Introduction to Docker

Idaho State University | Computer Science

Isaac Griffith

CS 3321
Department of Computer Science
Idaho State University

ROAR

Virtualization

IDG CS 3321

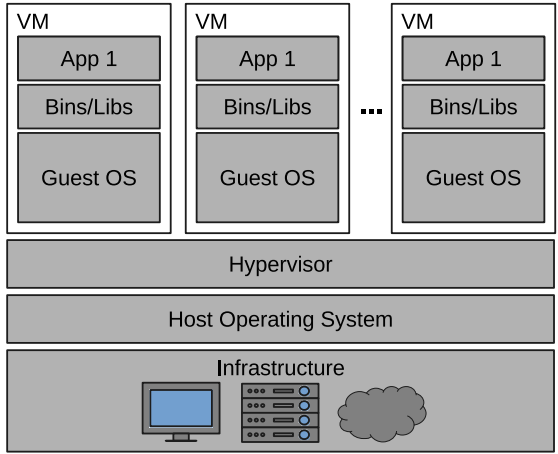
ROAR

Advantages of Virtualization

- Minimize hardware costs
  - Multiple virtual servers on one physical hardware
- Easily move VMs to other data centers
  - Provides disaster recovery. Hardware maintenance
  - Follow the sun (active users) or follow the moon (cheap power)
- Consolidate idle workloads. Usage is bursty and asynchronous
  - Increase device utilization
- Conserve power
  - Free up unused physical resources
- Easier automation
  - Simplified provisioning/administration of hardware and software
- Scalability and Flexibility: Multiple operating systems

Problems of Virtualization

- Each VM requires an operating system (OS)
  - Each OS requires a license
  - Each OS has its own compute and storage overhead
  - Needs maintenance, updates



Containers

IDG CS 3321

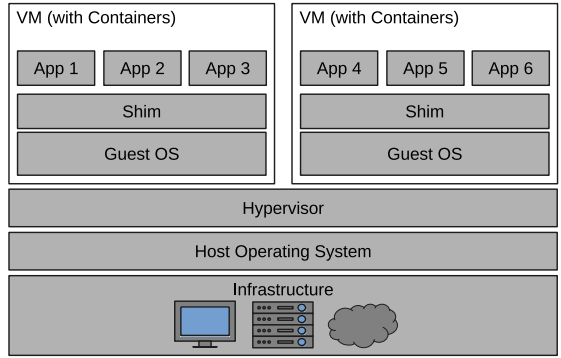
ROAR

Solution: Containers

- Run many apps in the same virtual machine
  - These apps share the OS and its overhead
  - But these apps can't interfere with each another
  - Can't access each other's resources without explicit permission
  - Like apartments in a complex

Containers

- Multiple containers run on one operating system on a virtual/physical machine
- All containers share the operating system
- Containers are isolated -> cannot interfere with each other
  - Own file system/data, own networking -> Portable



Containers

- Containers have all the good properties of VMs
  - Come complete with all files and data that you need to run
  - Multiple copies can be run on the same machine or different machine -> Scalable
  - Same image can run on a personal machine, in a data center or in a cloud
  - Operating system resources can be restricted or unrestricted as designed at container build time
  - Isolation: For example, “Show Process” (ps on Linux) command in a container will show only the processes in the container
  - Can be stopped. Saved and moved to another machine or for later run

VM vs. Containers

Criteria	VM	Containers
Image Size	3X	X
Boot Time	> 10s	~ 1s
Computer Overhead	> 10%	< 5%
Disk I/O Overhead	> 50%	Negligible
Isolation	Good	Fair
Security	Low-Medium	Medium-High
OS Flexibility	Excellent	Poor
Management	Excellent	Evolving
Impact on Legacy Application	Low-Medium	High

Docker

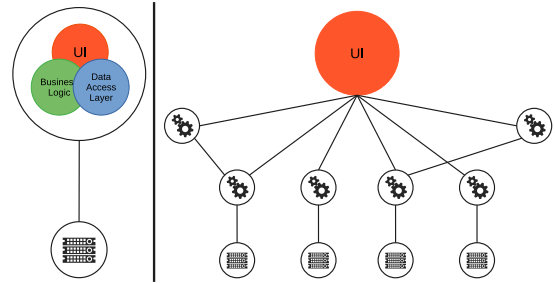
IDG CS 3321

ROAR

Idaho State University
Computer Science

Microservices

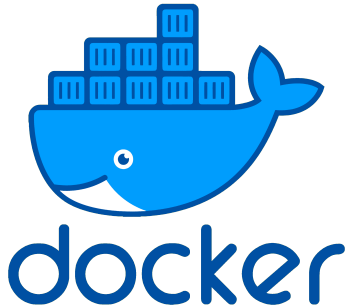
- Breaks large applications down into smaller executable components
- Easy to maintain and test
- Loosely coupled and can be deployed independently
- Can be combined with serverless architecture (i.e., AWS Lambda)



ROAR

Why Use Docker

- Develop applications that work on **any OS**
- Easy to **share** applications among teams
- Easy to **scale** across multiple servers
- Large applications can be broken into **multiple containers** - one for each microservice
- Great solution for **Cloud Computing**
- Big community and **library** of Docker images

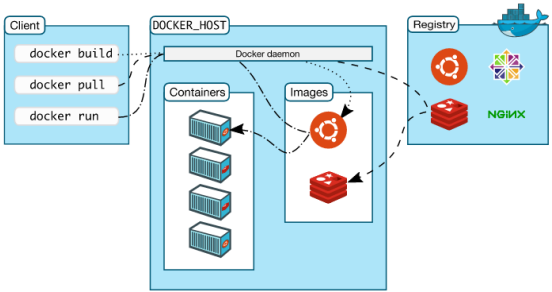


Serverless

- **Removes Dependency** on infrastructure
- Allows developers to focus on application development
- Microservices can be decoupled with different cloud services
- Usually more **cost effective**

Idaho State University
Computer Science

Docker



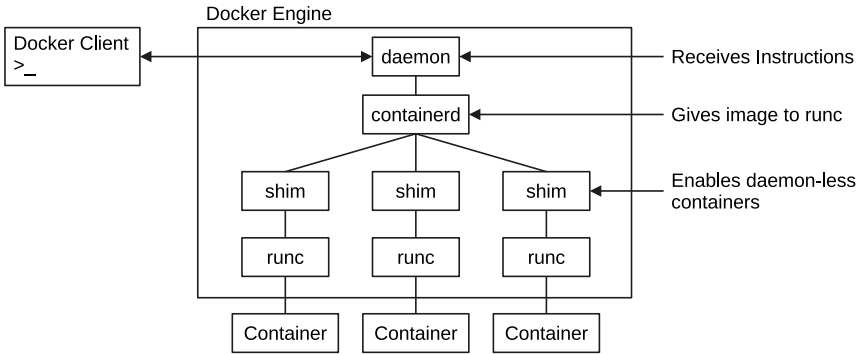
- Provides the isolation among containers
- Helps them share the OS
- Docker = Dock worker -> Manage containers
- Developed initially by Docker.com
- Downloadable for Linux, Windows, and Mac from docker.com
- Customizable with replacement modules from others

ROAR

Docker

- Docker Engine: Runtime
- Two Editions:
  - Community Edition (CE): Free for experimentation
  - Enterprise Edition (EE): For deployment with paid support
- Written in “Go” programming language from Google
- Now open source project under mobyproject.org
- Download the community edition and explore

Docker Engine Components



- daemon: API and other features
- containerd: Execution logic. Responsible for container lifecycle. Start, stop, pause, unpause, delete containers
- runc: A lightweight runtime CLI

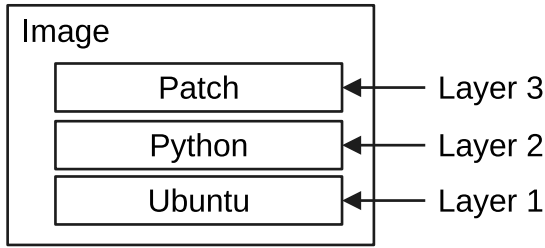
Image Registries

- Containers are built from images and can be saved as images
- Images are stored in registries
  - Local registry on the same host
  - Docker Hub Registry: Globally shared
  - Private registry on Docker.com
- Any component not found in the local registry is downloaded from specified location
- Official Docker Registry: Images vetted by Docker
- Unofficial Registry: Images not vetted (Use with care)
- Each image has several tags, e.g., v2, latest, …
- Each image is identified by its 256-bit hash

Layers

- Each image has many layers
- Image is built layer by layer
- Layers in an image can be inspected by Docker commands
- Each layer has its own 256-bit hash

- For example:
  - Ubuntu OS is installed, then
  - Python package is installed, then
  - a security patch to Python is installed
- Layers can be shared among many containers



Building Container Images

- Create a **Dockerfile** that describes the application, its dependencies and how to run it

```
FROM Alpine
LABEL maintainer="xx@gmail.com"
RUN apk add -update nodejs nodejs -npm
COPY ./src
WORKDIR /src
RUN npm install
EXPOSE 8080
ENTRYPOINT ["node", "./app.js"]
```

- **Note:** WORKDIR, EXPOSE, ENTRYPOINT result in tags. Others in Layers

Idaho State University
Computer Science

Docker commands

- `docker run`: Run the specified image
- `docker start`: start a stopped container
- `docker stop`: stop a container
- `docker ps`: shows the current process in a running container
- `docker image ls`: list running containers
- `docker rm`: delete a container
- `docker image rm`: delete an image
- `docker build`: creates a new image from docker file

ROAR

Idaho State University
Computer Science

Open Container Initiative (OCI)

- A company called CoreOS defined alternative image format and container runtime API's
- Led to formation of OCI under Linux Foundation to govern container standards
  - OCI Image spec
  - OCI Runtime spec
- Everyone including Docker is now moving to or on OCI

Idaho State University
Computer Science

Docker Compose

```yaml
version: 3

services:
  sysad-mysql:
    image: mysql:5.7
    environment:
      - "MYSQL_ROOT_PASSWORD=password"
    container_name: sysad-mysql
    restart: always
  sysad-apache2:
    image: sysad-crud
    ports:
      - "80:80"
    links:
      - "sysad-mysql"
    container_name: sysad-crud
    restart: always
    stdin_open: true
    tty: true
```

- Run multiple services in a single container
- Very useful for microservice applications
  - Service: Database
  - Service: Back-end server
  - Service: Front-end Web Client
- Specified as a **docker-compose.yml** file

ROAR

Docker Swarm

- Orchestrates thousands of Containers
- **Swarm**: a group of nodes collaborating over a network
- Two modes for Docker hosts:
  - Single Engine Mode: Not participating in a swarm
  - Swarm Mode: Participating in a Swarm
- A Service may run on a swarm
- Each swarm has a few managers that dispatch tasks to workers
  - Managers are also works (i.e., execute tasks)

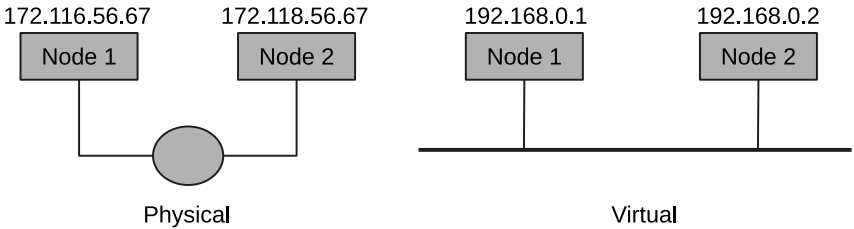
Docker Swarm

- The managers select a leader, who really keeps track of the swarm
- Assigns tasks, re-assigns failed worker's tasks, …
- Other managers just monitor passively and re-elect a leader if the leader fails
- Services can be scaled up or down as needed
- Several Docker commands:
  - `docker service`: Manage services
  - `docker swarm`: Manage swarms
  - `docker node`: Manage nodes

Idaho State University
Computer Science

Docker Overlay Networking

- Nodes in a swarm may not be in the same LAN
- VXLAN is used to provide virtual overlay networking



ROAR

Docker Security

- All built-in security mechanisms in Linux are used and more
- Cryptographic node IDs
- Mutual Authentication
- Automatic Certificate Authority configuration
- Automatic Certificate Renewal on expiration
- Encrypted Cluster Store
- Encrypted Network traffic
- Signed images in Docker Content Trust (DCT)
- Docker Security Scanning detects vulnerabilities
- Docker secrets are stored in encrypted cluster store, encrypted transmission over network, and stored in in-memory file system when in use

Kubernetes

- Open Source Container Orchestration alternative
- Original source released by Google
- Cloud Native Computing Foundation (CNCF) project in Linux Foundation
- Pre-cursor to Swarms
- Facilities similar to Swarms
- A set of related containers is called a “Pod”
  - A Pod runs on a single host
- Swarm is called a “Cluster”

Are there any questions?