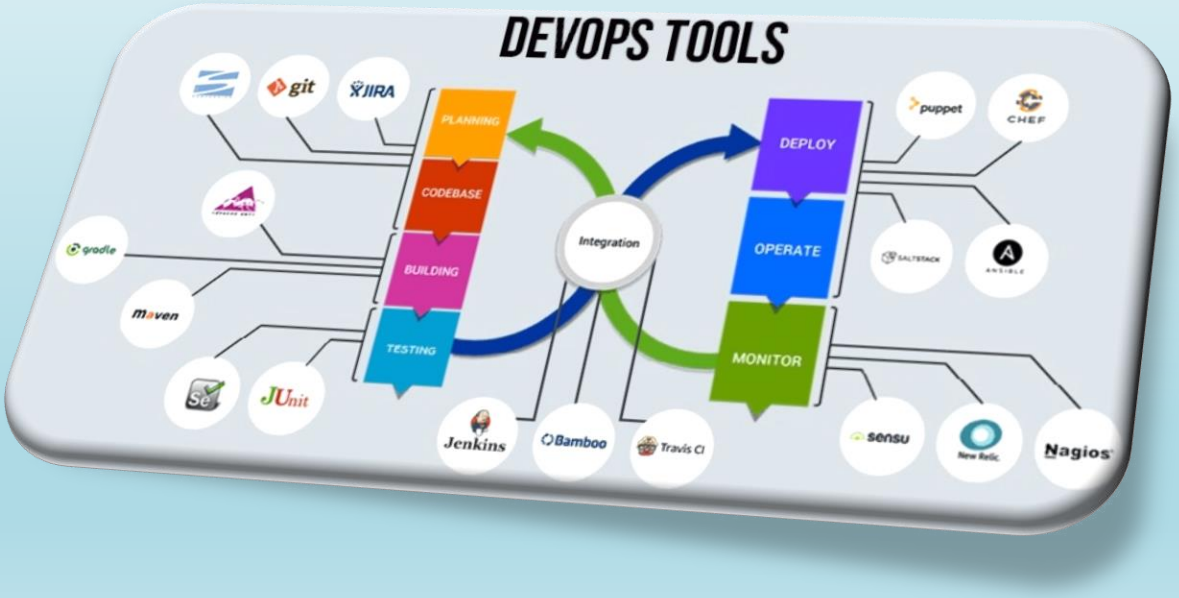
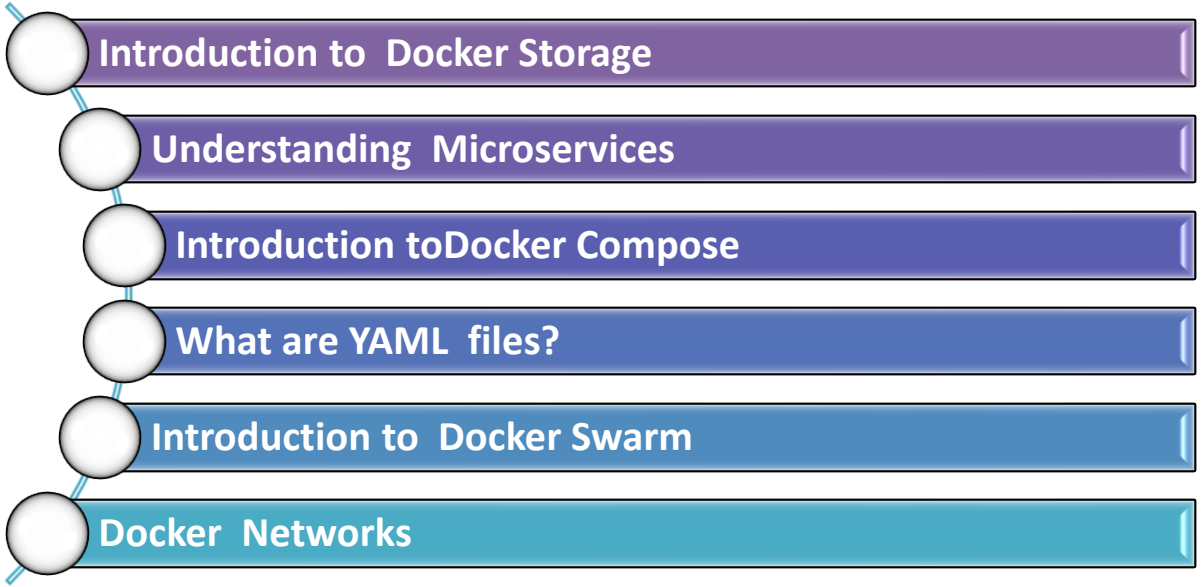
CLOUD TRAIN
ACCELERATE YOUR GROWTH

Containerization [Docker (II)]



CLOUD TRAIN
ACCELERATE YOUR GROWTH

Agenda



CLOUD TRAIN
ACCELERATE YOUR GROWTH

Introduction to Docker Storage

Introduction to Docker Storage

CLOUD TRAIN
ACCELERATE YOUR GROWTH

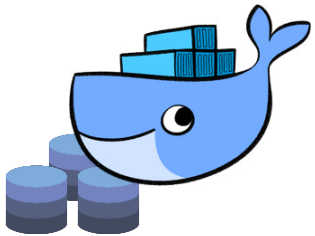
By default, all data of a container is stored on a writable container layer. This layer has the following properties:

- Data only exists while the container is active. If the container no longer exists, the data is also deleted along with the container.
- The writable container layer is tightly coupled with the host machine; hence, it is not portable.
- Data on the writable layer in the container is written using a storage driver.

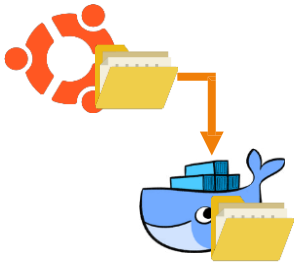
Introduction to Docker Storage

CLOUD TRAIN
ACCELERATE YOUR GROWTH

To persist data inside the container, even after it is deleted, we have two options:



Docker
Volumes



Bind
Mounts

Types of Docker Storage

- Docker Volumes
- Bind Mounts

A **Docker Volume** is a mountable entity which can be used to store data in the docker filesystem.

Syntax

```
docker volume create my-vol
```

```
ubuntu@ip-172-31-45-114:~$ docker volume create my-vol
my-vol
ubuntu@ip-172-31-45-114:~$
```

Types of Docker Storage

CLOUD TRAIN
ACCELERATE YOUR GROWTH

Docker Volumes

Bind Mounts

A **Docker Volume** is a mountable entity which can be used to store data in the docker filesystem.

Syntax

docker run -it --mount source=<source=folder>,destination=<destination-folder> -d <container-name>

```
ubuntu@ip-172-31-45-114:~$ docker run -it -d --mount source=my-vol,destination=/
app ubuntu
592f59807209a6881b7fd5fa7de0db2c6a6c97bc48ec0905af3832a0642a4ace
ubuntu@ip-172-31-45-114:~$
```

Types of Docker Storage

CLOUD TRAIN
ACCELERATE YOUR GROWTH

Docker Volumes

Bind Mounts

Bind Mounts mount a directory of the host machine to the docker container.

Syntax

docker run -it –v <source-directory>:<destination-directory> -d <container-name>

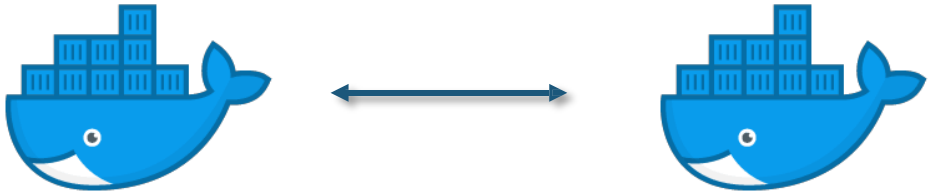
```
ubuntu@ip-172-31-17-56:~$ docker run -it -v /home/ubuntu/hello:/app -d ubuntu
979e6a564f141f38e9c18bb6d36c569185ea77b3f8d77aece2111c7af2396aa3
ubuntu@ip-172-31-17-56:~$
```

Linking Docker Containers

Linking Docker Containers

CLOUD TRAIN
ACCELERATE YOUR GROWTH

- Linking is a legacy feature of Docker, which is used to connect multiple containers.
- With linking, containers can communicate among each other.
- Name of containers is an important aspect while linking containers.
- Once you link containers, they can reach out to others using their names.



Linking Docker Containers

CLOUD TRAIN
ACCELERATE YOUR GROWTH

Syntax

docker run –it - -link <name-of-container> -d <image-name>

```
ubuntu@ip-172-31-45-114:~$ docker run -it --name container1 -d ubuntu
f98c3ecd6960198ac0ad9ce080ac6869bd5a9fbe65ed131b1470dab2b4fb7f7a
ubuntu@ip-172-31-45-114:~$ docker run -it --name container2 --link container1 -d
 ubuntu
9ab39c46d329d67eb0f024077a8cc8765c2d84e3ab6b2801bb688d4a3beb81ef
```

Hands-on: Linking Docker Containers

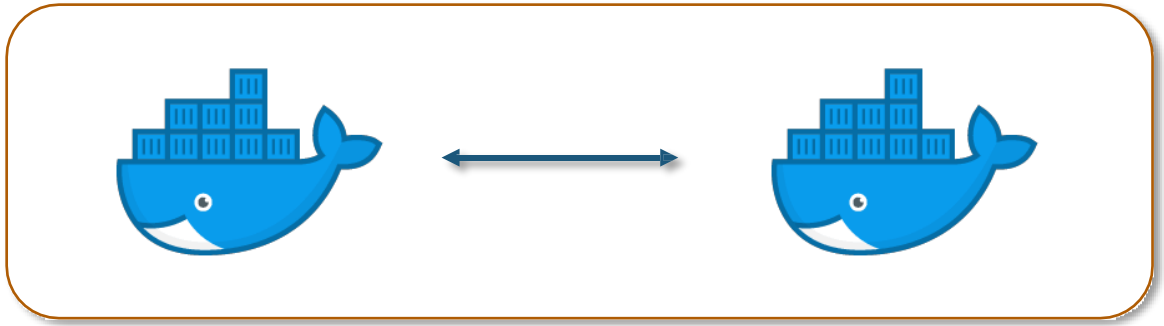
Hands-on: Linking Docker Containers

CLOUD TRAIN
ACCELERATE YOUR GROWTH

1. Create two containers of Ubuntu with names as follows: container1 and container2

2. Link container2 to container1

3. Try pinging from container2 to container1 by just using the command “ping container1”

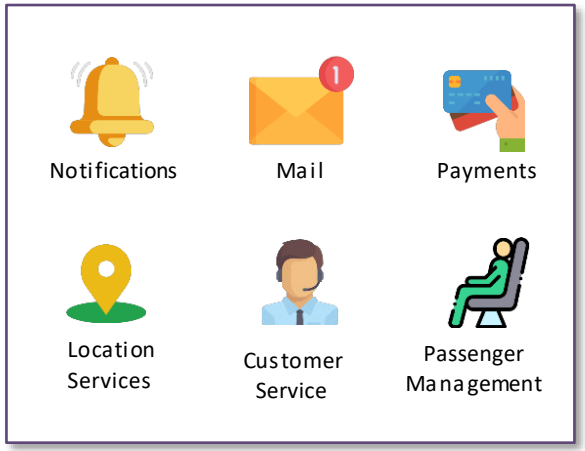


Understanding Microservices

What is a Monolithic Application?

CLOUD TRAIN
ACCELERATE YOUR GROWTH

A **Monolithic** application is a single-tiered software application in which different components are combined into a single program which resides in a single platform.



Disadvantages of a Monolithic Application

- Application is large and complex to understand.
- Entire application has to be re-deployed on an application update.
- Bug, in any module, can bring down the entire application.
- It has a barrier to adopting new technologies.

What are Microservices?

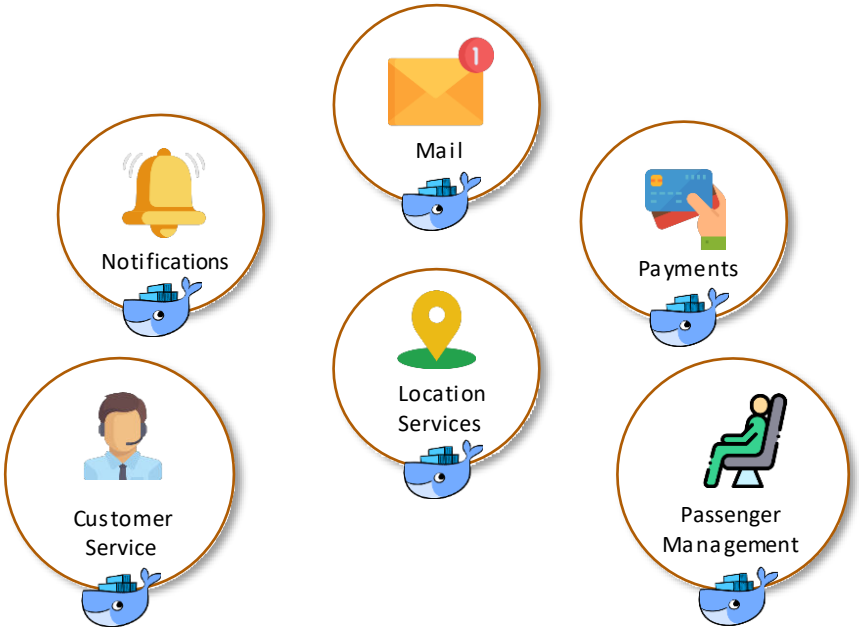
CLOUD TRAIN
ACCELERATE YOUR GROWTH

Microservices are a software development architectural style that structures an application as a collection of loosely coupled services.

What are Microservices?

CLOUD TRAIN
ACCELERATE YOUR GROWTH

Microservices are a software development architectural style that structures an application as a collection of loosely coupled services.



Advantages of Microservices

- Application is distributed, hence easy to understand.
- The code of only the Microservice which is supposed to be updated is changed.
- Bug, in one service, does not affect other services.
- There is no barrier to any specific technology.

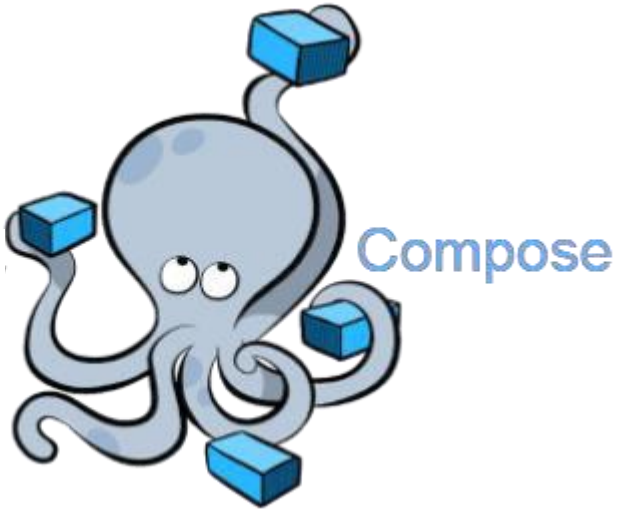
CLOUD TRAIN
ACCELERATE YOUR GROWTH

Introduction to Docker Compose

What is Docker Compose?

CLOUD TRAIN
ACCELERATE YOUR GROWTH

Compose is a tool for defining and running multi-container **Docker** applications. With **Compose**, you use a YAML file to configure your application's services. Then, with a single command, you create and start all the services from your configuration. Run **docker-compose** up and **compose** starts and runs your entire app.



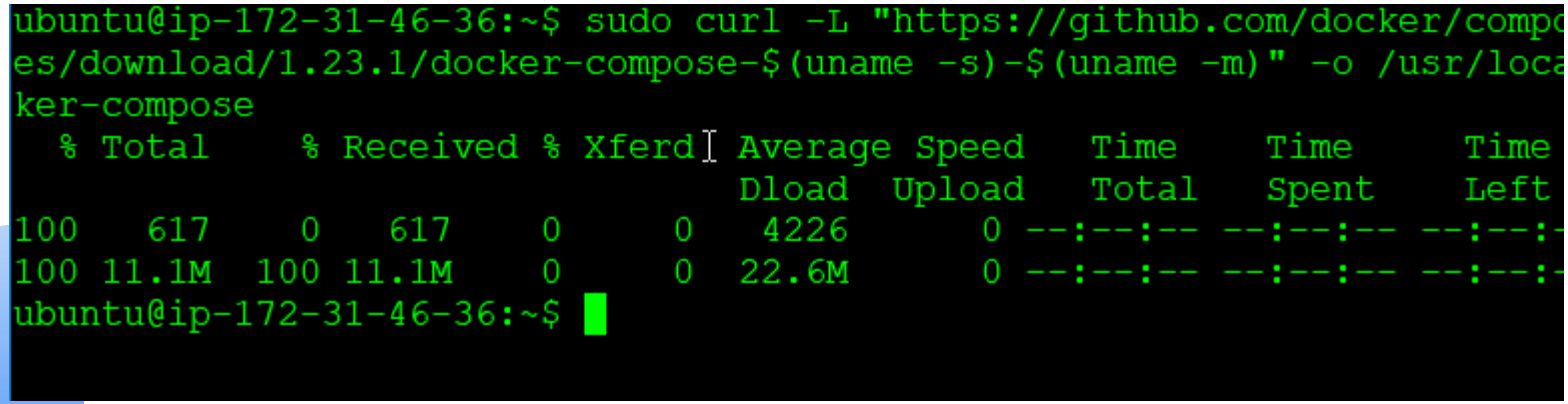
Installing Docker Compose

Installing Docker Compose

CLOUD TRAIN
ACCELERATE YOUR GROWTH

1. First, download the Docker Compose file using the following command:

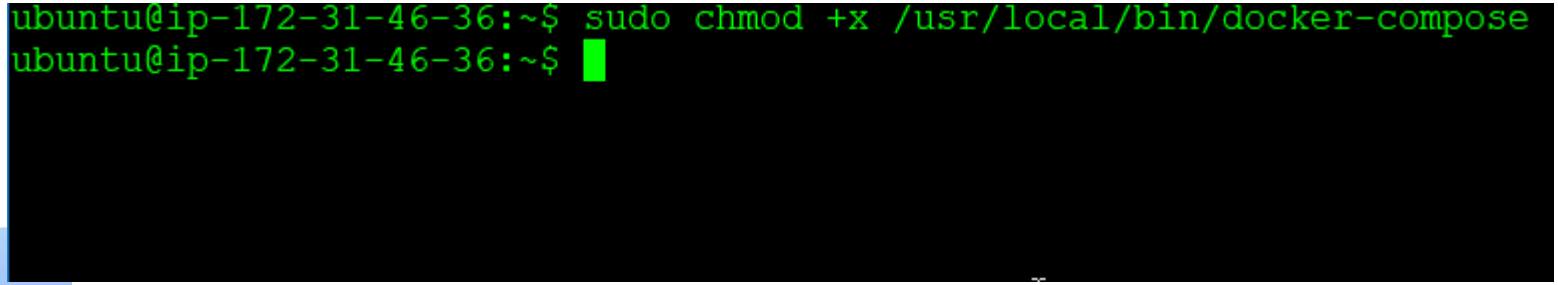
```
sudo curl -L "https://github.com/docker/compose/releases/download/1.24.1/docker-compose-$(uname -s)-$(uname -m)" -o /usr/local/bin/docker-compose
```



Installing Docker Compose

2. Now, give the required permission to the Docker Compose file to make it executable:

```
sudo chmod +x /usr/local/bin/docker-compose
```

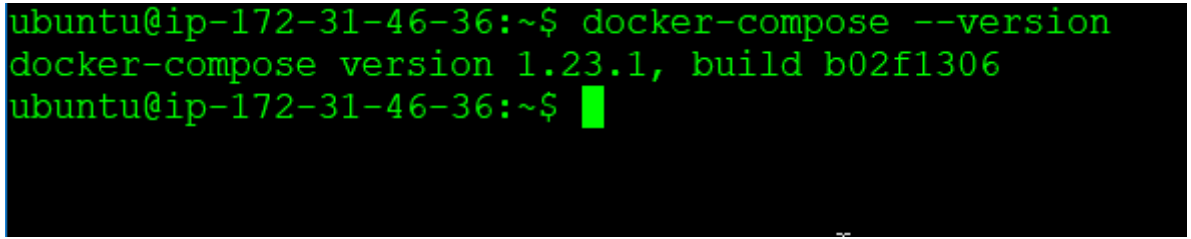


Installing Docker Compose

CLOUD TRAIN
ACCELERATE YOUR GROWTH

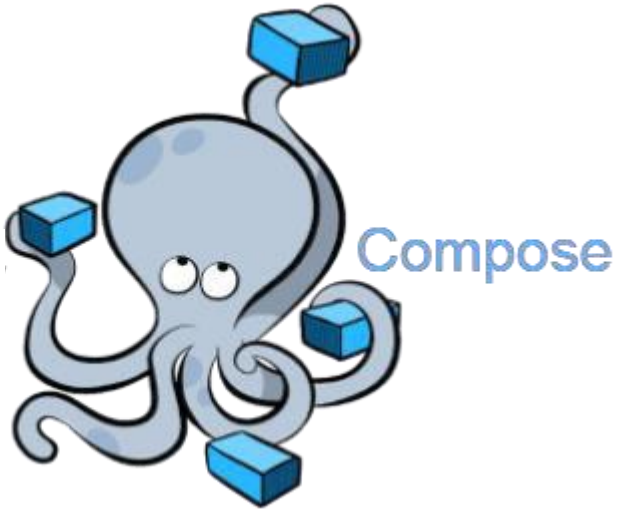
3. Finally, verify your installation using the following command:

```
docker-compose --version
```



What is Docker Compose?

For deploying containers using Docker Compose, we use YAML files.



What are YAML files?

What are YAML files?

CLOUD TRAIN
ACCELERATE YOUR GROWTH

YAML is a superset of a JSON file. There are only two types of structures in YAML which you need to know to get started:

- **Maps**
- **Lists**

What are YAML files?

CLOUD TRAIN
ACCELERATE YOUR GROWTH

Maps

Lists

When we map **a key** to **a value** in YAML files, they are termed as Maps.

```
<key> : <value>
```

For example:

```
Name: Training
Course: Devops
```

What are YAML files?

CLOUD TRAIN
ACCELERATE YOUR GROWTH

Maps

Lists

YAML Lists are a sequence of objects.

```
Args
- arg 1
- arg 2
- arg 3
```

For example:

```
args
  - sleep
  - "1000"
  - message
  - "Bring back Firefly!"
```

Writing a Docker Compose File

Writing a Docker Compose File

CLOUD TRAIN
ACCELERATE YOUR GROWTH

```
version: '3'
services:
  sample1:
    image: httpd
    ports:
     - “80:80"
  sample2:
    image: nginx
```

Sample Docker Compose File

Hands-on: Running a Sample Docker Compose File

Hands-on: Sample Docker Compose File

CLOUD TRAIN
ACCELERATE YOUR GROWTH

1. Create a folder called “docker”
2. Write the sample YAML file in “docker-compose.yml” file
3. To build this docker-compose file, the syntax is as follows:

```
docker-compose up -d
```

4. Ensure that all your containers are running

CLOUD TRAIN
ACCELERATE YOUR GROWTH

Hands-on: Deploying WordPress

Hands-on: Deploying WordPress

CLOUD TRAIN
ACCELERATE YOUR GROWTH

```
version: '3.3'
services:
  db:
    image: mysql:5.7
    volumes:
    - db_data:/var/lib/mysql
    restart: always
    environment:
      MYSQL_ROOT_PASSWORD: somewordpress
      MYSQL_DATABASE:  wordpress
      MYSQL_USER: wordpress
      MYSQL_PASSWORD:  wordpress
  wordpress:
    depends_on:
      - db
    image: wordpress:latest
    ports:
      - "8000:80"
    restart: always
    environment:
      WORDPRESS_DB_HOST:  db:3306
      WORDPRESS_DB_USER: wordpress
      WORDPRESS_DB_PASSWORD: wordpress
volumes:
  db_data:
```

docker-compose.yaml

Hands-on: Deploying WordPress

1. Create a folder called “docker-wordpress”
2. Write the sample YAML file in “docker-compose.yml” file
3. To build this docker-compose file, the syntax is as follows:

```
docker-compose up -d
```

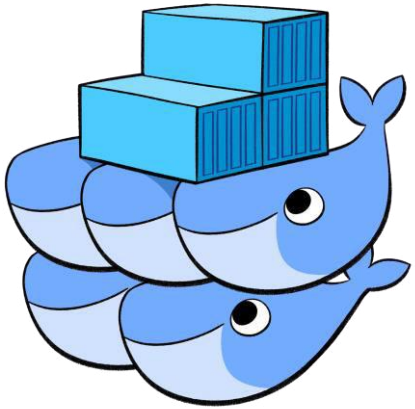
4. Ensure that all your containers are running

What is Container Orchestration?

What is Container Orchestration?

CLOUD TRAIN
ACCELERATE YOUR GROWTH

Applications are typically made up of individually containerized components (often called microservices) that must be organized at the networking level in order for the application to run as **intended**. The process of organizing multiple **containers** in this manner is known as **container orchestration**.



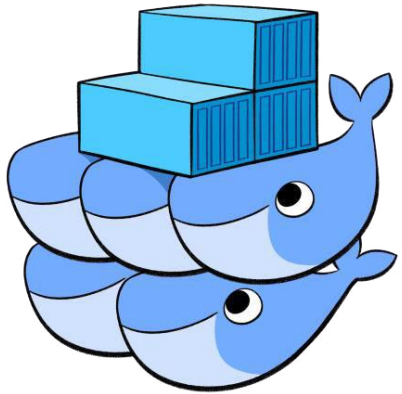
CLOUD TRAIN
ACCELERATE YOUR GROWTH

Introduction to Docker Swarm

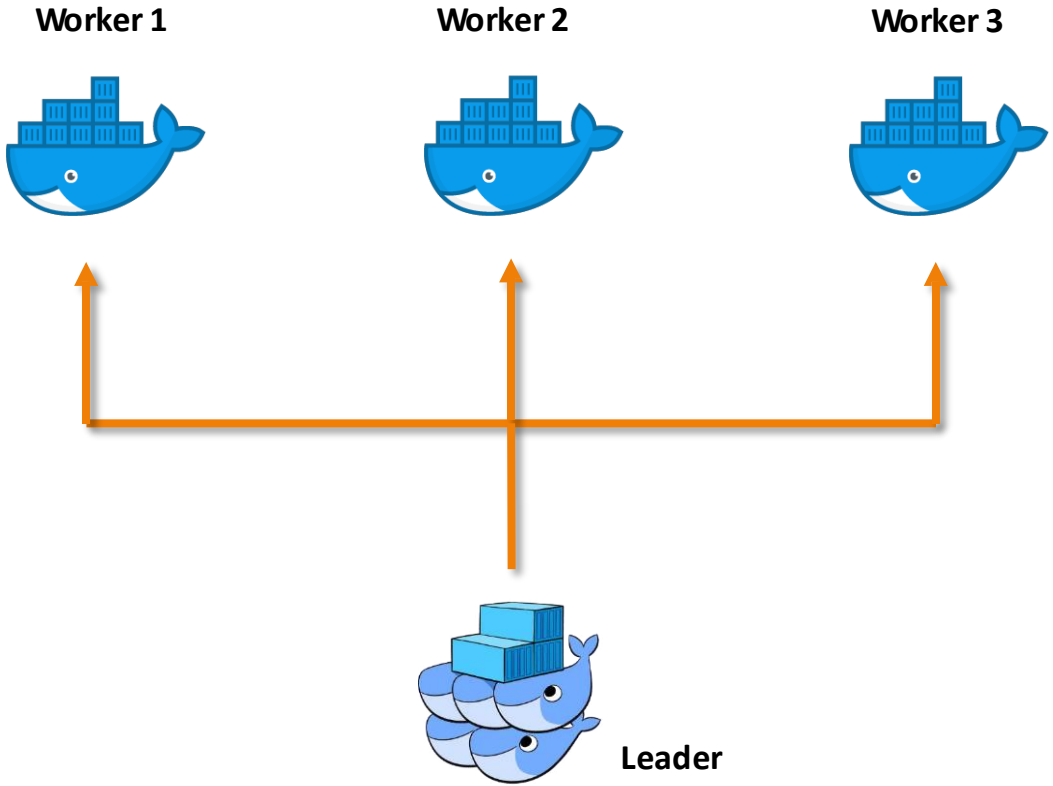
What is Docker Swarm?

CLOUD TRAIN
ACCELERATE YOUR GROWTH

Docker Swarmis a clustering and scheduling tool for **Docker** containers. With **Swarm**, IT administrators and developers can establish and manage a cluster of **Docker** nodes as a single virtual system.



What is Docker Swarm?



Creating a Docker Swarm Cluster

Creating a Docker Swarm Cluster

CLOUD TRAIN
ACCELERATE YOUR GROWTH

```
docker swarm init --advertise-addr=<ip-address-of-leader>
```

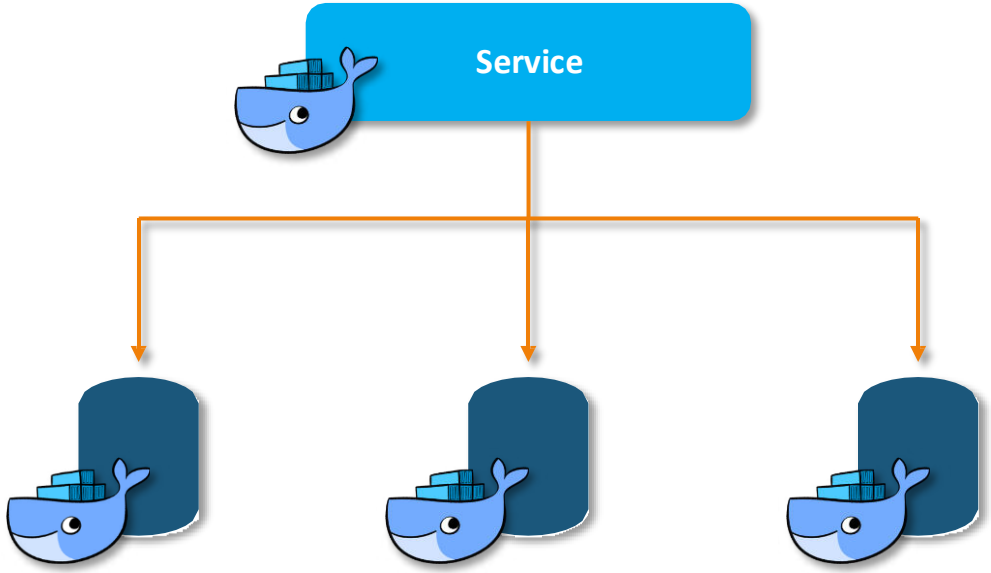
```
ubuntu@ip-172-31-26-120:~/wordpress$ docker swarm init --advertise-addr=172.31.2
Swarm initialized: current node (ptde8fg2vbxp8py931vrxdbpp) is now a manager.

To add a worker to this swarm, run the following command:

    docker swarm join --token SWMTKN-1-2m8bntbbysh354anwigivubiqwf21kq6xkww4kjnq
.26.120:2377

To add a manager to this swarm, run 'docker swarm join-token manager' and follow

ubuntu@ip-172-31-26-120:~/wordpress$
```

This command should be passed on to the worker node to join the docker swarm cluster.

Introduction to Services

What is a Service?

Containers on the cluster are deployed using **services** on Docker Swarm. A **service** is a long-running **Docker** container that can be deployed to any node worker.



Creating a Service

CLOUD TRAIN
ACCELERATE YOUR GROWTH

```
docker service create --name <name-of-service> --replicas <number-of-replicas> <image-name>
```

```
ubuntu@ip-172-31-26-120:~$ docker service create --name apache --replicas 3 -p 80:80 hshar/webapp
osftoz95rma0dkbsganqk0f3o
overall progress: 3 out of 3 tasks
1/3: running   [==================================================>]
2/3: running   [==================================================>]
3/3: running   [==================================================>]
verify: Service converged
ubuntu@ip-172-31-26-120:~$
```

CLOUD TRAIN
ACCELERATE YOUR GROWTH

Hands-on: Creating a Service in Docker Swarm

Hands-on: Creating a Service in Docker Swarm

CLOUD TRAIN
ACCELERATE YOUR GROWTH

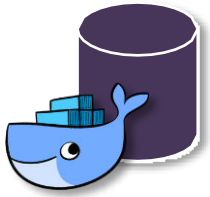
1. Create a Service for nginx webserver
2. There should be 3 replicas of this service running on the swarm cluster
3. Try accessing the service from Master IP and Slave IP

Docker Networks

Why Docker Networks?

Let's take an example. Say, there are two containers which we deploy in the docker ecosystem.

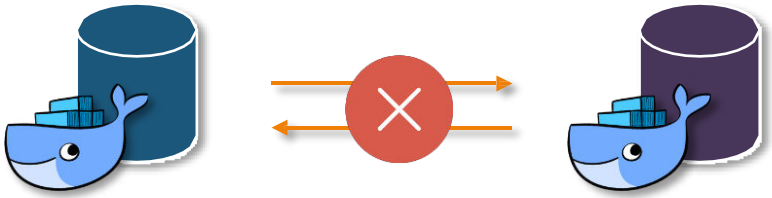
Website Container



Database Container

Why Docker Networks?

By default, these containers cannot communicate with each other.



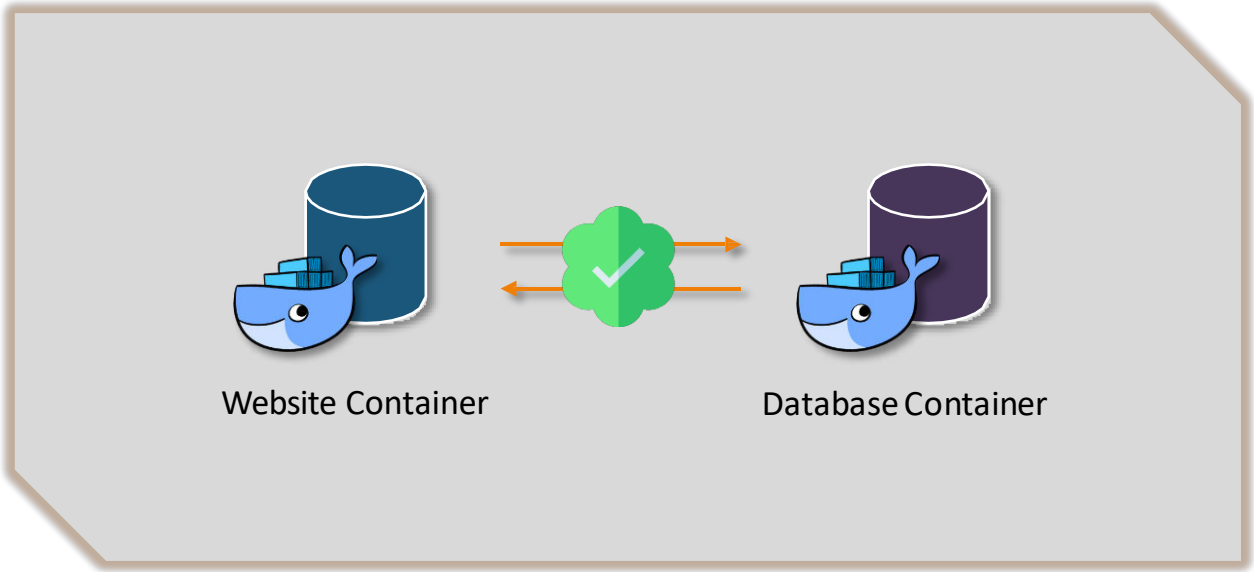
Website Container

Database Container

Why Docker Networks?

CLOUD TRAIN
ACCELERATE YOUR GROWTH

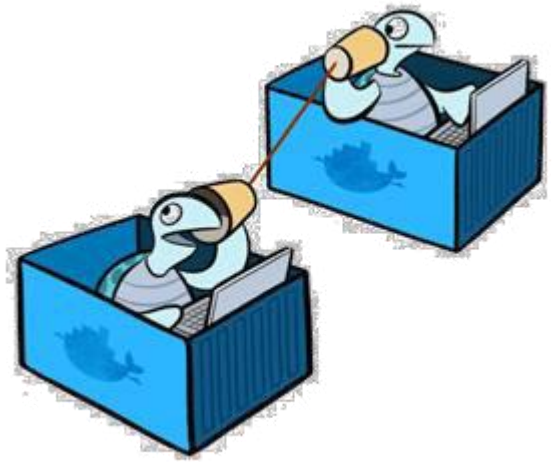
Therefore, in-order to have interactions between Docker Containers, we need Docker Networks.



What are Docker Networks?

CLOUD TRAIN
ACCELERATE YOUR GROWTH

One of the reasons Docker containers and services are so powerful is that you can connect them together or connect them to non-Docker workloads. And, this can be accomplished using Docker Networks.



Docker Network Types

Docker Networks are of the following types:

- bridge
- host
- overlay
- macvlan
- none

Docker Network Types

CLOUD TRAIN
ACCELERATE YOUR GROWTH

- **bridge**
- host
- overlay
- macvlan
- none

Bridge Networks

The default network driver. If you don't specify a driver, this is the type of network you are creating. Bridge networks are usually used when your applications run in standalone containers that need to communicate.

Docker Network Types

CLOUD TRAIN
ACCELERATE YOUR GROWTH

- bridge
- **host**
- overlay
- macvlan
- none

Host Networks

For standalone containers, remove network isolation between the container and the Docker host and use the host's networking directly. Host is only available for swarm services on Docker 17.06 and higher.

Docker Network Types

CLOUD TRAIN
ACCELERATE YOUR GROWTH

- bridge
- host
- **overlay**
- macvlan
- none

Overlay Networks

Overlay networks connect multiple Docker daemons together and enable swarm services to communicate with each other. You can also use overlay networks to facilitate communication between a swarm service and a standalone container or between two standalone containers on different Docker daemons.

Docker Network Types

CLOUD TRAIN
ACCELERATE YOUR GROWTH

- bridge
- host
- overlay
- **macvlan**
- none

Macvlan Networks

Macvlan networks allow you to assign a MAC address to a container, making it appear as a physical device on your network. The Docker daemon routes traffic to containers by their MAC addresses.

Docker Network Types

CLOUD TRAIN
ACCELERATE YOUR GROWTH

- bridge
- host
- overlay
- macvlan
- **none**

None

For this container, disable all networking. This is usually used in conjunction with a custom network driver. And, none is not available for swarm services.

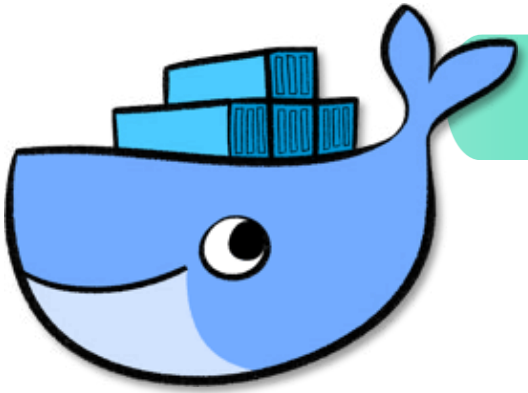
CLOUD TRAIN
ACCELERATE YOUR GROWTH

Hands-on: Deploying a Multi-tier App in Docker Swarm

Hands-on: Multi-tier App in Docker Swarm

CLOUD TRAIN
ACCELERATE YOUR GROWTH

1. Create an overlay network named “my-overlay”
2. Deploy a website container in the overlay network
   - Image: hshar/webapp
3. Deploy a database container in the overlay network
   - image: **hshar/mysql:5.6**
   - username: **root** password: **<password>**
4. Make changes in the website code to point to MySQL service
5. Test the configuration by entering values in the website



CLOUD TRAIN
ACCELERATE YOUR GROWTH

Got queries or need more info?

Contact us

TO ACCELERATE YOUR CAREER GROWTH

For questions and more details:

please call @ +91 98712 72900 or

visit https://www.thecloudtrain.com/ or

email at join@thecloudtrain.com or

WhatsApp us >>