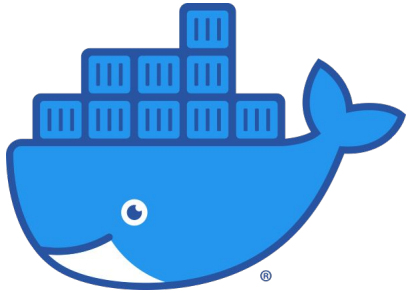


Docker

A Gentle Introduction

By: Pole

Motivation

Traditional developing way:

1. SSH into production server
2. Go to project directory
3. $ git pull
4. Install dependencies (curl, interpreter for our Language…)
5. Deploy our code

Motivation

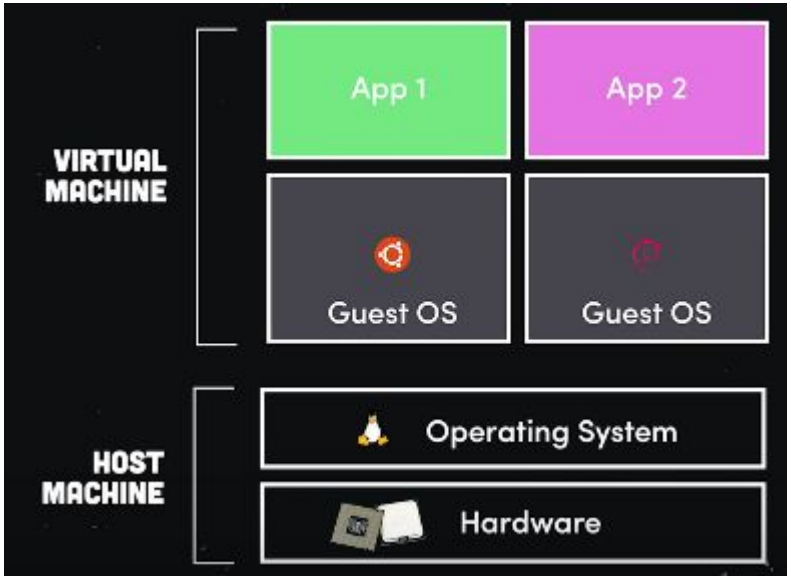
Running multiple applications on the same host:

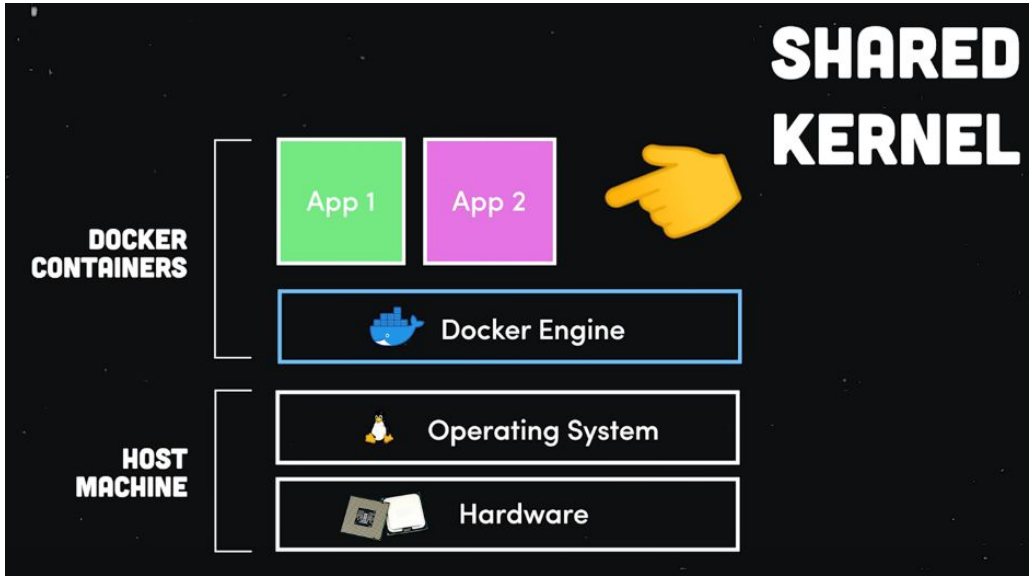
- Isolation for security reasons;
- Separate hosts for each application;
- Run in different VMs in the host

Solution: Docker

- Faster development;
- Encapsulation is easy;
- Same behavior on local machine, dev, staging and production server;
- Easy to scale;
- Good monitoring;
- Lighter than VMs (faster boot, less space taken, not OS-dependent);
- When using Docker on applications, we implicitly standardize our application (by following Docker's guidelines), hence making our application runnable on any environment, mainly because Docker is used by 99% of cloud services;

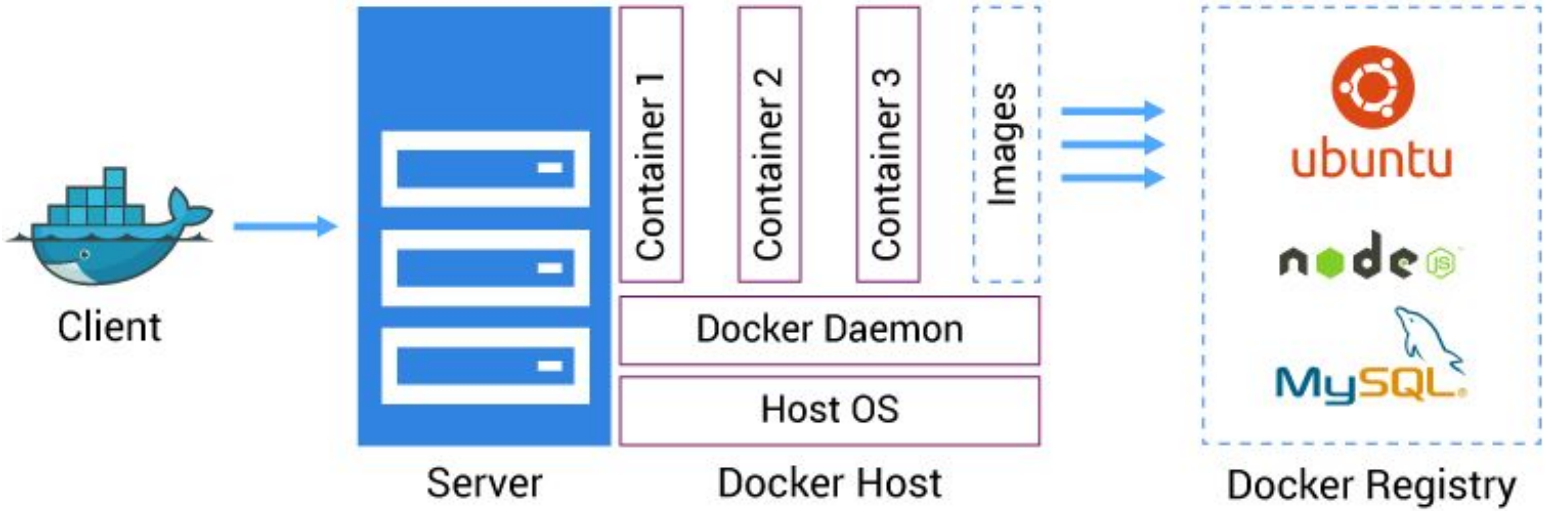
How it works?





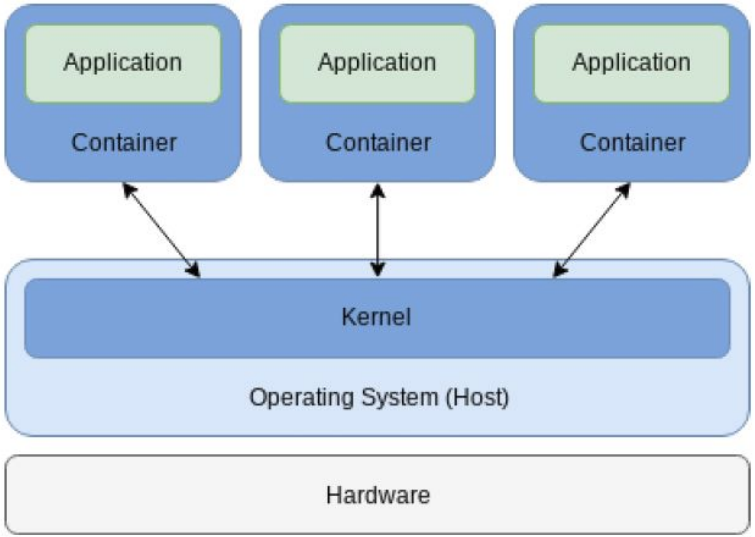
“Docker is just a fancy way to run a process, not a virtual machine.”

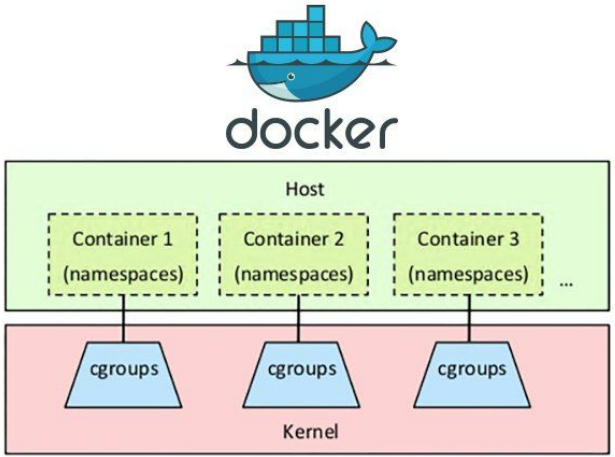
How it works?


Client
Container 1
Container 2
Container 3
Images
Docker Daemon
Host OS
Server
Docker Host
ubuntu
node
MySQL
Docker Registry

Why Docker excels?

- AuFS/overlay2/bttrfs: layered filesystem;
- LXC/runC (libcontainer): relies on namespaces and cgroups;
- Shares resources with the Host OS;





Why Docker excels?

- The layered filesystem is one of the factors why Docker excels;

- VM case:

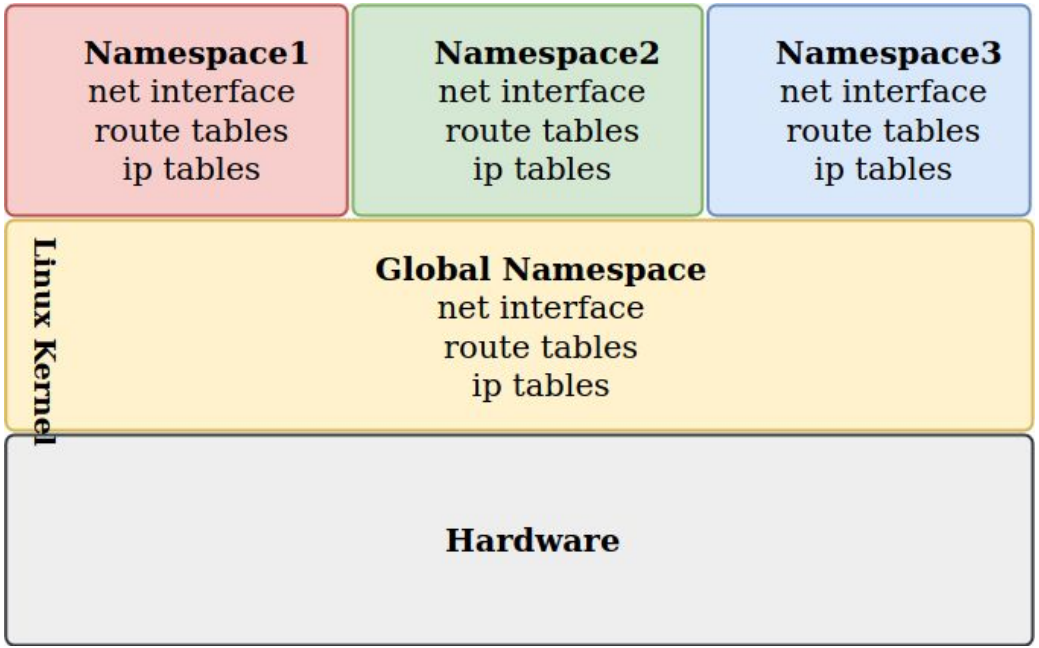
  1GB container image -> 1 VM that needs 1GB x NumberOfVMs

- Docker case: (layered filesystem)

  The 1GB Image is shared throughout the whole OS kernel, 1000 containers running?

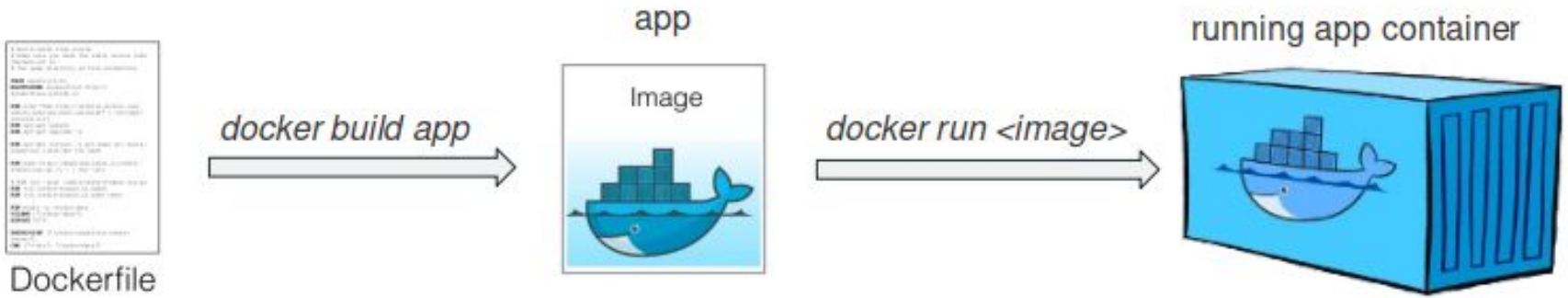
Why Docker excels?

- LXC (Linux Container): namespaes



Terminology

- Container: instance that encapsulates wished SW, created by images;
- Image: basic element of a container;
- Ports: a way to containers to talk with each other or with the Host OS
- Volume: shared directory



BORA METER MÃO EM DOCKER