Docker

Haoyi Wang

10/12/2024

UNIVERSITY OF PLYMOUTH

What we will cover today

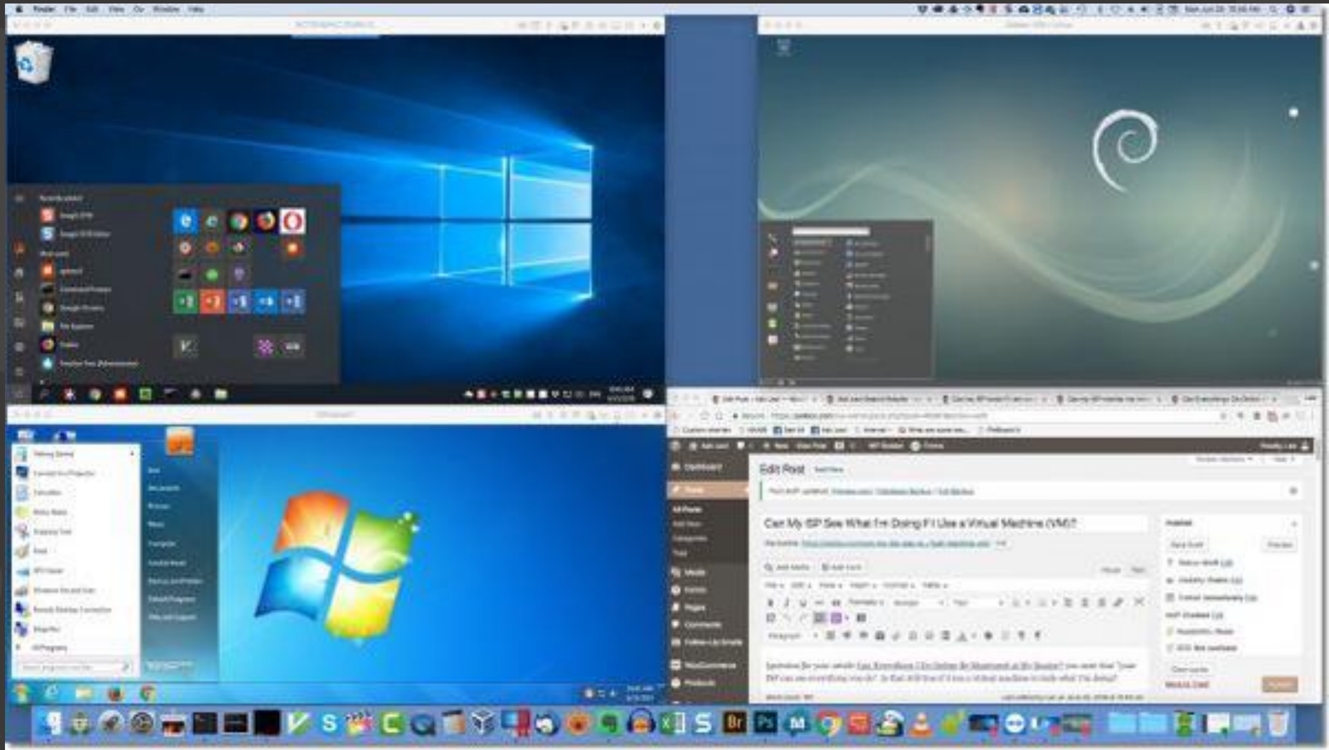
- Why Docker?
- What is Docker?
- Basic Usage
- Docker Engine/Images/Containers

UNIVERSITY OF PLYMOUTH

Why Docker?

- Old days – one application per server
  - Big fast servers – Waste of money and environmental resources

- Virtual machine (VM)
  - Run multiple applications on one server



Why Docker?

- VMs are great, but…
  - Every VM requires its own dedicated OS, and every OS consumes CPU, RAM and other resources
  - Slow to boot

- Big tech companies have been using containers
  - Do not require their own OS
  - Reduce potential licensing costs and overhead of OS patching and other maintenance

Why Docker?

- Linux Containers (LXC) were relatively complex to setup and manage.
- To run a simple web server inside an LXC, you need to:
  - Set up control groups and namespaces by hand or through complex LXC configuration files
  - Install and configure the necessary packages inside that container environment
  - Manually manage filesystem isolation and networking

UNIVERSITY OF PLYMOUTH

With Docker: `docker run -p 80:80 nginx`

What about Windows containers?

- Windows containers run Windows apps that require a host system with a Windows kernel
- Any Windows host running the Windows Subsystem for Linux (WSL) can also run Linux containers
- Most containers are Linux containers because they are smaller and faster

No Mac containers

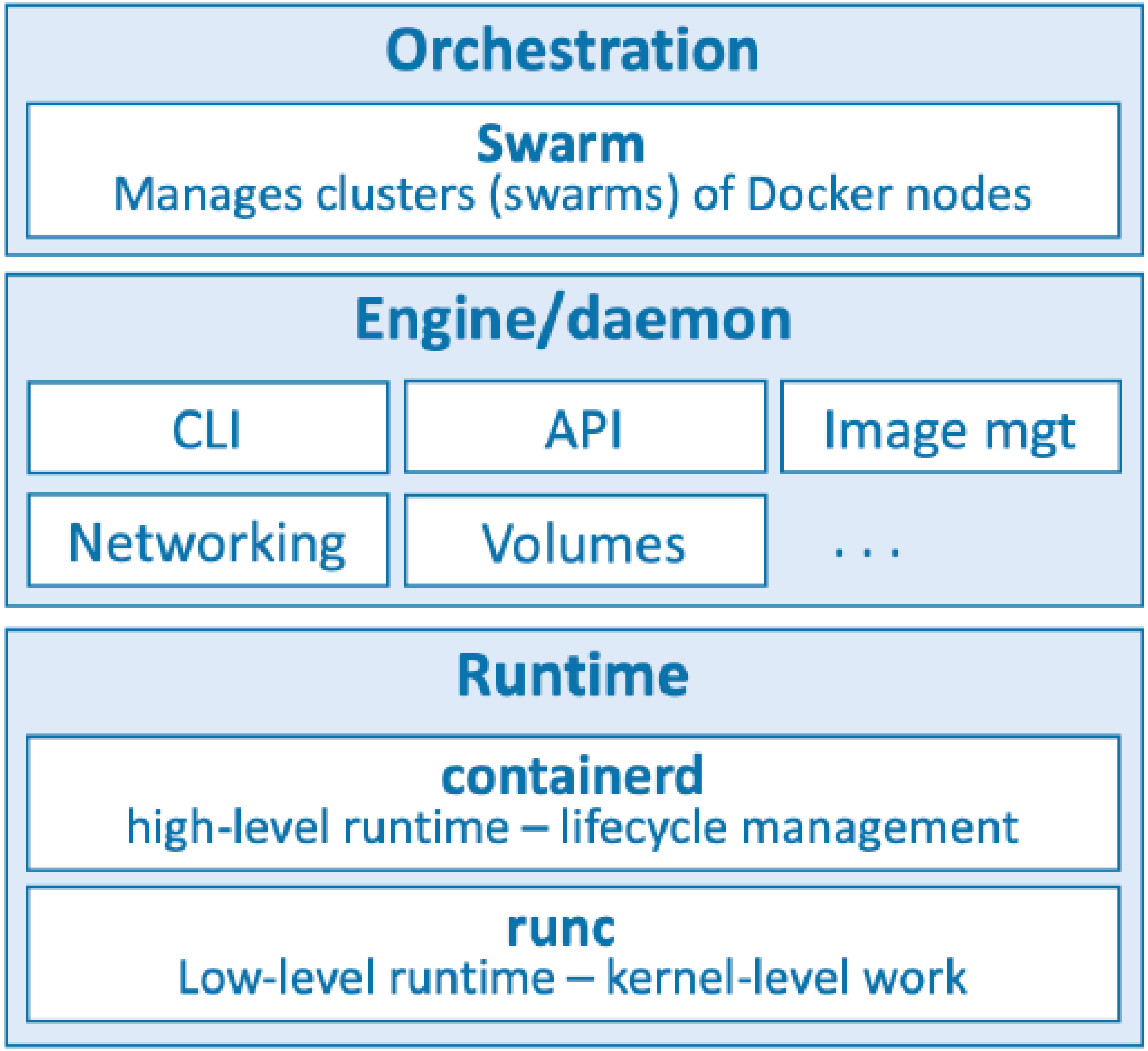
Docker

- Runs on Linux/Mac and Windows
- Creates, manages, and orchestrates containers
- Built from various tools from the Moby open-source project
  - https://github.com/moby/moby

Docker Arch

- The runtime
  - Start and stop containers
- The daemon
  - Expose API
  - Manage images
  - …
- The orchestrator
  - Manage clusters

UNIVERSITY OF
PLYMOUTH



Orchestration

Swarm
Manages clusters (swarms) of Docker nodes

Engine/daemon

CLI
API
Image mgt
Networking
Volumes
. . .

Runtime

containerd
high-level runtime – lifecycle management

runc
Low-level runtime – kernel-level work

Basic Usage

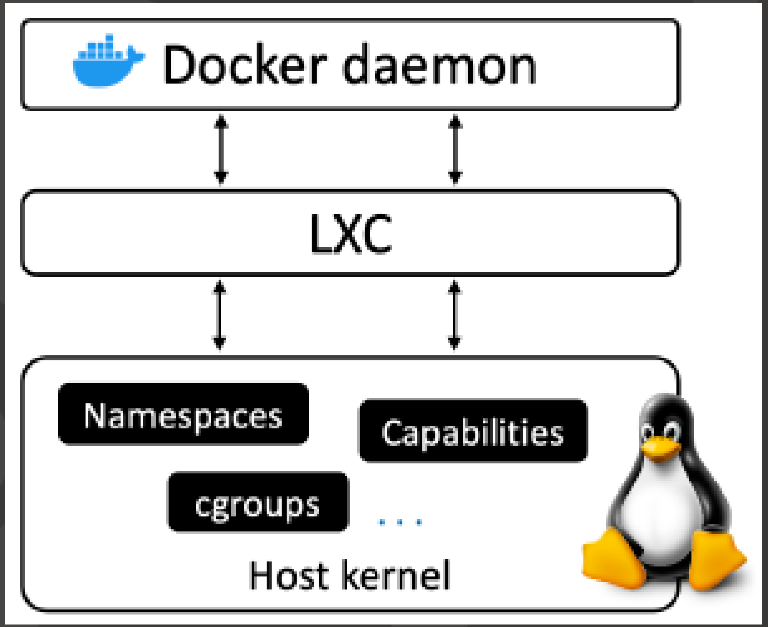
- Ops
  - Start, log in, run commands, destroy
- Dev
  - Dockerfile, run containers

Docker Engine

- The old Docker architecture has two major components:
  - The Docker daemon
  - LXC

- Problem with LXC:
  - Linux-specific
  - Huge risk to rely on an external tool

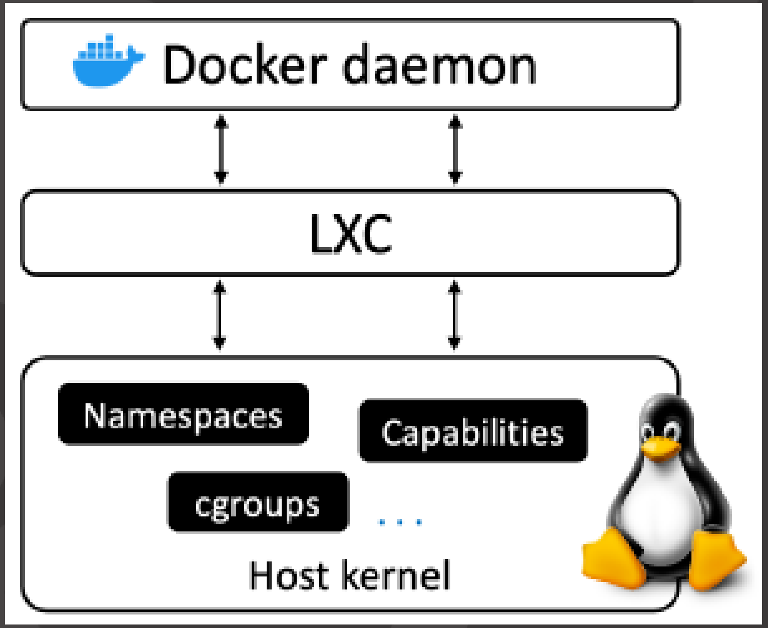
- libcontainer



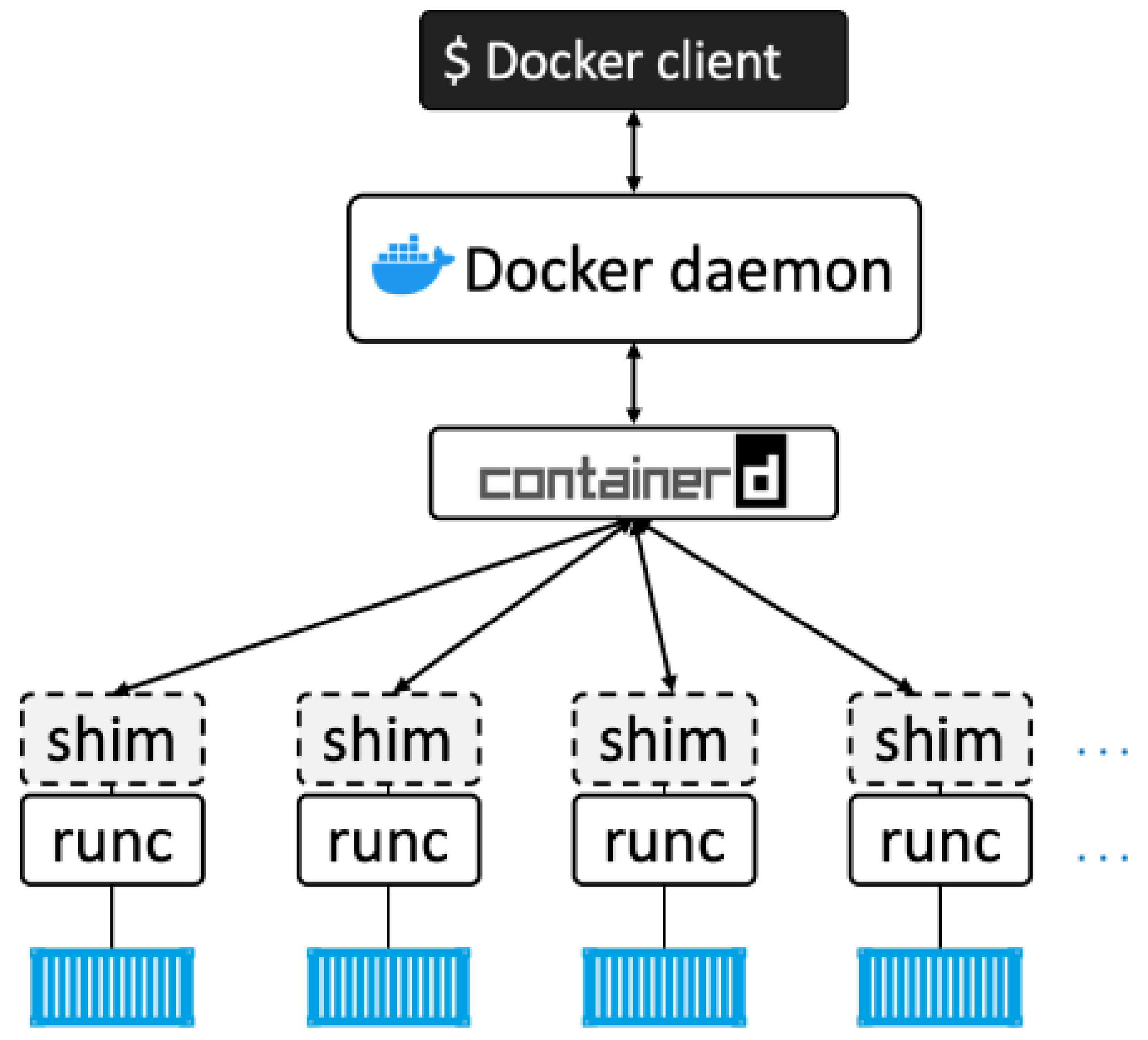
UNIVERSITY OF PLYMOUTH

Problems with Old Docker Engine

- It's hard to innovate on
  - Tightly coupled
- It got slower
  - Performance bottlenecks
- It wasn't what the ecosystem wanted
  - Wasn't flexible enough



New Architecture



Docker commands (CLI)

API and other features

Container supervisor

Enables daemonless containers

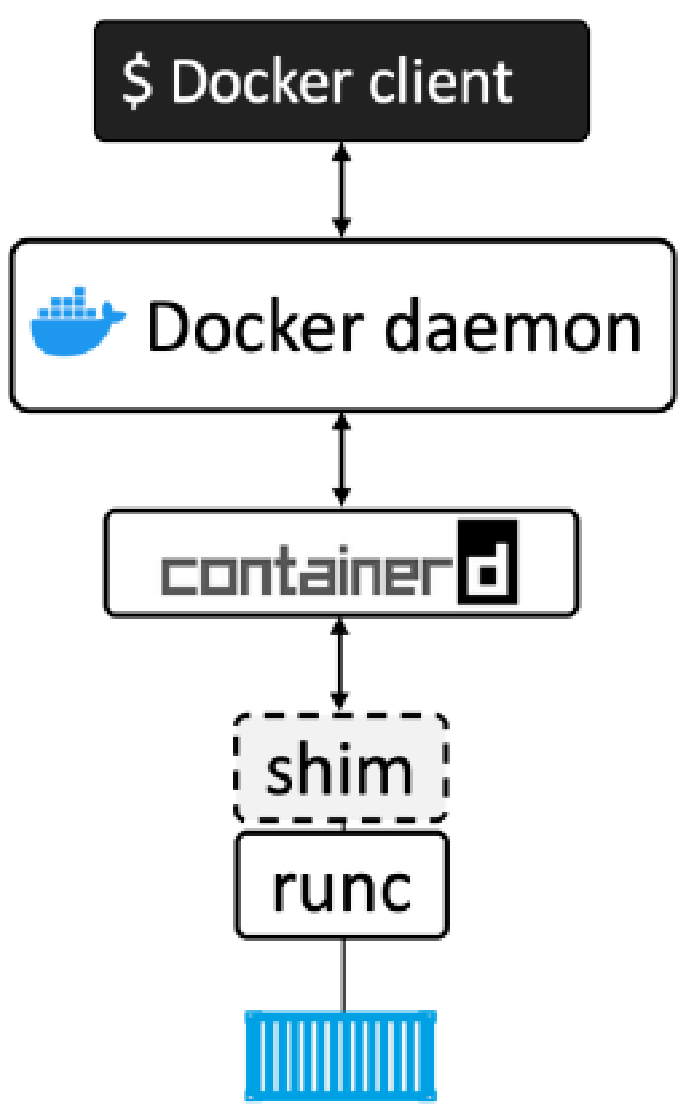
Container runtime

Running containers

New Architecture

- runc
  - Small, lighweight CLI wrapper for libcontainer to create containers
- containerd
  - Has all of the container execution logic
- shim
  - Decouples running containers from the daemon

Starting a new container



Issue ‘`docker container run`’ command to Docker API exposed by Docker daemon

Receive instruction at API endpoint. Instruct containerd (via gRPC API) to start new container

Receive instruction to create containers
Instruct runc to create containers

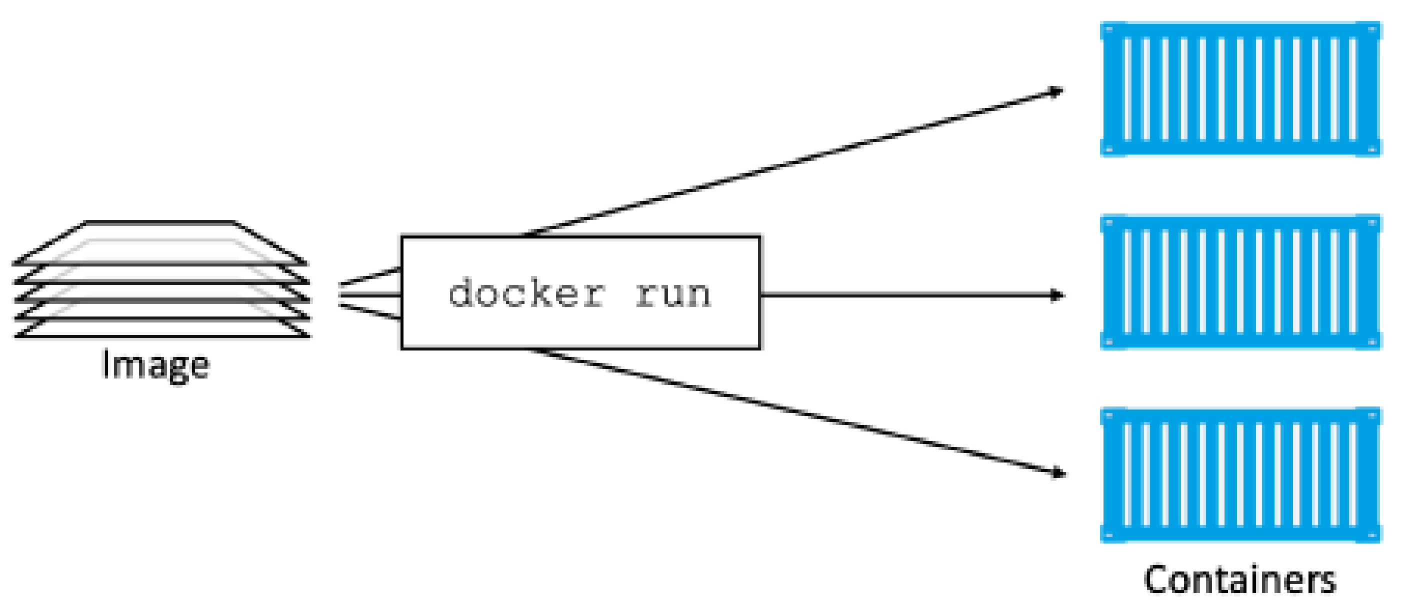
Build and start container
runc exits after container starts
shim becomes container’s parent process

Purpose of Docker daemon

- Push and pull images
- Implement the Docker API
- Authentication
- Security
- …

Images and Containers

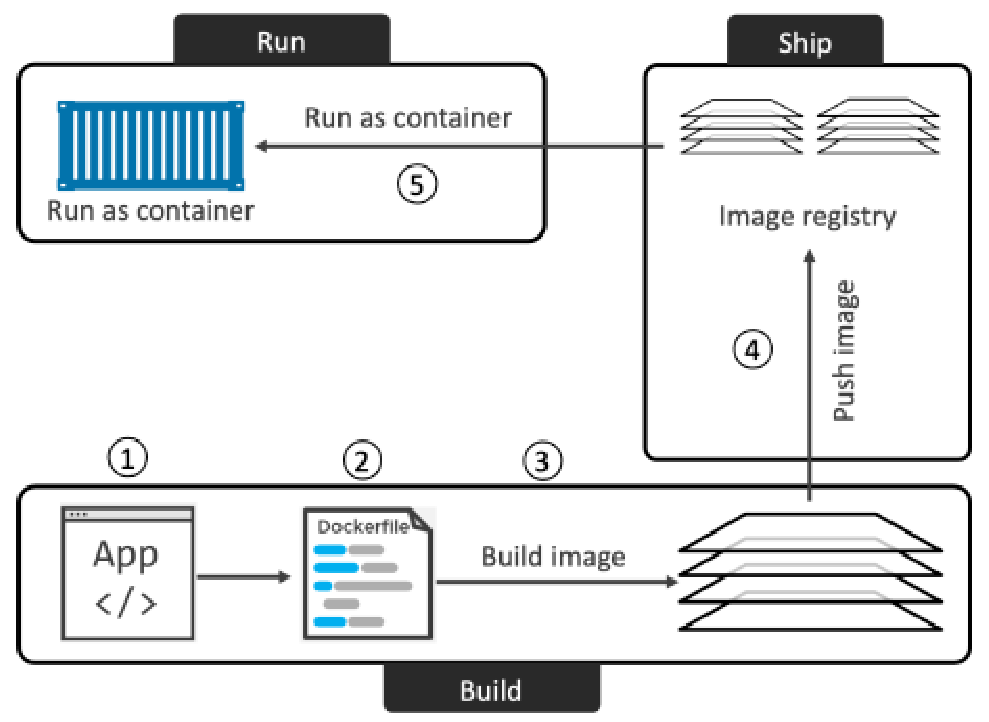
- A container is the runtime instance of an image
- Images and containers are dependent on each other
- Images are stored in registries – Docker Hub



UNIVERSITY OF PLYMOUTH

Containerizing an app

- Steps:
  1. Start with your application code and dependencies
  2. Create a Dockerfile
  3. Build it into an image
  4. Push the image to a registry (optional)
  5. Run a container from the image



UNIVERSITY OF PLYMOUTH

Exercise

Brainstorm potential use cases for Docker in different industries (e.g., finance, healthcare, education).

What we covered today

- Why Docker?
- What is Docker?
- Basic commands
- Docker architecture (old vs. new), images, containers, …

Questions?

UNIVERSITY OF
PLYMOUTH