Intro to Docker

Gianluca Rizzo

Why Docker?

- What does it do?

  Docker solves the **“same app, different computer” problem**

- (when software runs fine on one computer but fails—or behaves differently--on another)
- Why it happens? Differences in:
  - Operating systems
  - Settings
  - Required components (dependencies) and versions

The Challenge

Multiplicity of Stacks

Static website

nginx 1.5 + modsecurity + openssl + bootstrap 2

User DB

postgresql + pgv8 + v8

Queue

Redis + redis-sentinel

Analytics DB

hadoop + hive + thrift + OpenJDK

Background workers

Python 3.0 + celery + pyredis + libcurl + ffmpeg + libopencv + nodejs + phantomjs

Web frontend

Ruby + Rails + sass + Unicorn

API endpoint

Python 2.7 + Flask + pyredis + celery + psycopg + postgresql-client

Do services and apps interact appropriately?

Multiplicity of hardware environments

Development VM

QA server

Customer Data Center

Public Cloud

Disaster recovery

Production Servers

Production Cluster

Contributor's laptop

Can I migrate smoothly and quickly?

Cargo Transport Pre-1960

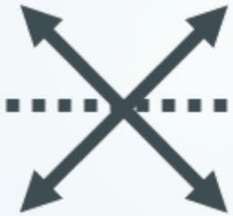
Multiplicity of Goods







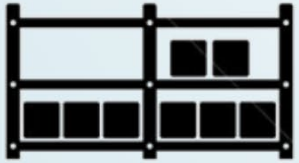
Do I worry about how goods interact (e.g. coffee beans next to spices)



Multipilicity of methods for transporting/storing



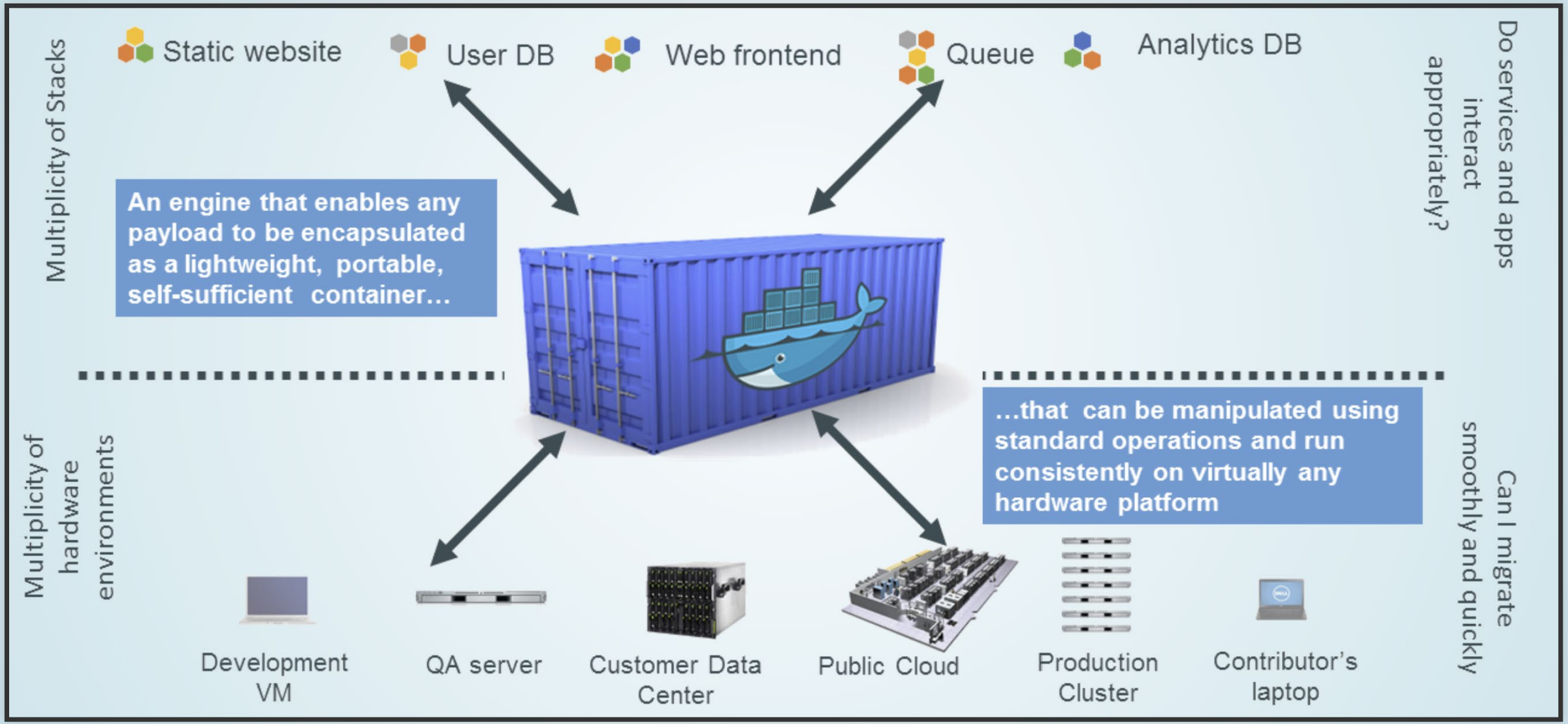



Can I transport quickly and smoothly (e.g. from boat to train to truck)

Solution: Intermodal Shipping Container

Docker is a Container System for Code

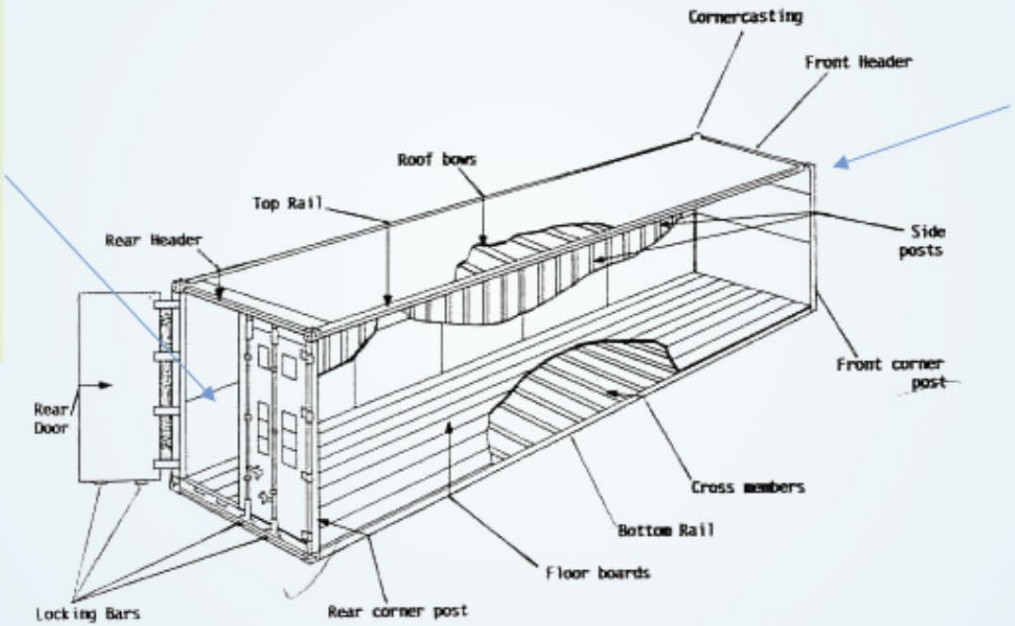


How does it solve it?

- Packaging software with everything it needs
  - And all configuration details
- And running it on a Docker app
- Result: it runs in the same way on all machines
  - Build once, run everywhere

Why it Works: Separation of Concerns

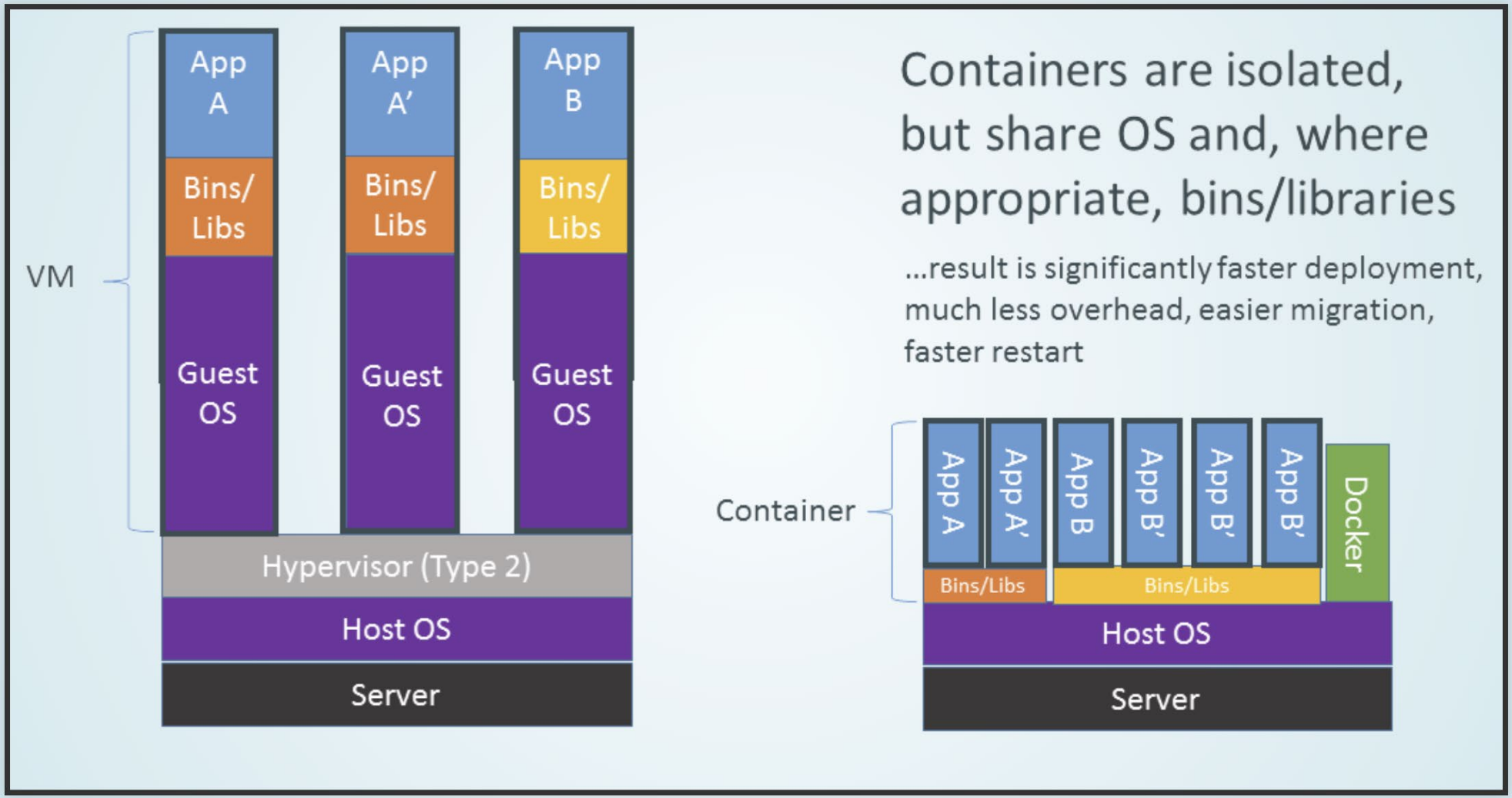
- **Dan the Developer**
  - Worries about what's "inside" the container
    - His code
    - His Libraries
    - His Package Manager
    - His Apps
    - His Data
  - All Linux servers look the same



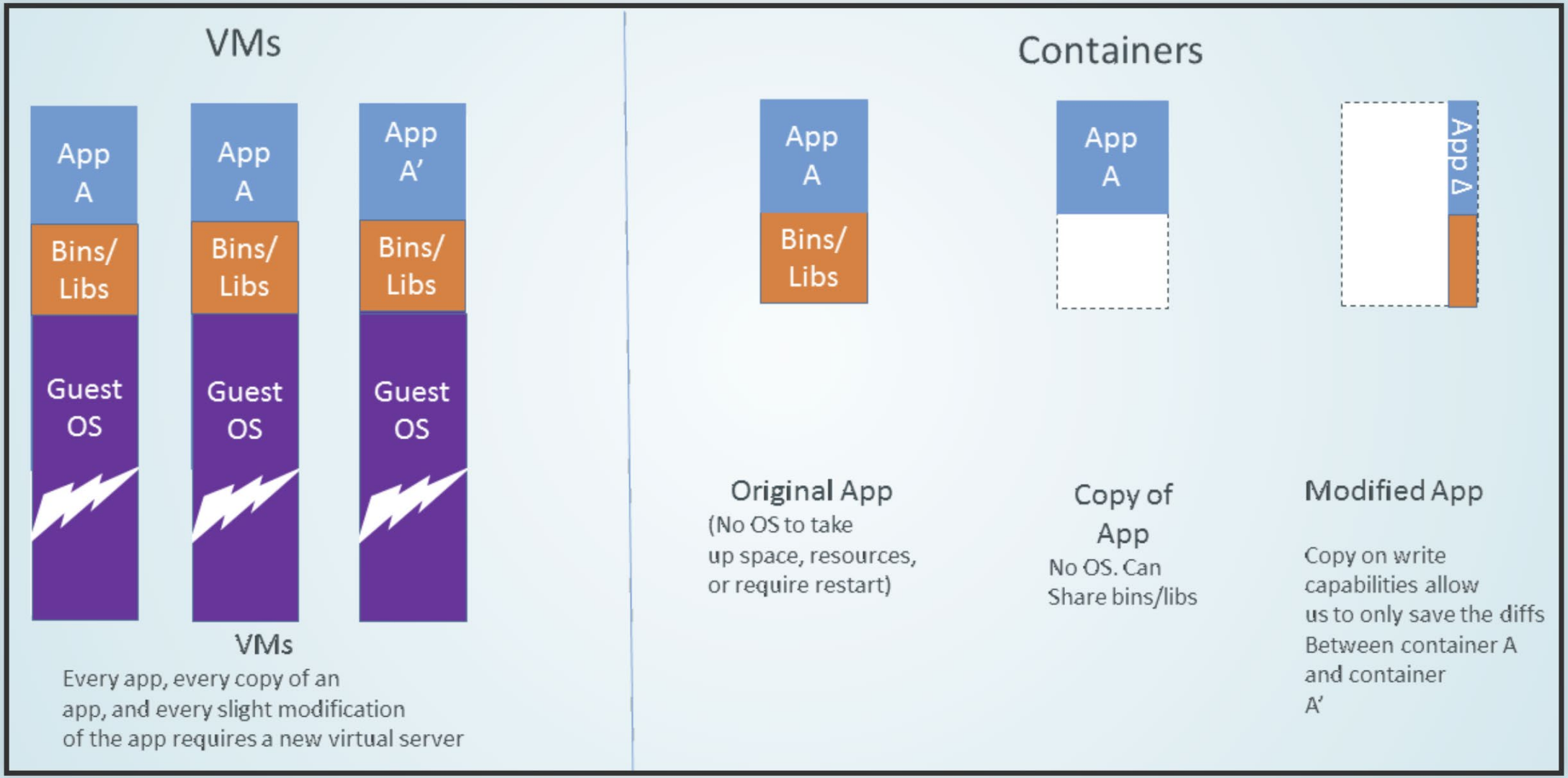
Major components of the container:

- **Oscar the Ops Guy**
  - Worries about what's "outside" the container
    - Logging
    - Remote access
    - Monitoring
    - Network config
  - All containers start, stop, copy, attach, migrate, etc. the same way

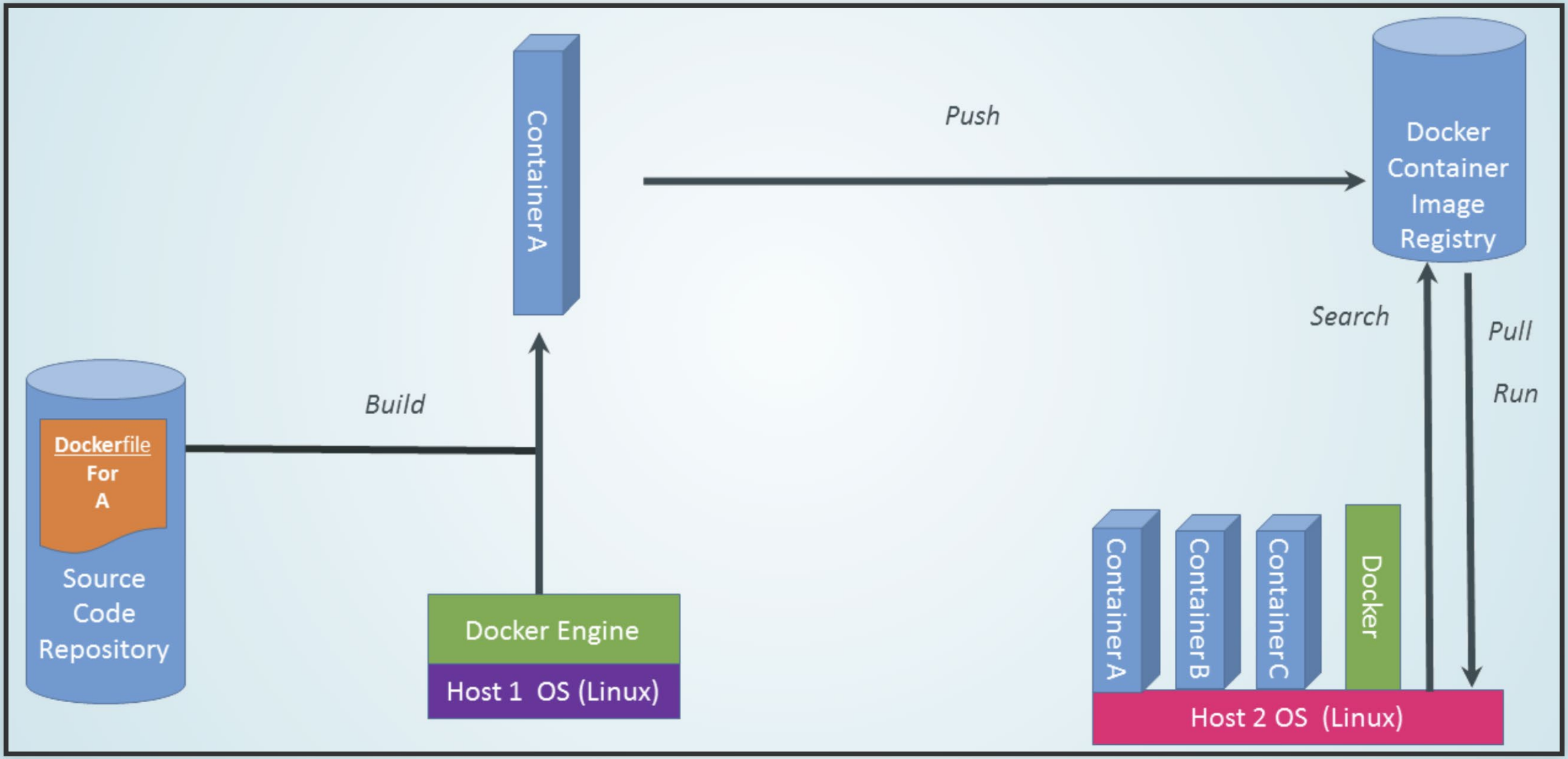
VMs vs Containers



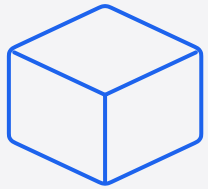
Why are Docker Containers Lightweight?



What are the Basics of a Docker System?



Container terminology



Container = Isolated Process

Not a virtual machine. Just a process. Runs independent of other containers and what's on the host machine

Image = Standard packaging

Contains all binaries, files, dependencies, and configuration needed to run the containerized process

Registry = Image repository

A centralized location for the hosting and distribution of container images. Can be available publicly or privately.