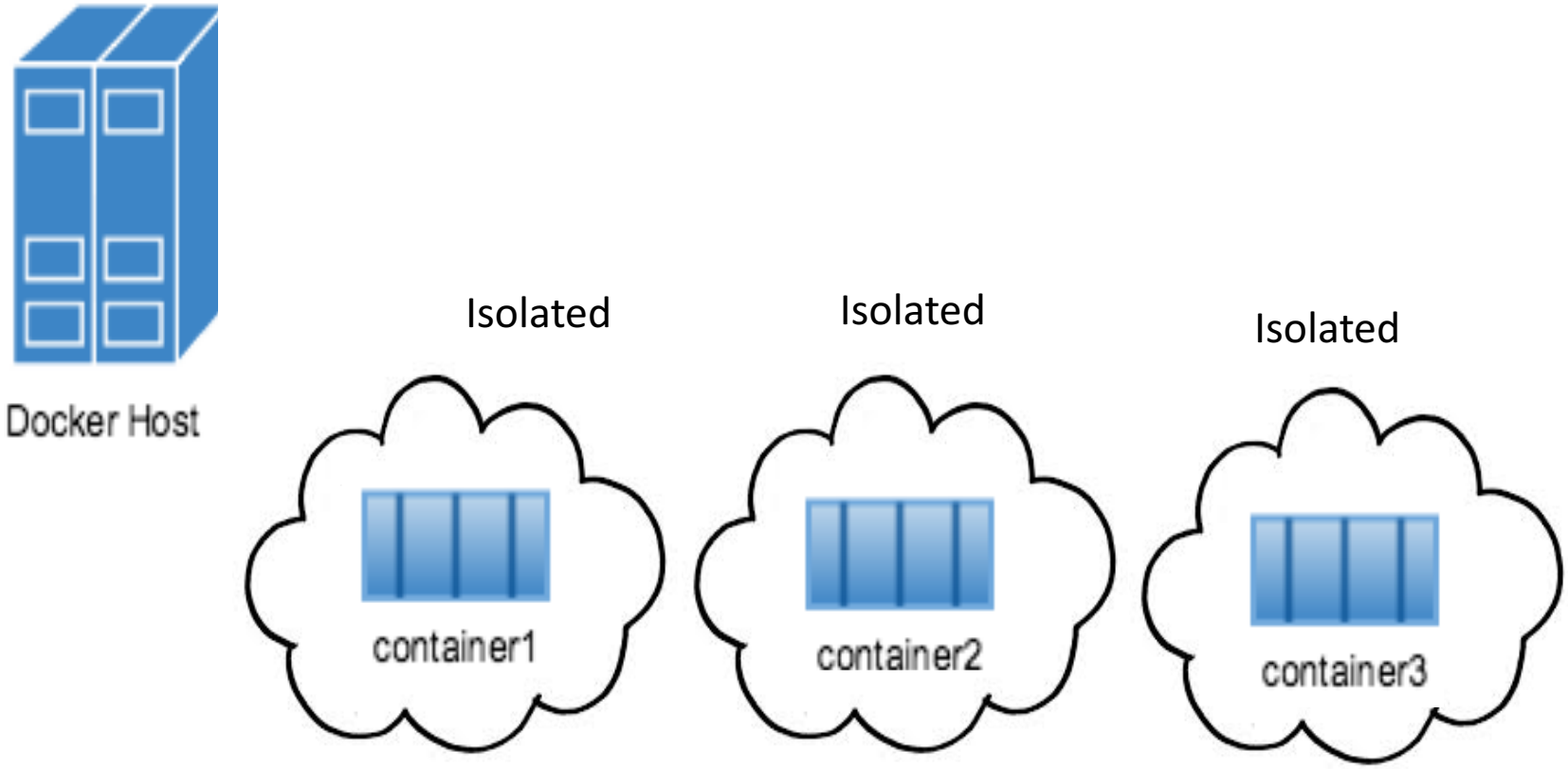
- Closed Network / None Network
- Bridge Network
- Host Network
- Overlay Network

LEVEL UP
www.level-up.one

None Network

LEVEL UP
www.level-up.one

None Network



LEVEL UP
www.level-up.one

None Network

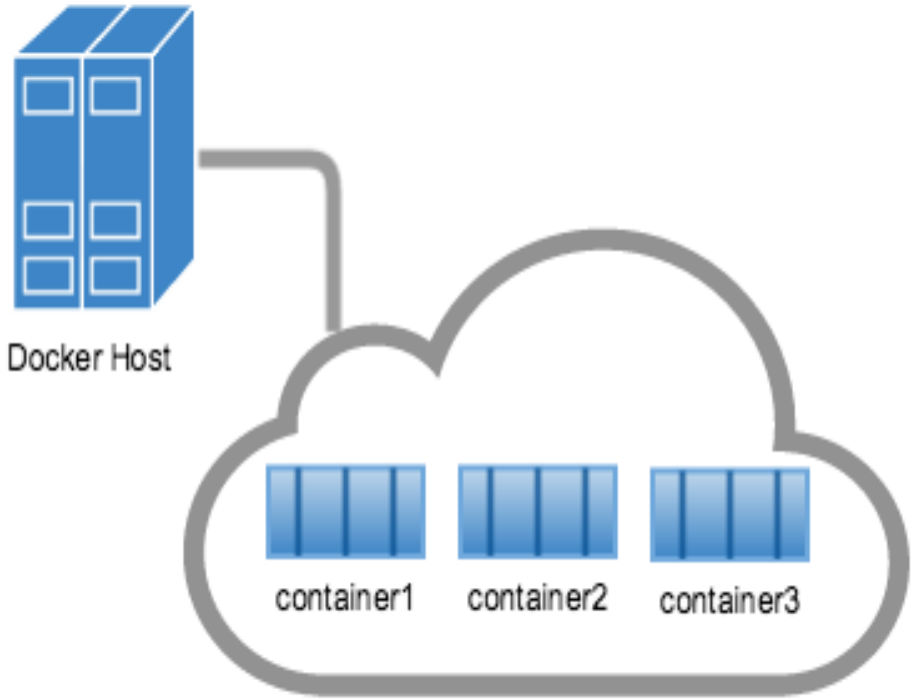
- Provides the maximum level of network protection.
- Not a good choice if network or Internet connection is required.
- Suites well where the container require the maximum level of network security and network access is not necessary.

LEVEL UP
www.level-up.one

Bridge Network

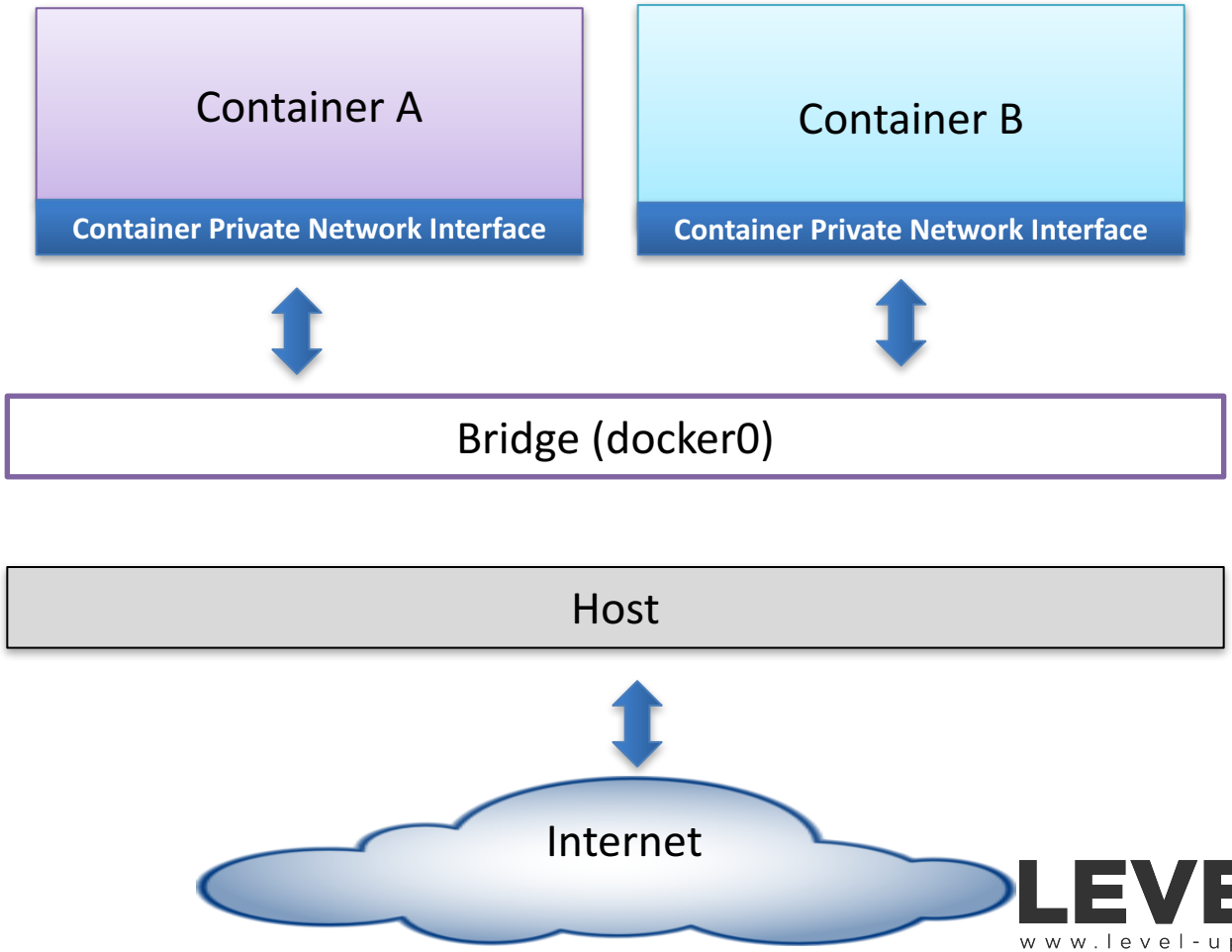
LEVEL UP
www.level-up.one

Bridge Network



LEVEL UP
www.level-up.one

Bridge Network



LEVEL UP
www.level-up.one

Host and Overlay Network

LEVEL UP
www.level-up.one

Host Network

- The least protected network model, it adds a container on the host's network stack.
- Containers deployed on the host stack have full access to the host's interface.
- This kind of containers are usually called open containers.

LEVEL UP
www.level-up.one

Host Network

- Minimum network security level.
- No isolation on this type of open containers, thus leave the container widely unprotected.
- Containers running in the host network stack should see a higher level of performance than those traversing the docker0 bridge and iptables port mappings.

LEVEL UP
www.level-up.one

Define Container Networks with Docker Compose

LEVEL UP
www.level-up.one

Write and Run Unit Tests in Docker Containers

Unit Tests in Containers

- Unit tests should test some basic functionality of our docker app code, with no reliance on external services.

- Unit tests should run as quickly as possible so that developers can iterate much faster without being blocked by waiting for the tests results.

- Docker containers can spin up in seconds and can create a clean and isolated environment which is great tool to run unit tests with.

LEVEL UP
www.level-up.one

Incorporating Unit Tests into Docker Images

Pros:

- A single image is used through development, testing and production, which greatly ensures the reliability of our tests.

Cons:

- It increases the size of the image.

LEVEL UP
www.level-up.one

Fit Docker Technology into Continuous Integration(CI) Process

LEVEL UP
www.level-up.one

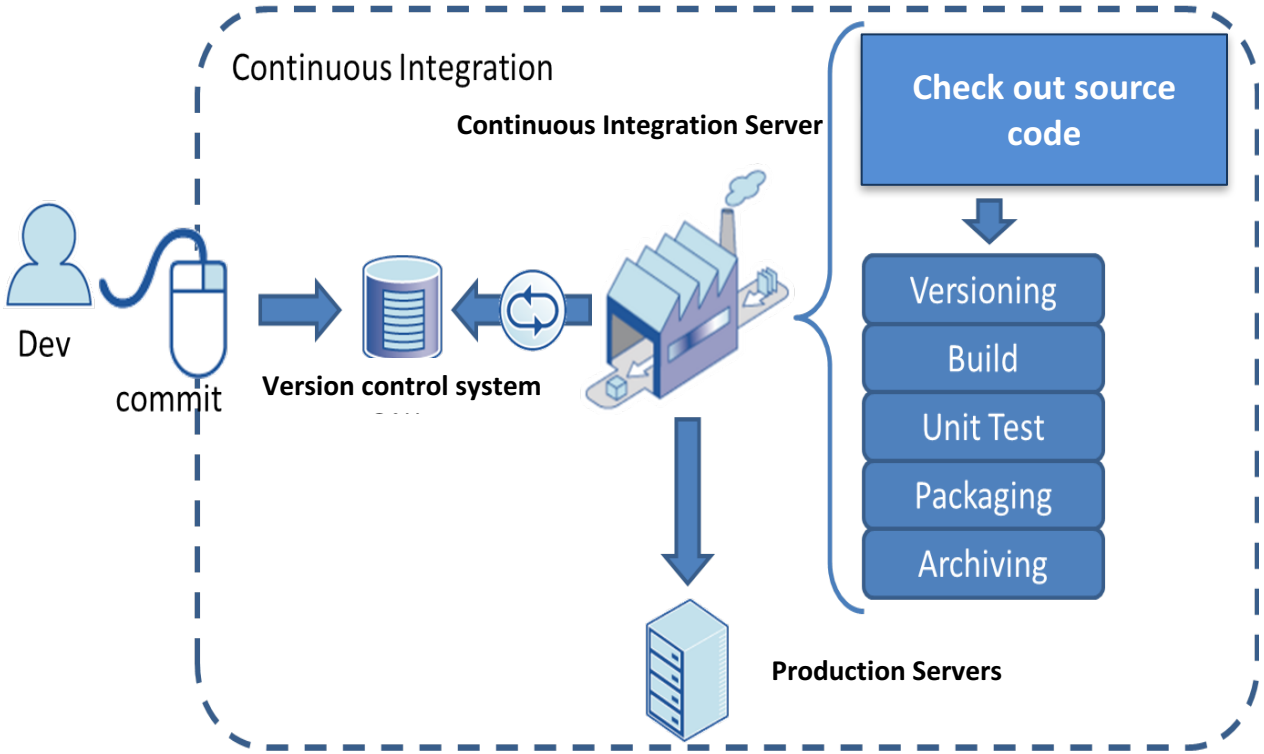
What is Continuous Integration?

- Continuous integration is a software engineering practice in which isolated changes are immediately tested and reported when they are added to a larger code base.

- The goal of Continuous integration is to provide rapid feedback so that if a defect is introduced into the code base, it can be identified and corrected as soon as possible.

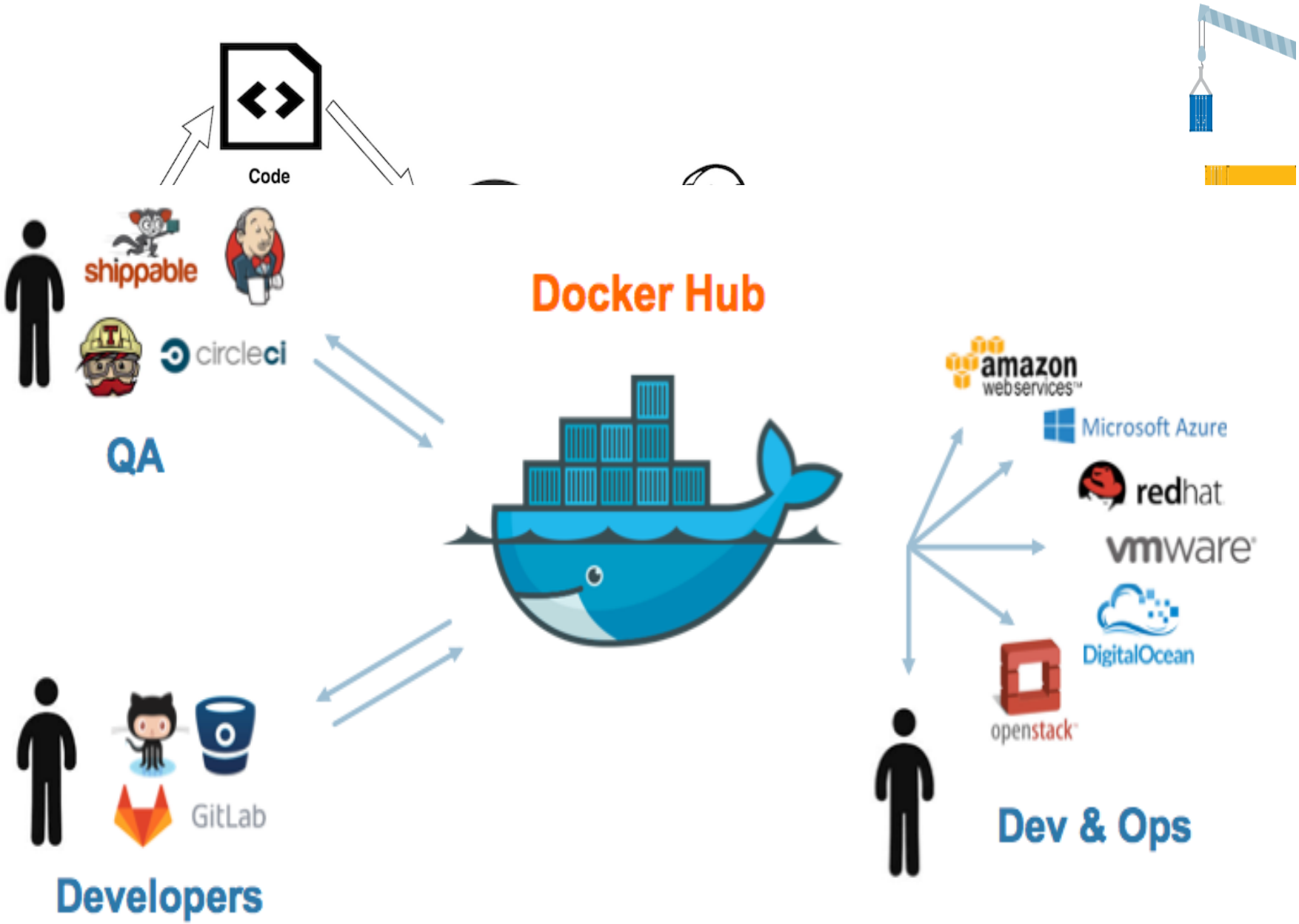
LEVEL UP
www.level-up.one

A Typical CI Pipeline without Docker

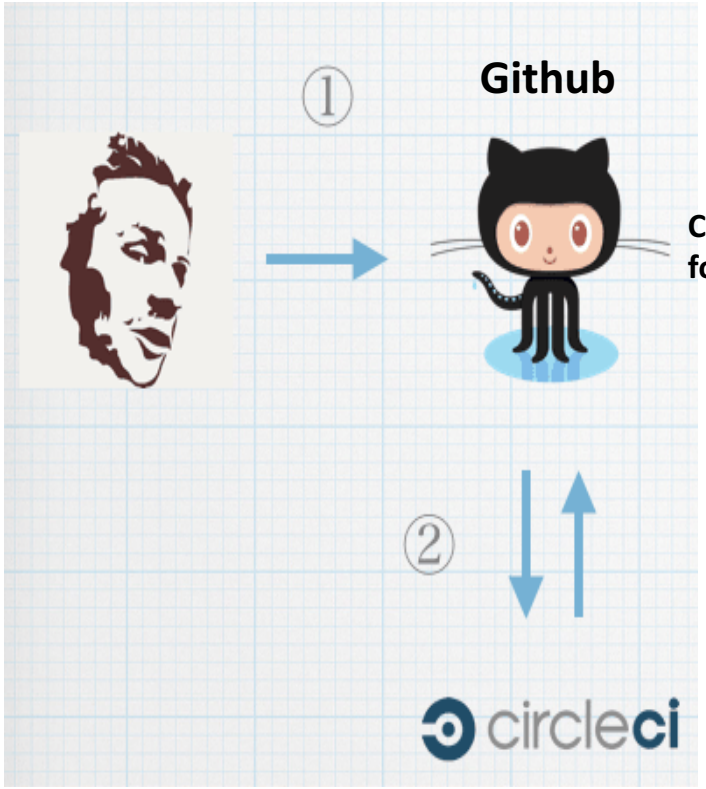


LEVEL UP
www.level-up.one

CI process with Docker technologies involved



Our Continuous Integration Pipeline



LEVEL UP
www.level-up.one

Text Direction: Introduction to Continuous Integration

URL of the Github account to fork:

https://github.com/jleetutorial/dockerapp

Checking for existing SSH keys:

https://help.github.com/articles/checking-for-existing-ssh-keys/

Generating a new SSH key and adding it to the ssh-agent:

https://help.github.com/articles/generating-a-new-ssh-key-and-adding-it-to-the-ssh-agent

Adding a new SSH key to your GitHub account:

https://help.github.com/articles/adding-a-new-ssh-key-to-your-github-account/

LEVEL UP
www.level-up.one

Link Circle CI with GitHub Account

to build a Continuous Integration pipeline

LEVEL UP
www.level-up.one

Set up SSH keys for Github Account

- SSH keys are a way to identify trusted computers without involving password.
- Generate a SSH key pair and save the private SSH key in your local box and add the public key to your GitHub account.
- Then you can directly push your changes to github repository without typing password.

LEVEL UP
www.level-up.one

How to check if SSH public key files are available on your local box?

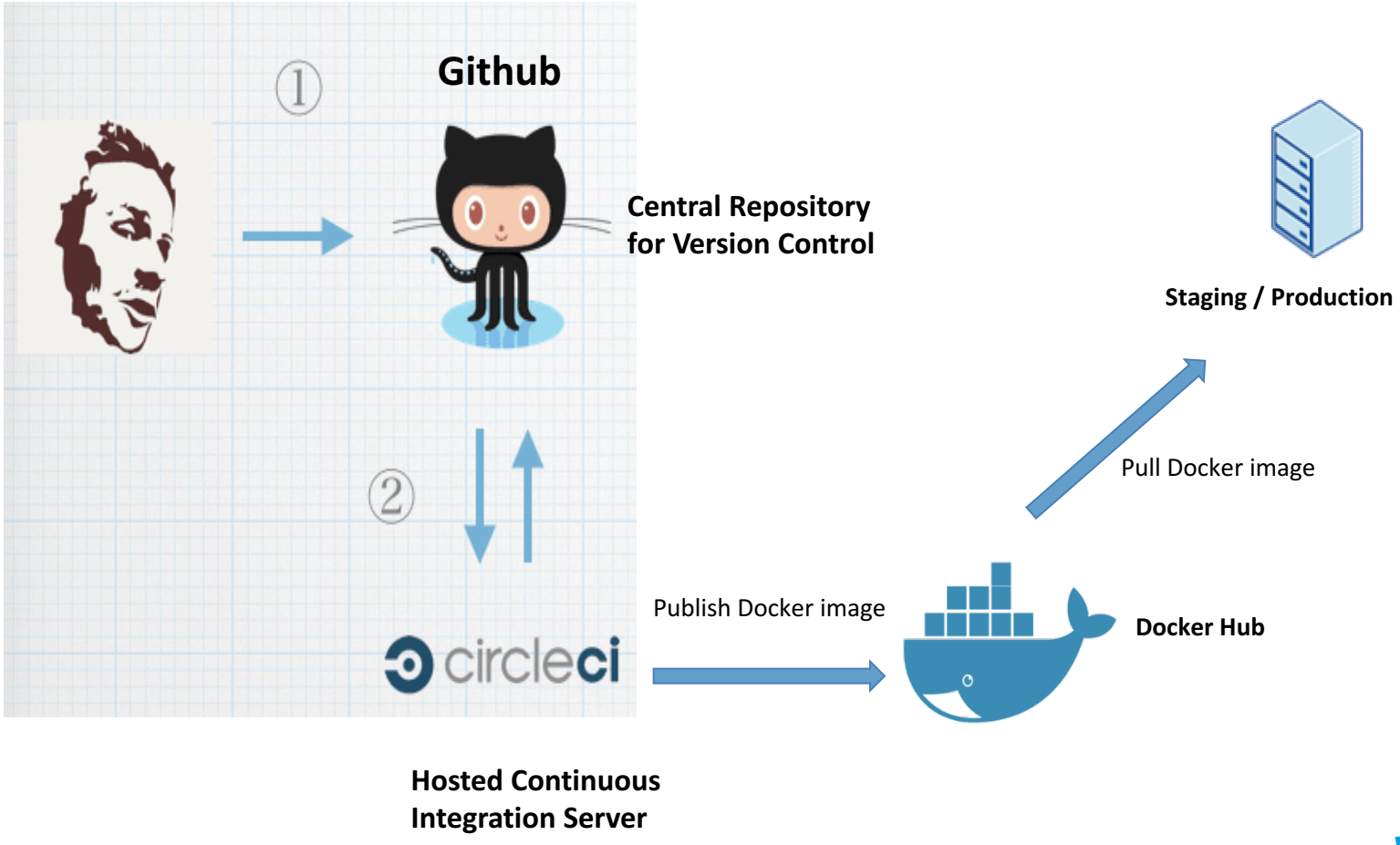
The SSH public key file usually sits under ***~/.ssh/*** directory and ends with ***.pub*** extension.

LEVEL UP
www.level-up.one

Link Circle CI with GitHub Account

LEVEL UP
www.level-up.one

Complete CI Workflow



LEVEL UP
www.level-up.one

Tag the Docker Images with Two Tags

1. commit hash of the source code
2. latest

LEVEL UP
www.level-up.one

Introduction to Running Docker in Production

Opinions about Running Docker in Production

- One one hand, many docker pioneers are confident that a distributed web app can be deployed at scale using Docker and have incorporated Docker into their production environment.

- On the other hand, there are still some people who are reluctant to use Docker in production as they think docker workflow is too complex or unstable for real life use cases.

LEVEL UP
www.level-up.one

Is Docker Production Ready Now?

LEVEL UP
www.level-up.one

Concerns about Running Docker in Production

- There are still some missing pieces about Docker around data persistence, networking, security and identity management.

- The ecosystem of supporting Dockerized applications in production such as tools for monitoring and logging are still not fully ready yet.

LEVEL UP
www.level-up.one

Companies which already run Docker in Production

LEVEL UP
www.level-up.one

Why Running Docker Containers inside VMs?

- To address security concerns.
- Hardware level isolation.

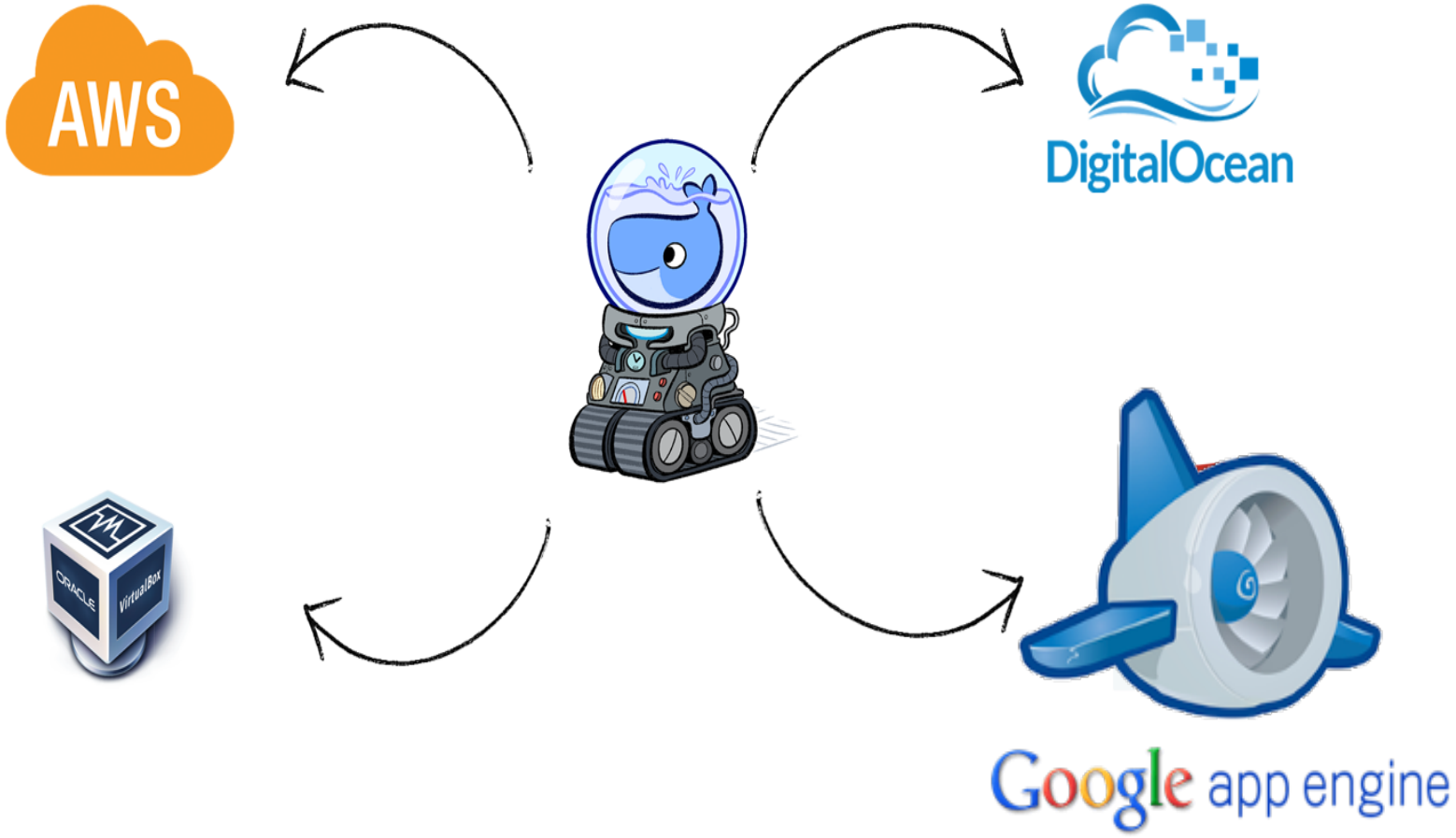
LEVEL UP
www.level-up.one

They all run containers inside VMs

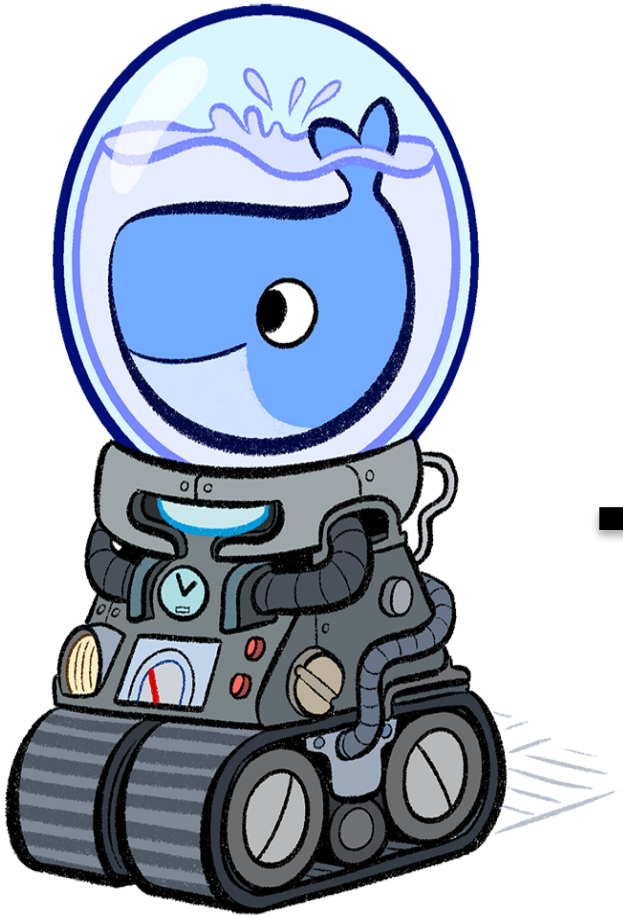
LEVEL UP
www.level-up.one

Docker Machine

LEVEL UP
www.level-up.one



LEVEL UP
www.level-up.one



VirtualBox

LEVEL UP
www.level-up.one

Register Digital Ocean Account to Deploy Containerized Applications

LEVEL UP
www.level-up.one

Deploy Docker App to the Cloud with Docker Machine

LEVEL UP
www.level-up.one

Text Direction: Deploy Docker App to the Cloud with Docker Machine

Docker Machine Create command

docker-machine create --driver digitalocean --digitalocean-access-token <xxxxx> docker-app-machine

LEVEL UP
www.level-up.one

Introduction to Docker Swarm
and
Set up Swarm cluster

LEVEL UP
www.level-up.one

How Swarm cluster works

- To deploy your application to a swarm, you submit your service to a manager node.
- The manager node dispatches units of work called tasks to worker nodes.
- Manager nodes also perform the orchestration and cluster management functions required to maintain the desired state of the swarm.
- Worker nodes receive and execute tasks dispatched from manager nodes.
- An agent runs on each worker node and reports on the tasks assigned to it. The worker node notifies the manager node of the current state of its assigned tasks so that the manager can maintain the desired state of each worker.

LEVEL UP
www.level-up.one

Deploy Docker App Services to the Cloud via Docker Swarm

Docker Swarm

LEVEL UP
www.level-up.one

Docker Services

- The services can be defined in our Docker compose file.
- The service definition includes which Docker images to run, the port mapping and dependency between services.

```
version: "3.0"
services:
  dockerapp:
    build: .
    ports:
      - "5000:5000"
    depends_on:
      - redis
  redis:
    image: redis:3.2.0
```

LEVEL UP
www.level-up.one

Docker Services

- When we are deploying services in the swarm mode, we can also set another important configuration, which is the deploy key and it is only available on Compose file formats version 3.x and up.

- The deploy key and its sub-options can be used to load balance and optimize performance for each service.

```
version: "3.0"
services:
  dockerapp:
    image: jleetutorial/dockerapp
    ports:
      - "5000:5000"
    depends_on:
      - redis
    deploy:
      replicas: 2
  redis:
    image: redis:3.2.0
```

LEVEL UP
www.level-up.one

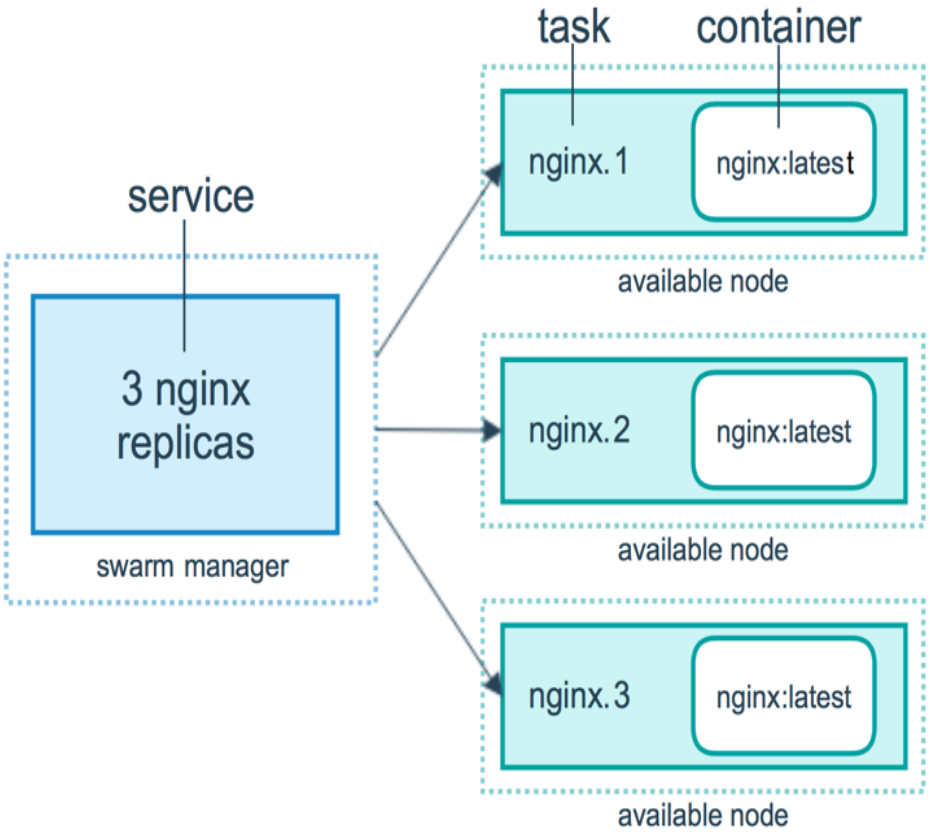
Deploy Key in Docker Compose file

docker-compose.yml

```yaml
version: "3.3"
services:
  nginx:
    image: nginx:latest
    ports:
        - "80:80"
    deploy:
      replicas: 3
      resources:
        limits:
          cpus: '0.1'
          memory: 50M
      restart_policy:
        condition: on-failure
```

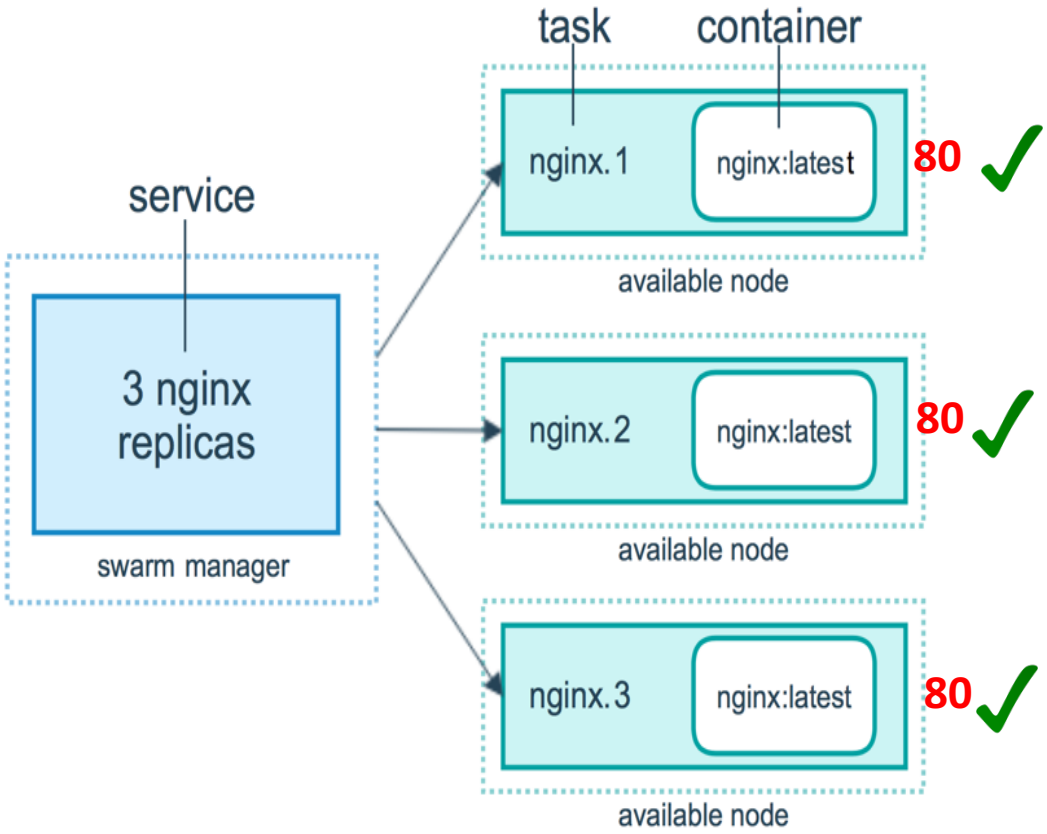
LEVEL UP
www.level-up.one

Replicas



LEVEL UP
www.level-up.one

Replicas

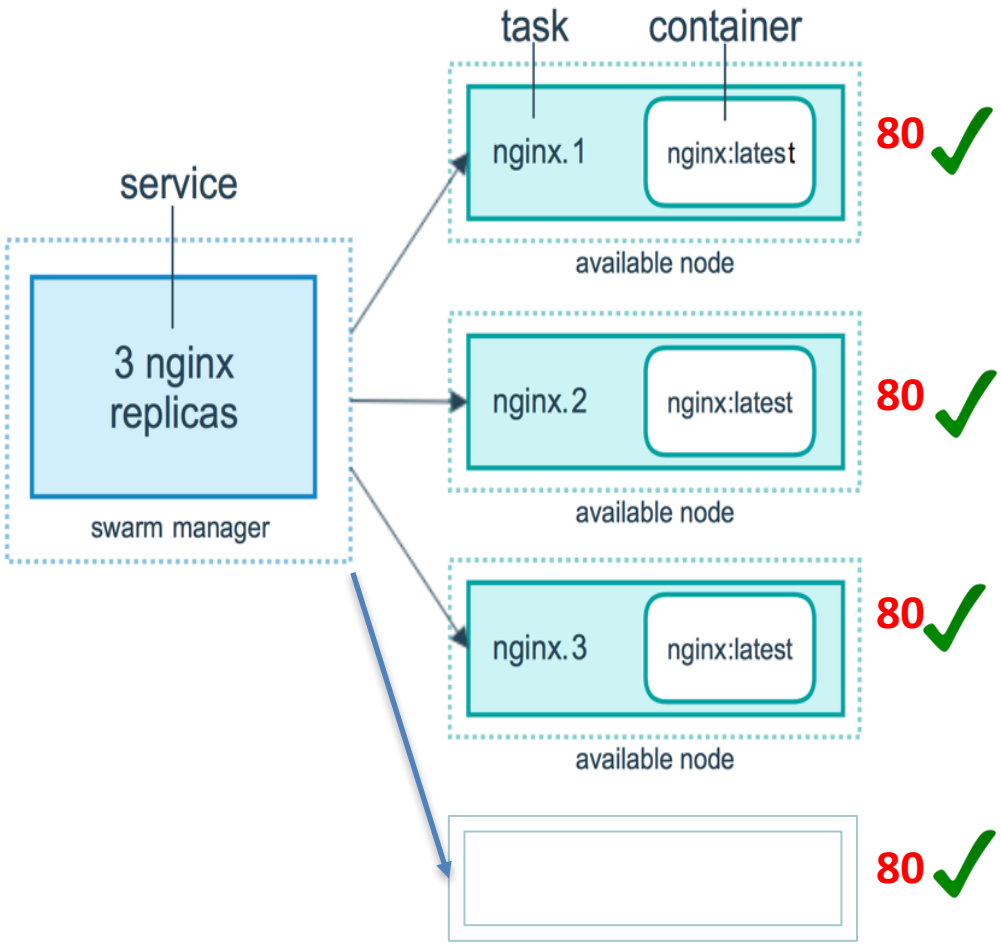


docker-compose.yml

```yaml
version: "3.3"
services:
  nginx:
    image: nginx:latest
    ports:
      - "80:80"
    deploy:
      replicas: 3
      resources:
        limits:
          cpus: '0.1'
          memory: 50M
      restart_policy:
        condition: on-failure
```

LEVEL UP
www.level-up.one

Replicas



- We can connect to the nginx service through a node which does NOT have nginx replicas.
- *Ingress load balancing*
  - All nodes listen for connections to published service ports.
  - When that service is called by external systems, the receiving node will accept the traffic and internally load balance it using an internal DNS service that Docker maintains.

LEVEL UP
www.level-up.one

Docker Stack

- A docker stack is a group of interrelated services that share dependencies, and can be orchestrated and scaled together.
- You can image that a stack is a live collection of all the services defined in your docker compose file.
- Create a stack from your docker compose file:
  - docker stack deploy
- In the Swarm mode,
  - Docker compose files can be used for service definitions.
  - Docker compose commands can't be reused. Docker compose commands can only schedule the containers to a single node.
  - We have to use docker stack command. You can think of docker stack as the docker compose in the swarm mode.

LEVEL UP
www.level-up.one

How to update our services in Production?

LEVEL UP
www.level-up.one

Provision a Swarm Cluster

- Step 1: Deploy two VMs, one will be used for the Swam manager node, and the other one will be used as a worker node.

- Step 2: Appoint the first VM as Swarm manager node and initialize a Swarm cluster.
  - docker swarm init

- Step 3: Let the second VM join the Swarm cluster as a worker node.
  - docker swarm join

LEVEL UP
www.level-up.one

Provision a Swarm Cluster

- Step 1: Deploy two VMs, one will be used for the Swam manager node, and the other one will be used as a worker node.

- Step 2: Appoint the first VM as Swarm manager node and initialize a Swarm cluster.
  - docker swarm init

- Step 3: Let the second VM join the Swarm cluster as a worker node.
  - docker swarm join

LEVEL UP
www.level-up.one

Provision a Swarm Cluster

- Step 1: Deploy two VMs, one will be used for the Swam manager node, and the other one will be used as a worker node.

- Step 2: Appoint the first VM as Swarm manager node and initialize a Swarm cluster.
  - docker swarm init

- Step 3: Let the second VM join the Swarm cluster as a worker node.
  - docker swarm join

LEVEL UP
www.level-up.one

Docker Swarm commands

- docker swarm init
  - Initialize a swarm. The docker engine targeted by this command becomes a manager in the newly created single-node swarm.
- docker swarm join
  - Join a swarm as a Swarm node.
- docker swarm leave
  - Leave the swarm.

LEVEL UP
www.level-up.one