Docker Recap

Students, write your response!

Pear Deck Interactive Slide
Do not remove this bar

CLARUSWAY©
WAY TO REINVENT YOURSELF

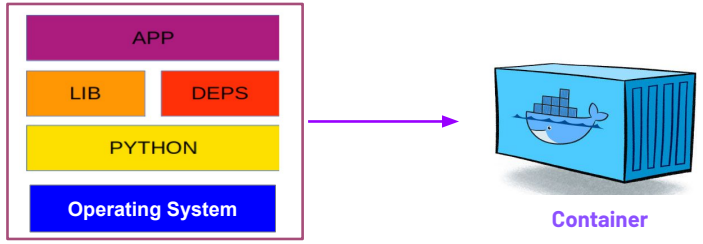
Table of Contents

2

What is Container?

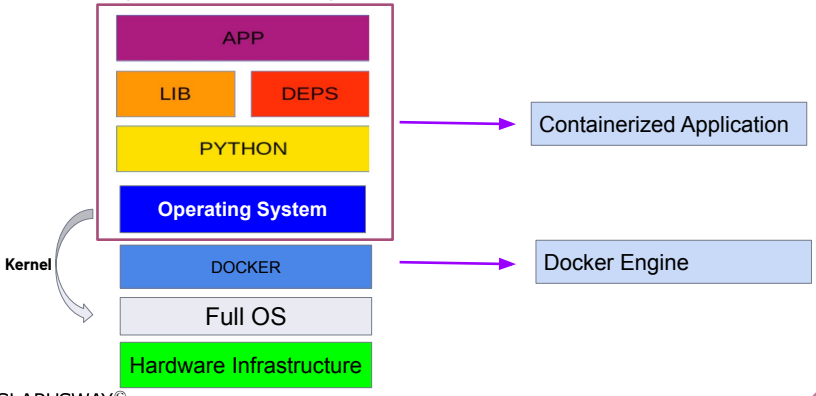
CLARUSWAY©
WAY TO REINVENT YOURSELF

What is Container?



CLARUSWAY©
WAY TO REINVENT YOURSELF

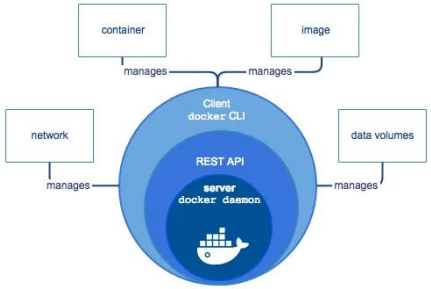
What is Container?



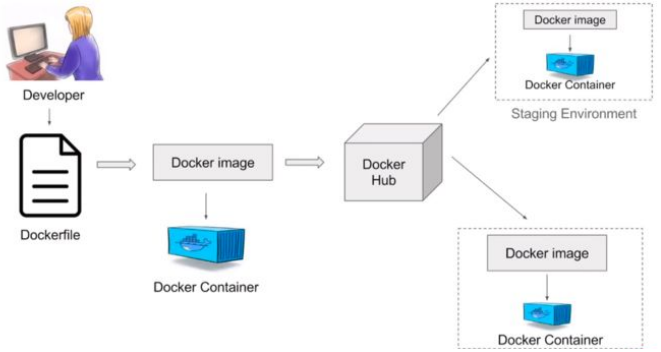
What is Container?

Docker Architecture

Docker uses a client-server architecture. The Docker client talks to the Docker daemon, which does the heavy lifting of building, running, and distributing your Docker containers. The Docker client and daemon can run on the same system, or you can connect a Docker client to a remote Docker daemon. The Docker client and daemon communicate using a REST API, over UNIX sockets or a network interface.

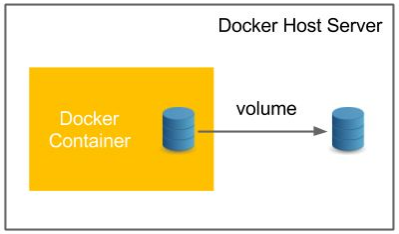


Images and Containers



Manage data in Docker

- By default, all files created inside a container are stored on a **writable container layer**. This means that the data doesn't persist when that container no longer exists.
- Docker volumes, which are special directories in a container, store files in the host machine so that the files are **persisted** even after the container stops.



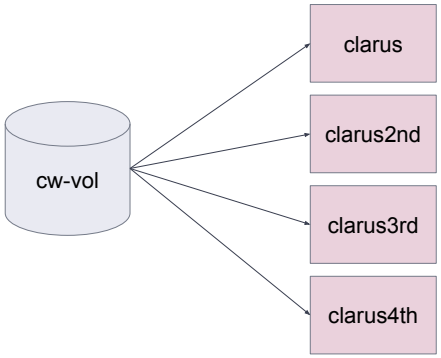
Manage data in Docker

- When we create a volume, it is stored within a directory on the Docker host. When we mount the volume into a container, this directory is what is mounted into the container.

```
$ docker volume inspect firstvolume
[
    {
        "CreatedAt": "2020-07-12T13:19:27Z",
        "Driver": "local",
        "Labels": {},
        "Mountpoint": "/var/lib/docker/volumes/firstvolume/_data",
        "Name": "firstvolume",
        "Options": {},
        "Scope": "local«
    }
]
```

Declaration of volumes

We can use the same Volume with different Containers.



Docker Volume Behaviours

No	Situation	Behaviour
1	If there is no target directory.	The target directory is created and files inside volume are copied to this directory.
2	If there is a target directory, but it is empty.	The files in the volume are copied to the target directory.
3	If there is a target directory and it is not empty, but volume is empty.	The files in the target directory are copied to volumes.
4	If the volume is not empty.	There will be just the files inside volume regardless of the target directory is full or empty.

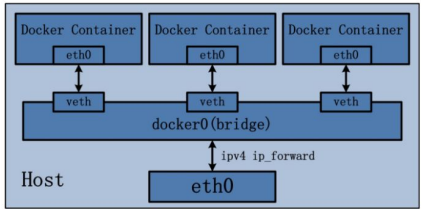
Network drivers

As default, docker has three network drivers.

- Bridge
- Host
- none

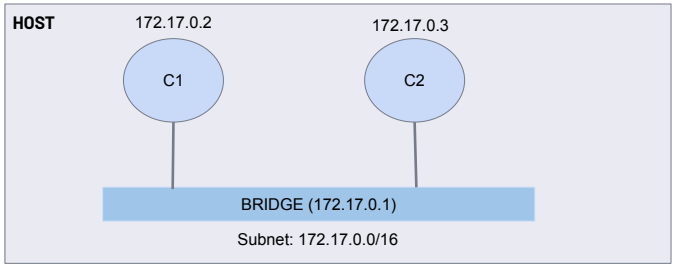
Network drivers

- **bridge** is the private default network driver. If we don't specify a driver, this is the type of network we are creating.
- When we install the docker, the Docker daemon creates virtual ethernet bridge **dockero** that performs the operation by automatically delivering packets among various network interfaces.



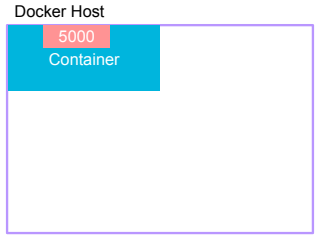
Network drivers

- When we create containers, it will automatically attach to the bridge driver.



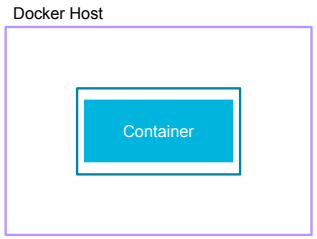
Network drivers

- **Host** removes network isolation between the docker host and docker containers. It uses the host's networking directly.
- Host networks are best when the network stack should not be isolated from the Docker host, but we want other aspects of the container to be isolated.



Network drivers

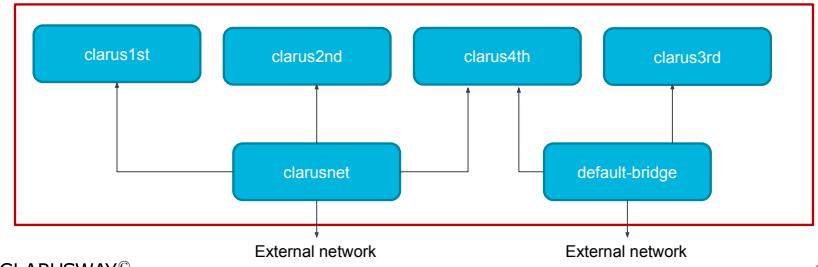
- **None** network driver disable all networking of containers.
- **None** network driver will not configure any IP for the container and doesn't have any access to the external network as well as for other containers.
- It is used when a user wants to block the networking access to a container.



User-defined bridge networks

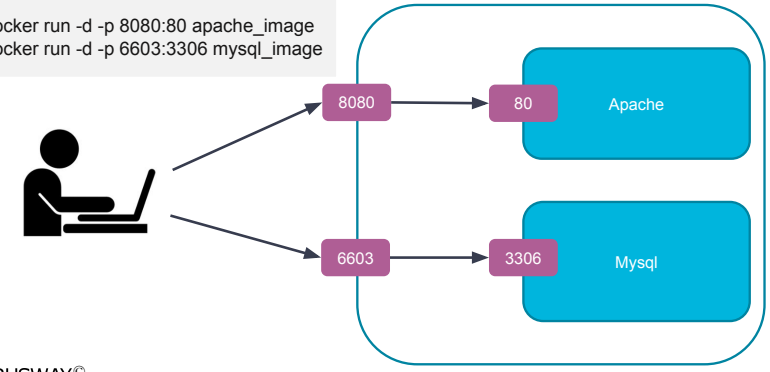
In addition to the default networks, users can create their own networks called **user-defined networks** of any network driver type.

```
$ docker network create --driver bridge clarusnet
```



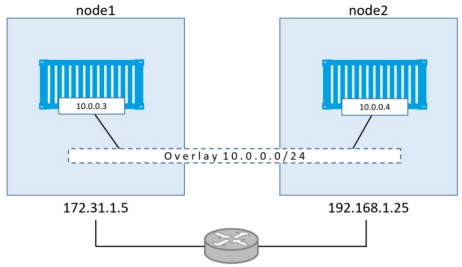
Run - Port mappings

```
$ docker run -d -p 8080:80 apache_image
$ docker run -d -p 6603:3306 mysql_image
```



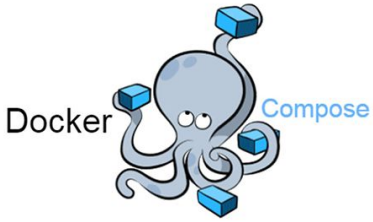
Network drivers

- **Overlay** networks connect multiple Docker daemons together and enable swarm services to communicate with each other.



What is Docker Compose?

- **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.
- Compose works in all environments: production, staging, development, testing, as well as CI workflows.



Docker Compose File

- The Compose file is a **YAML** file defining:
  - **services**,
  - **networks**
  - **volumes**
- The default path for a Compose file is ./docker-compose.yml.

```
version: "3.8"
services:
  web:
    build: .
    depends_on:
      - db
      - redis
  redis:
    image: redis
  db:
    image: postgres
```

```
version: '3.0'
services:
  web:
    build: .
    ports:
    - "5000:5000"
    volumes:
    - .:/code
    - logvolume01:/var/log
    links:
    - redis
  redis:
    image: redis
volumes:
  logvolume01: {}
```

THANKS!

Any questions?

You can find me at:

- alex.d@clarusway.com

Students, write your response!

Pear Deck Interactive Slide
Do not remove this bar