

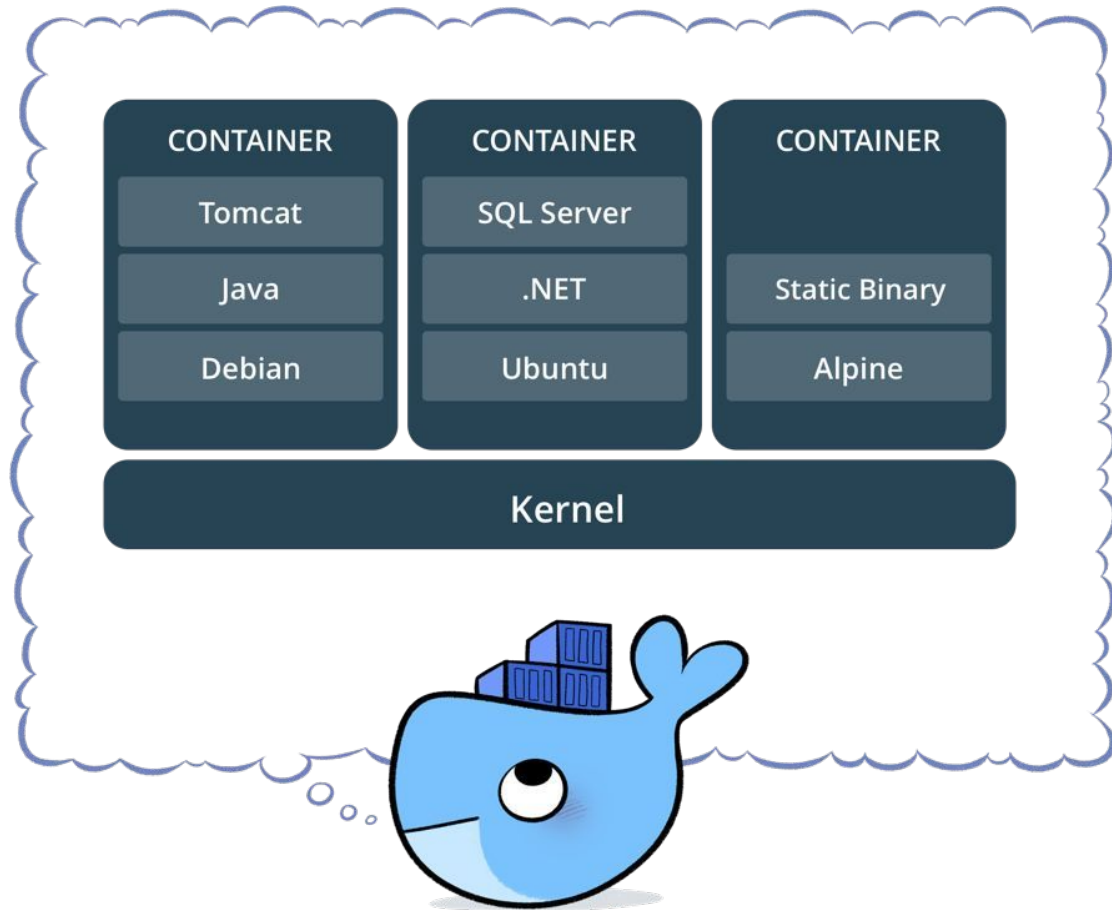
Presentation – 3

Docker containers

Test on VMs

- Google form with test:

What is a container?



- Standardized packaging for software and dependencies
- Isolated applications sharing the same OS kernel
- Supported on Linux and Windows

Terminology



Docker Image

The basis of a Docker container. Represents a full application



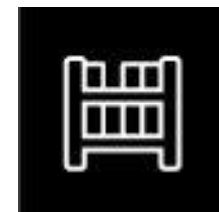
Docker Container

The standard unit in which the application service resides and executes



Docker Engine

Creates, ships and runs Docker containers deployable on a physical or virtual, host locally, in a datacenter or cloud service provider



Registry Service (Docker Hub (Public) or Docker Trusted Registry (Private))

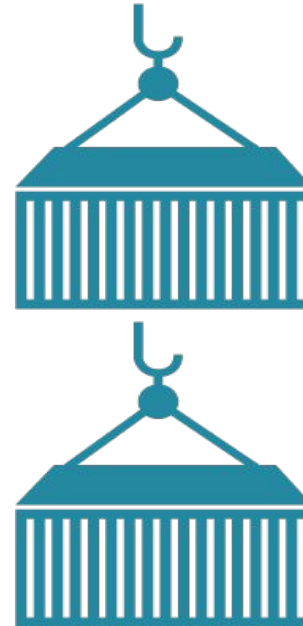
Cloud or server-based storage and distribution service for your images

Images and Containers



Docker Image

Examples: Nginx web server, DB server,
Nodejs Application



Docker Containers

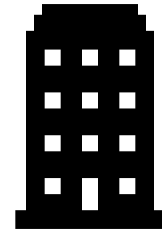
Each container is created from an image

Docker containers are NOT VMs

- Fundamentally different architectures
- Easy to manage
- No need to install separate OS

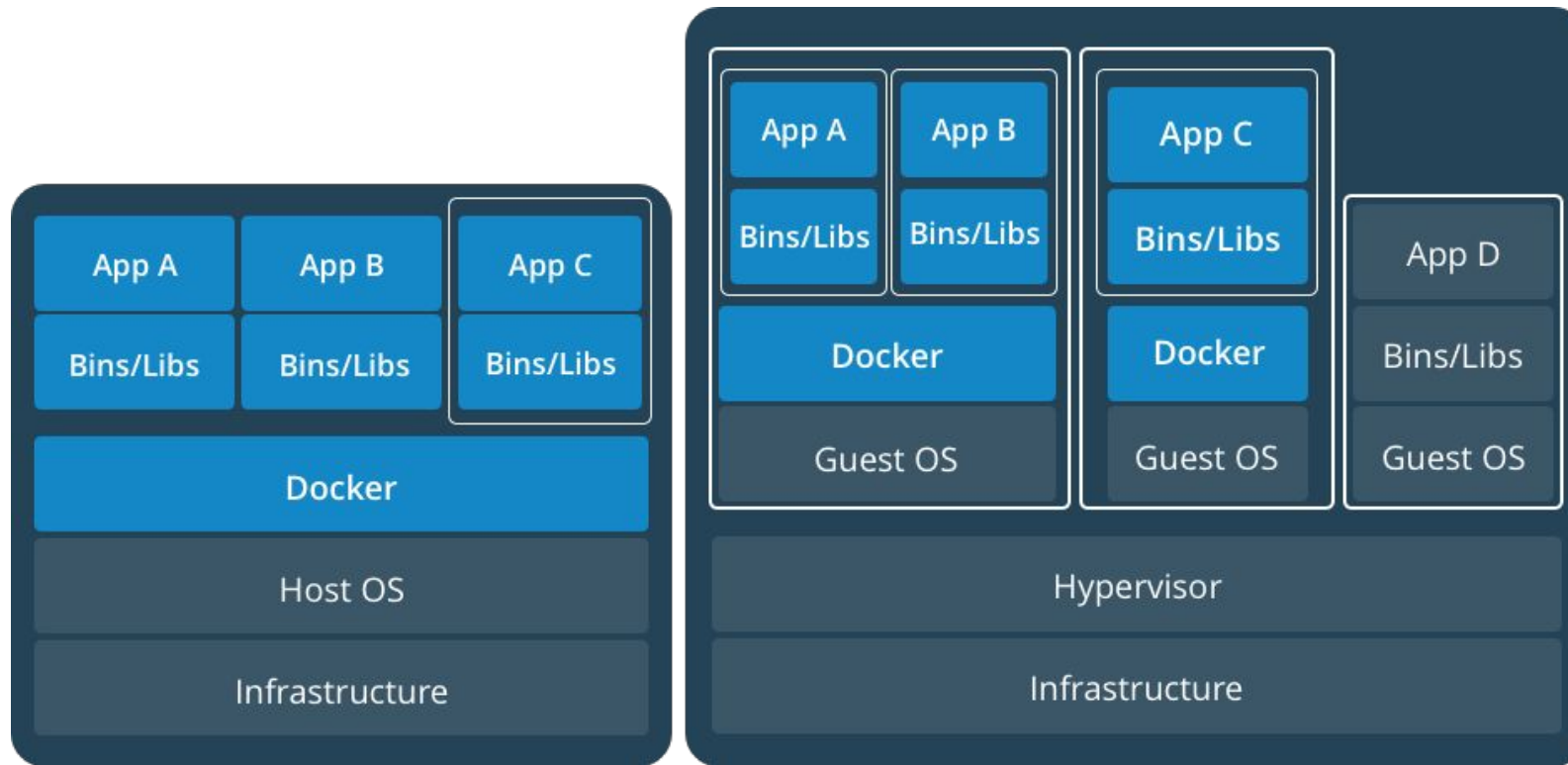


Virtual machine



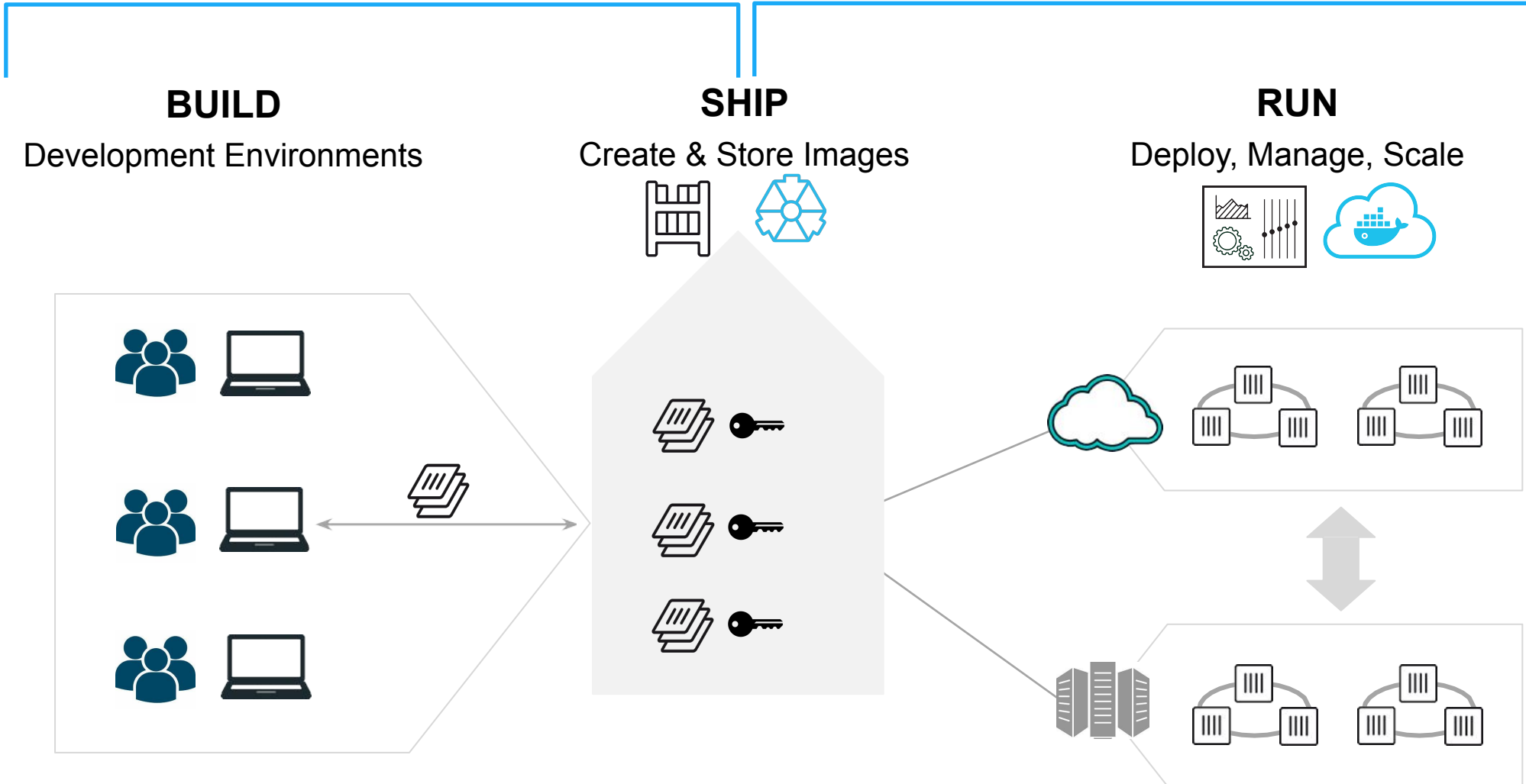
Containers on one host machine

Docker can work on virtual machines



Using Docker: Build, Ship, Run Workflow

Developers IT Operations



Basic Docker Commands

```
$ docker image pull node:latest
```

```
$ docker image ls
```

```
$ docker container run -d -p 5000:5000 --name node node:latest
```

```
$ docker container ps
```

```
$ docker container stop node(or <container id>)
```

```
$ docker container rm node (or <container id>)
```

```
$ docker image rmi (or <image id>)
```

```
$ docker build -t node:2.0 .
```

```
$ docker image push node:2.0
```

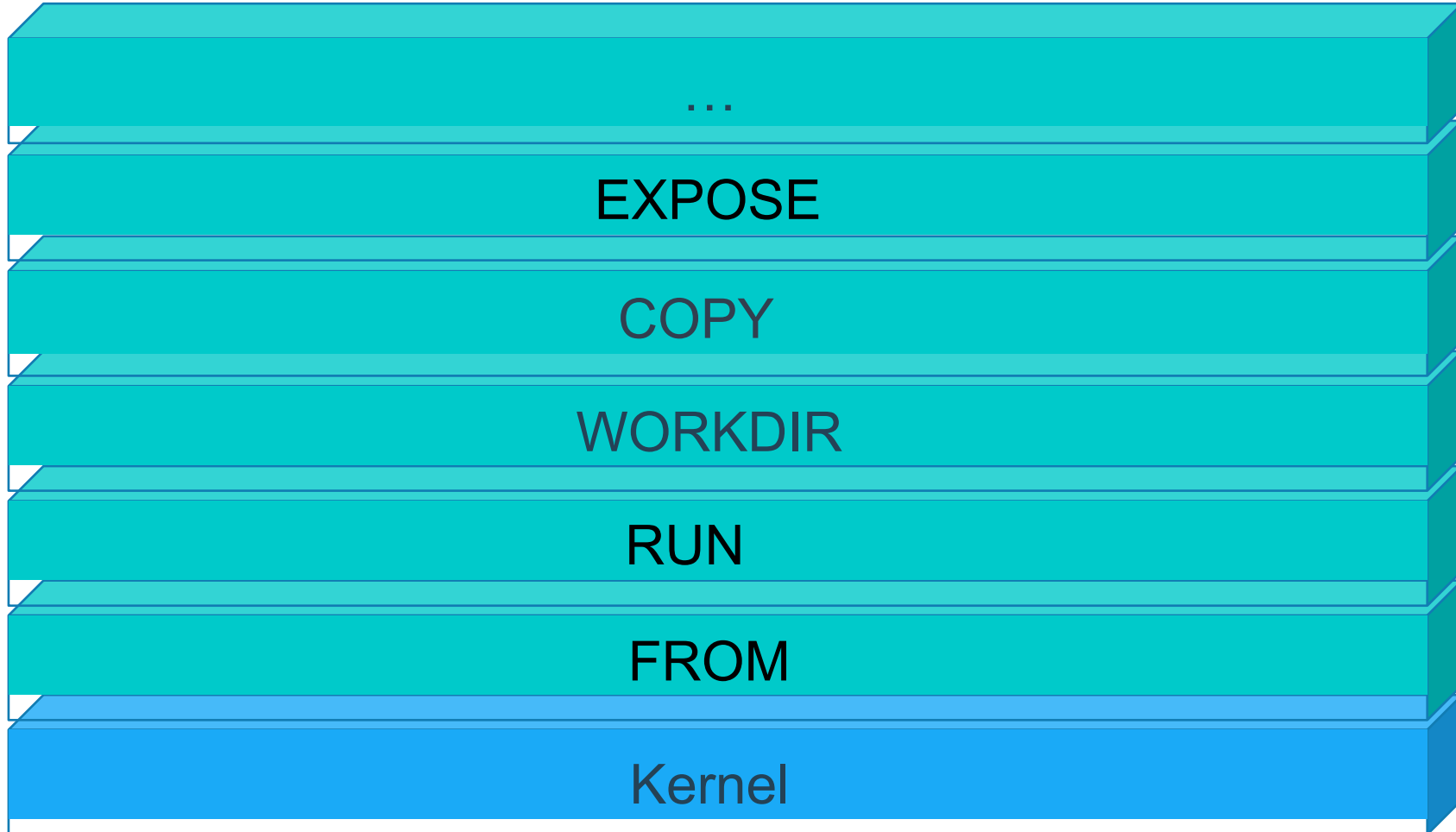
```
$ docker --help
```

Image can be easily created with **dockerfile**

```
Dockerfile x
1  # Create image based on the official Node 6 image from dockerhub
2  FROM node:latest
3
4  # Create a directory where our app will be placed
5  RUN mkdir -p /usr/src/app
6
7  # Change directory so that our commands run inside this new directory
8  WORKDIR /usr/src/app
9
10 # Copy dependency definitions
11 COPY package.json /usr/src/app
12
13 # Install dependencies
14 RUN npm install
15
16 # Get all the code needed to run the app
17 COPY . /usr/src/app
18
19 # Expose the port the app runs in
20 EXPOSE 4200
21
22 # Serve the app
23 CMD ["npm", "start"]
```

- Instructions on how to build a Docker image
- Looks very similar to “native” commands
- Important to optimize your Dockerfile

Each Dockerfile Command Creates a Layer



Docker Volumes – how to avoid data loss?

- Volumes mount a directory on the host into the container at a specific location
- Can be used to share (and persist) data between containers
 - Directory persists after the container is deleted
 - Unless you explicitly delete it
- Can be created in a Dockerfile or via CLI

Attaching a container directly to you source code folder

- You can mount local source code into a running container

```
docker container run -v  
$(pwd):/usr/src/app/ myapp
```

- Improve performance
 - As directory structures get complicated traversing the tree can slow system performance
- Data persistence



Path to working directory

Other topics to study at home and on practice classes

- Networking
 - How to connect containers on the same host. They will not see one another if network is not configured
- Docker compose
 - How to create a package of several containers.

Practice and homework

- Practice:
 - Basic operations in docker
- Homework:
 - Use docker compose to create a software package containing:
 - A database server
 - A web site that is connected to the database