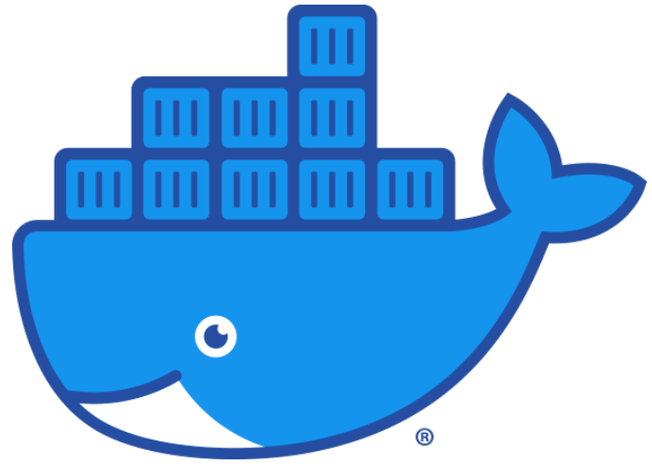
[544] Docker Deployment

Tyler Caraza-Harter



Outline

Virtualization

Images, Containers, and Dockerfiles

Ports

Docker Compose

What is virtualization?

Definition: the illusion of **private** resources, provided by software

Contexts this semester

- Virtual Memory
- Virtual Machines (hardware)
- Virtual Machines (languages)
- Virtual Operating System (container) *new today*

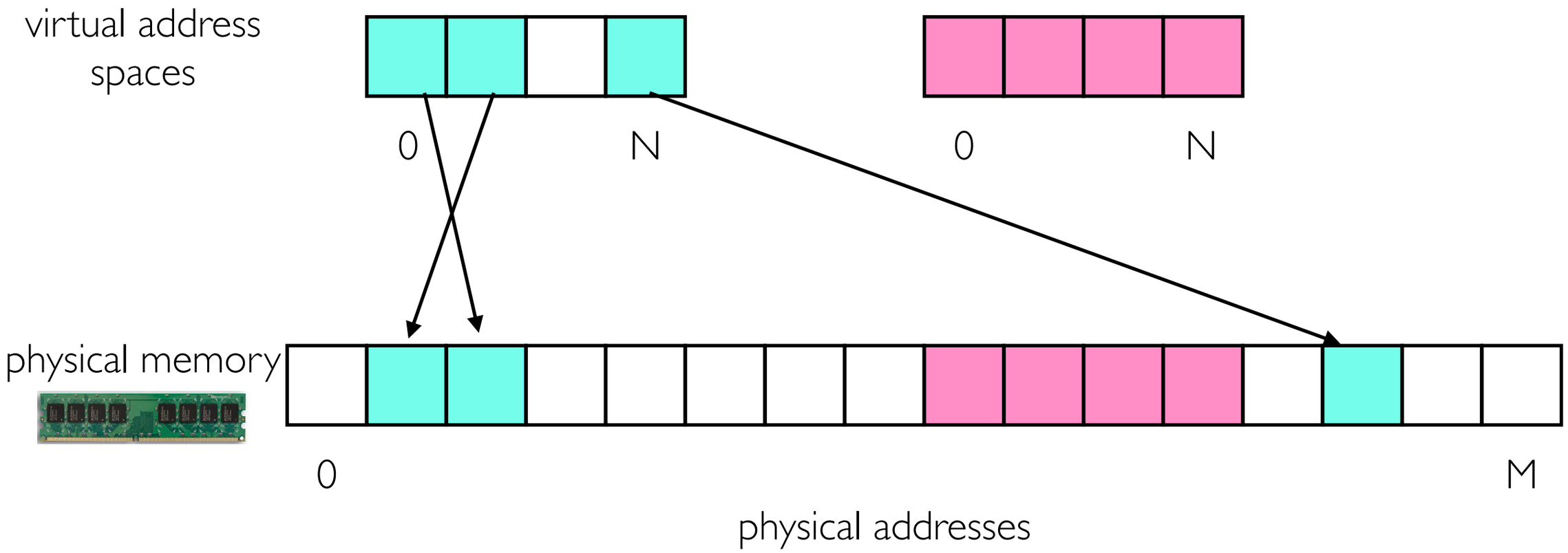
What is virtualization?

Definition: the illusion of **private** resources, provided by software

Contexts this semester

- Virtual Memory
- Virtual Machines (hardware)
- Virtual Machines (languages)
- Virtual Operating System (container)

each process using a virtual address space
isn't aware of other processes using memory
(illusion of private memory)



What is virtualization?

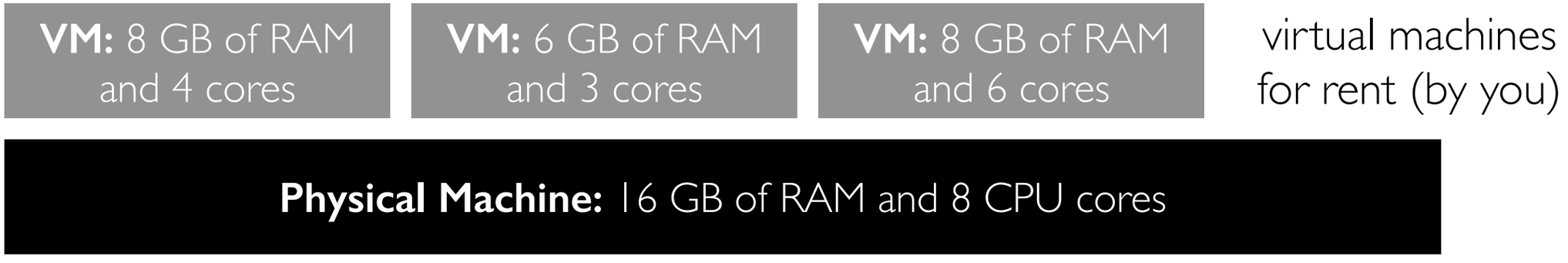
Definition: the illusion of **private** resources, provided by software

Contexts this semester

- Virtual Memory
- Virtual Machines (hardware)
- Virtual Machines (languages)
- Virtual Operating System (container)

virtualized resources include CPU, RAM, disks, network devices, etc

VMs rarely use all their allocated resources, so overbooking is possible



What is virtualization?

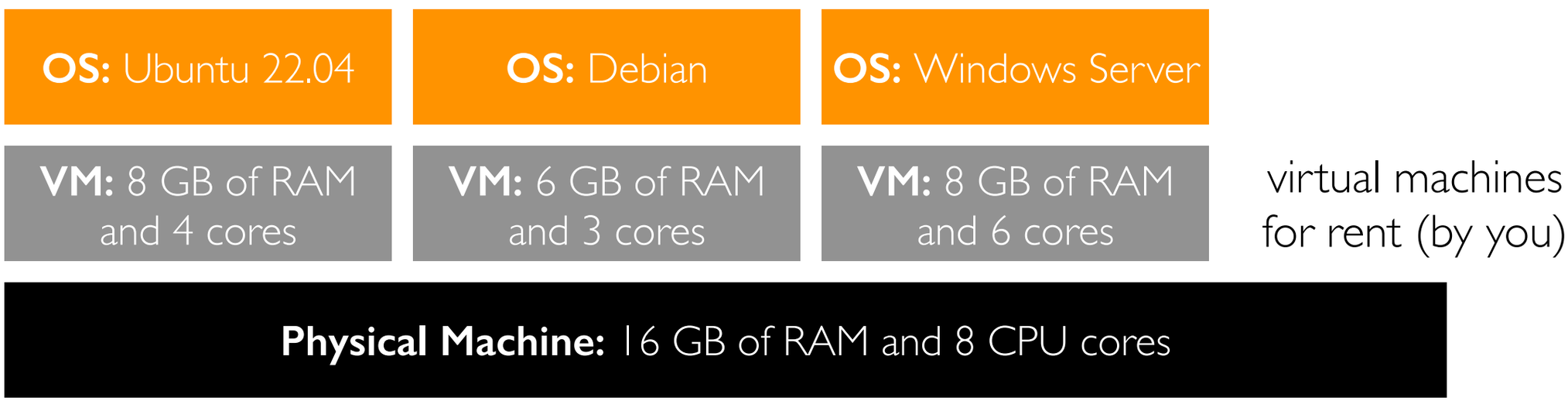
Definition: the illusion of **private** resources, provided by software

Contexts this semester

- Virtual Memory
- Virtual Machines (hardware)
- Virtual Machines (languages)
- Virtual Operating System (container)

problem: if each program is deployed to a different VM, operating system overheads will dominate

these operating systems are mostly unaware that their on VMs instead of physical hardware



virtual machines for rent (by you)

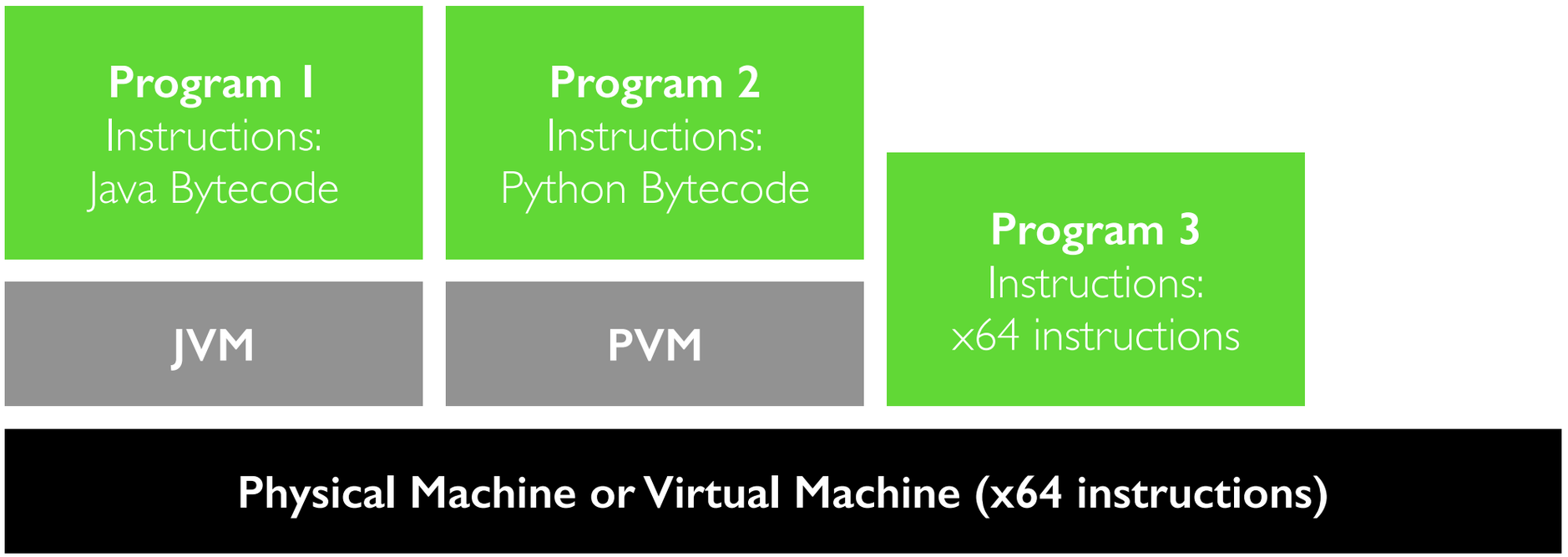
actual hardware bought by Google for their cloud services (GCP)

What is virtualization?

Definition: the illusion of **private** resources, provided by software

Contexts this semester

- Virtual Memory
- Virtual Machines (hardware)
- Virtual Machines (languages)
- Virtual Operating System (container)



What is virtualization?

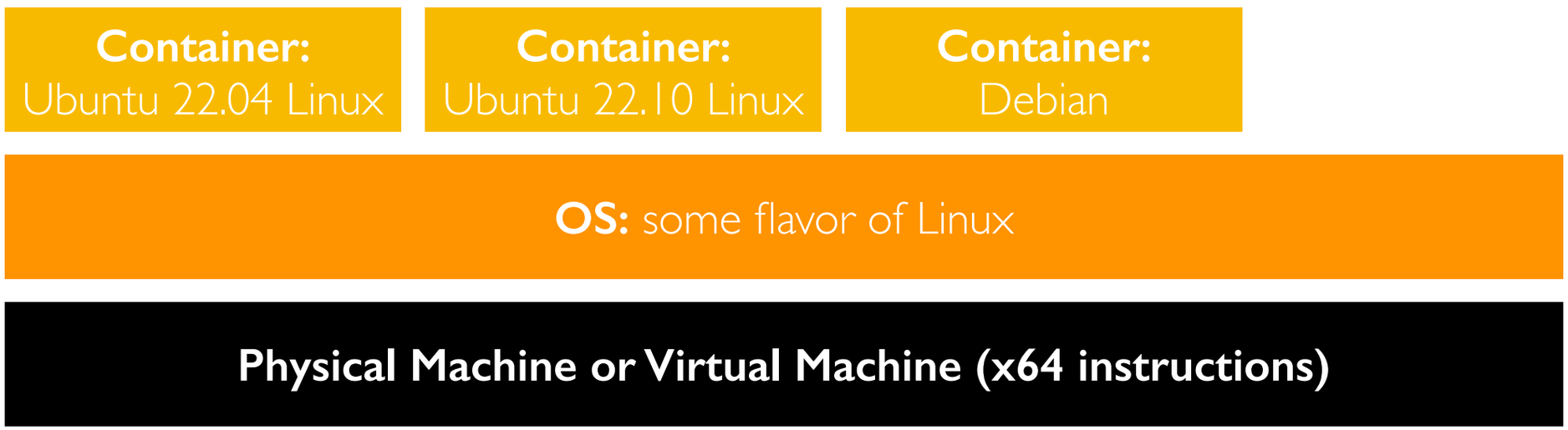
Definition: the illusion of **private** resources, provided by software

Contexts this semester

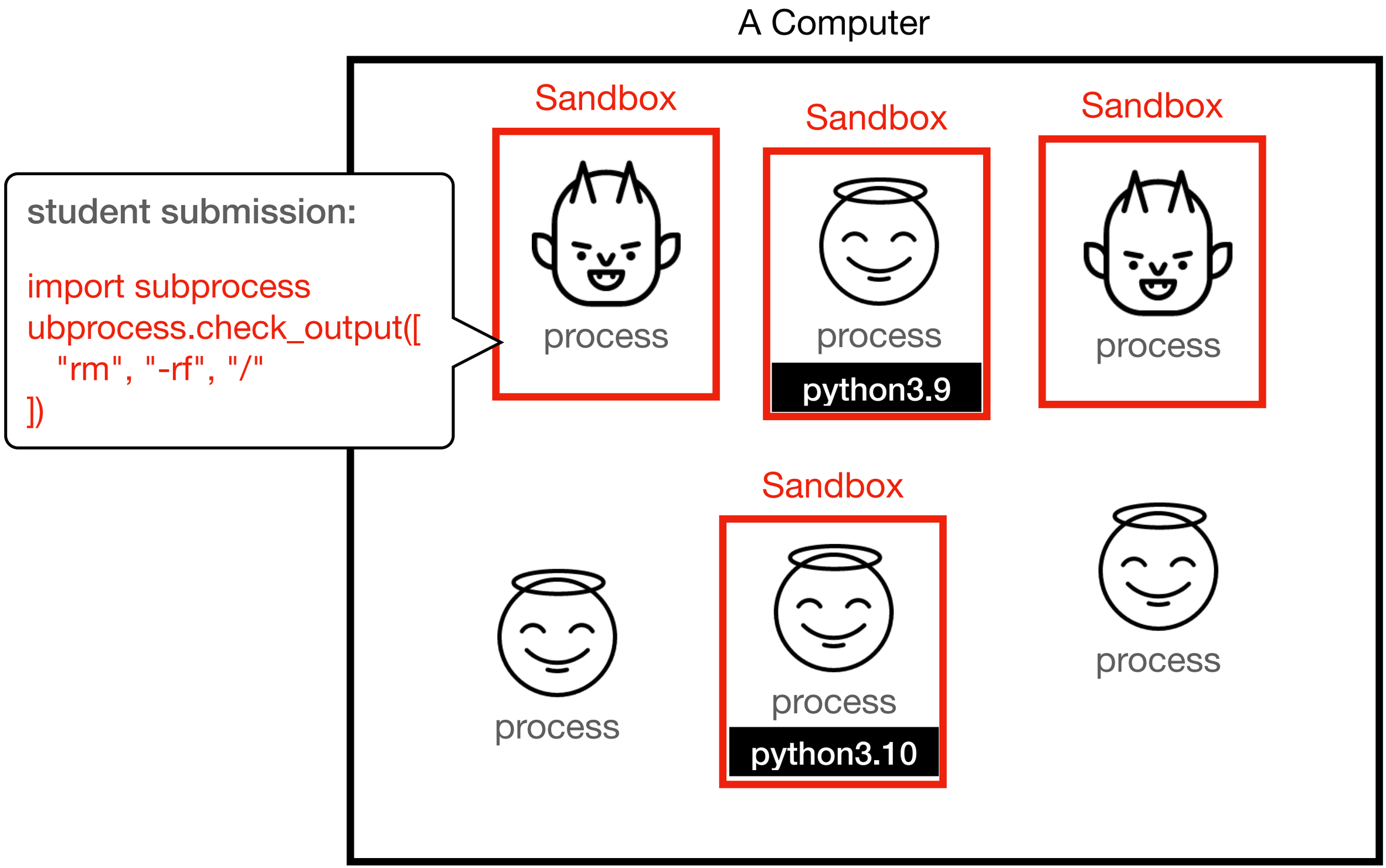
- Virtual Memory
- Virtual Machines (hardware)
- Virtual Machines (languages)
- Virtual Operating System (container)

Linux containers

- Docker makes creation easy
- The "physical" OS is shared, which is very efficient
- Programs in different containers can uses different flavors of Linux
- Cannot have a Windows container on Linux



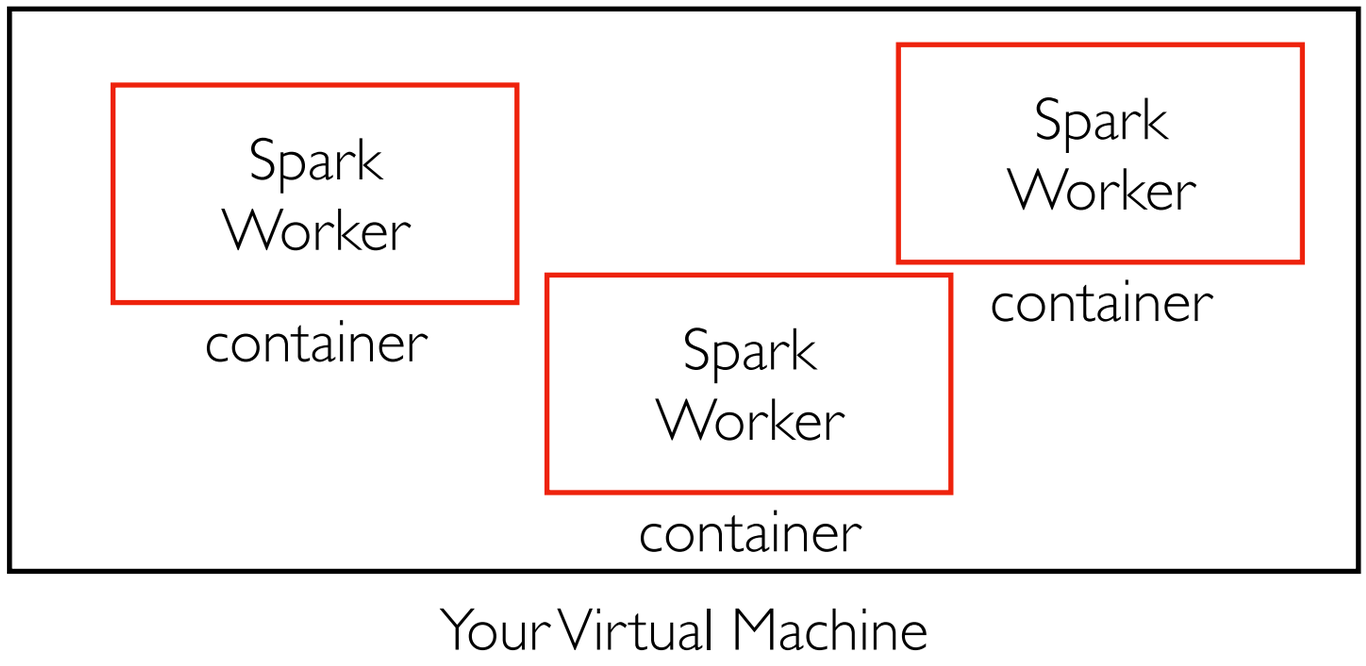
Containers and Virtual Machines are "Sandboxes"



Docker containers

Containers are a lightweight alternative to virtual machines.

You'll run Docker containers this semester to have your own "mini cluster"



Resources of the "cluster" are limited to those of a single VM, so we'll scale projects accordingly. But the techniques will apply to large clusters and datasets.

Outline

Virtualization

Images, Containers, and Dockerfiles

Ports

Docker Compose

TIP: make notes of docker commands

docker **SOME-COMMAND** arg1, arg2, ...

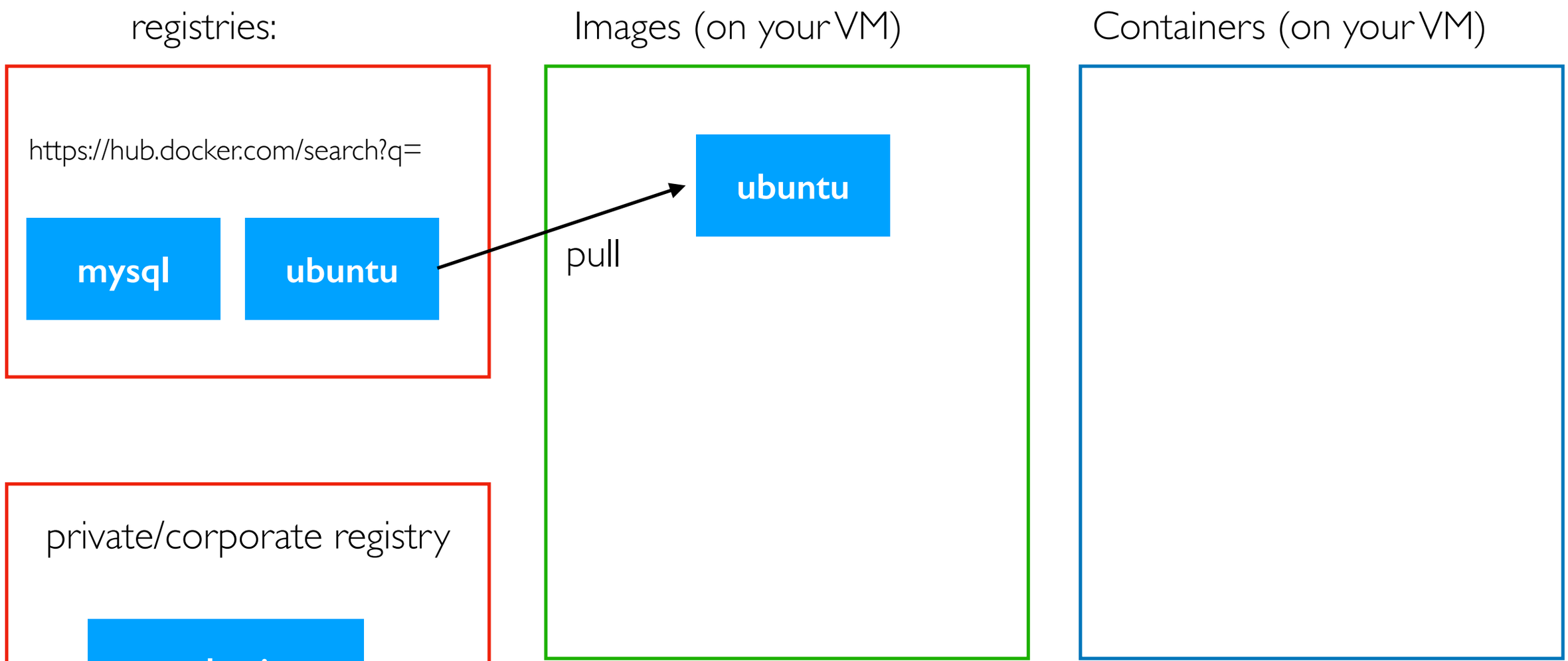
Docker Install

For Ubuntu 22.04:

```
sudo snap install docker

# allow regular user to use docker
sudo groupadd docker
sudo usermod -aG docker $USER
sudo chgrp docker /var/run/docker.sock
```

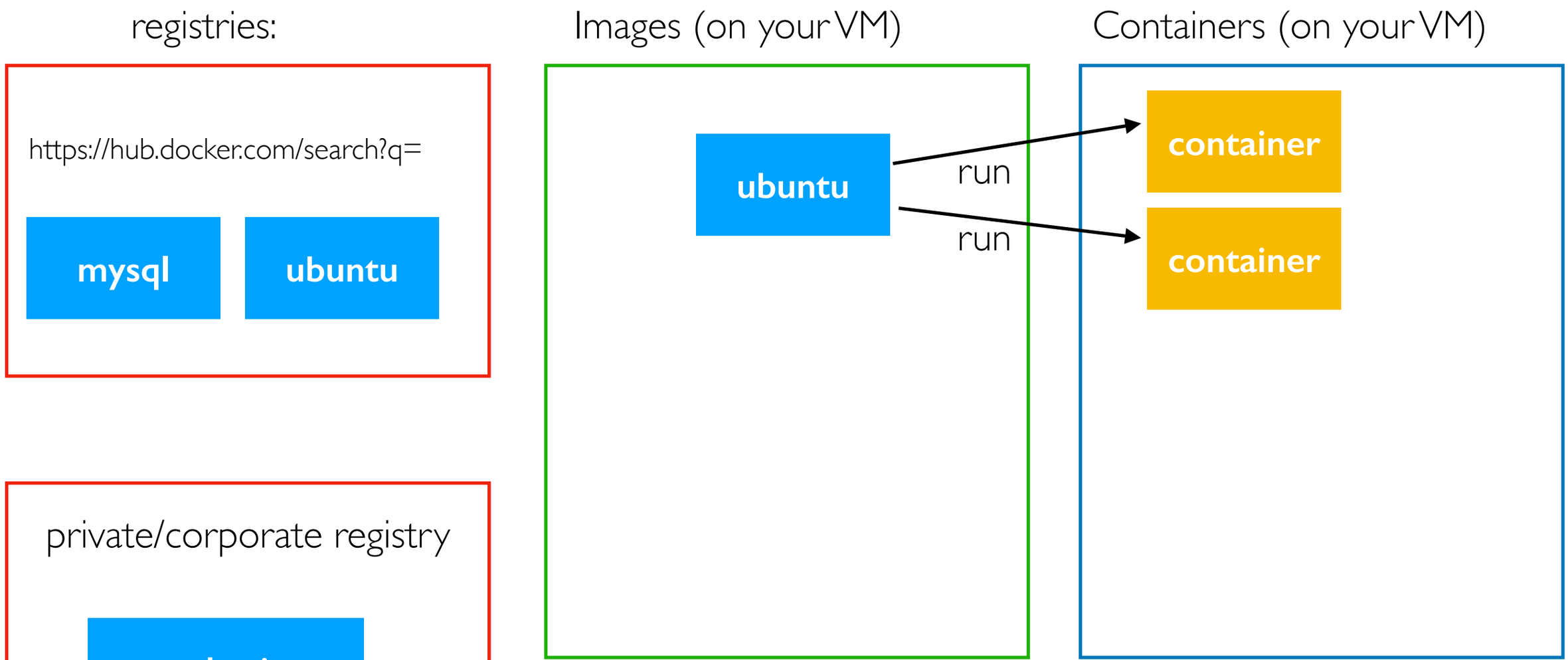
Registries, Images, Containers, and Dockerfiles



Images

- snapshot of installed software from which to create a container
- docker **pull** ubuntu:22.04

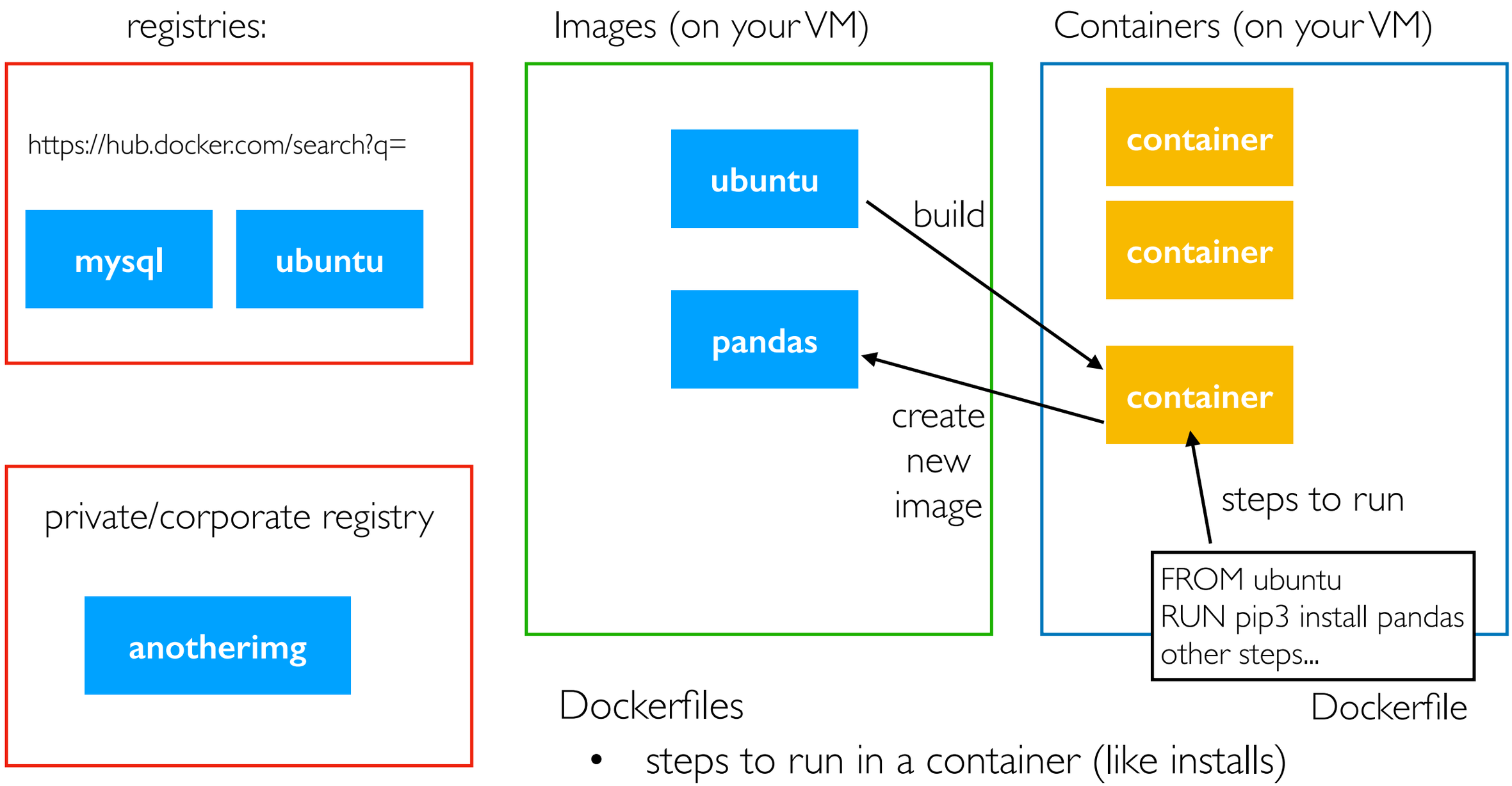
Registries, Images, Containers, and Dockerfiles



Containers

- Linux sandbox in which to run processes
- docker **run** ubuntu

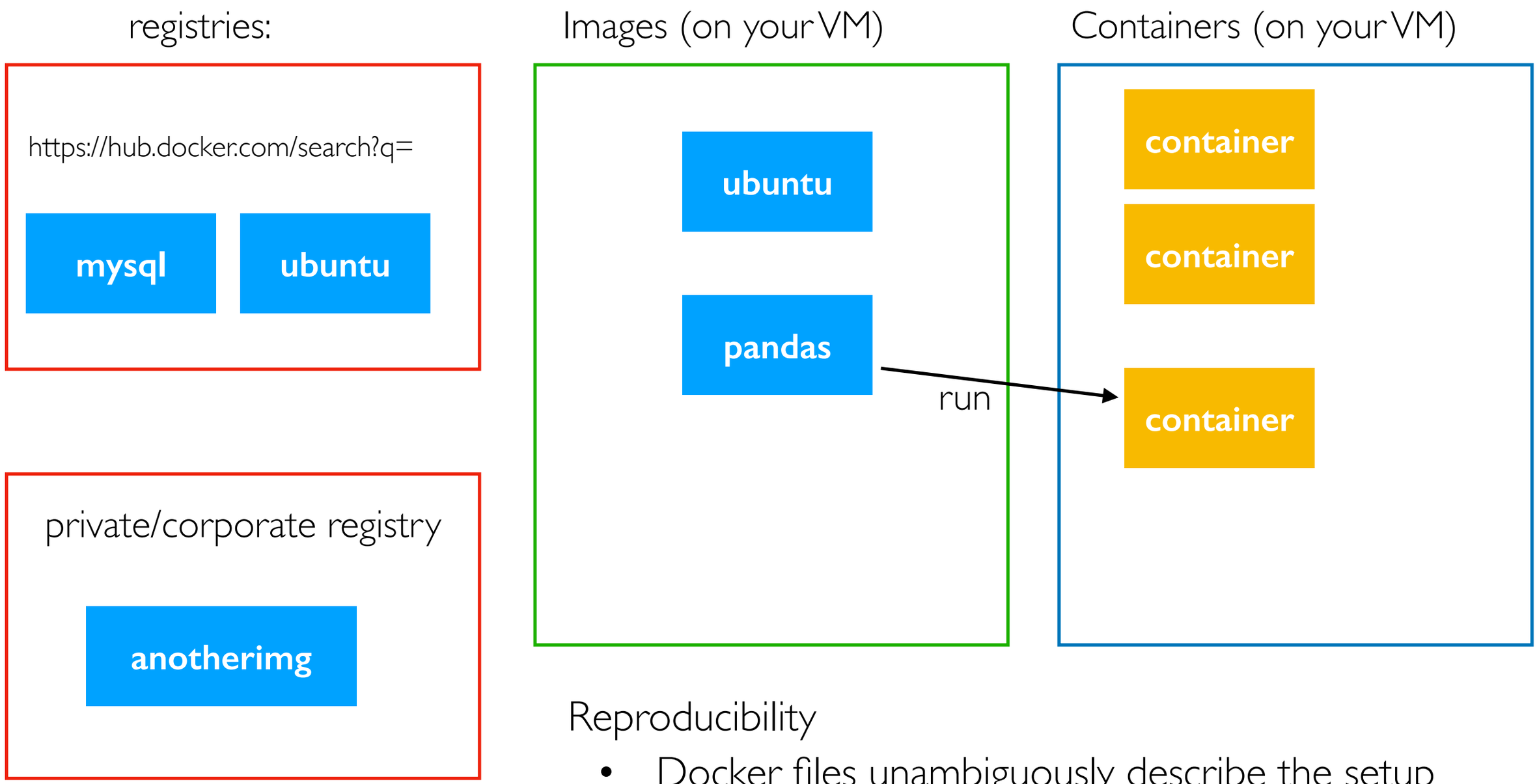
Registries, Images, Containers, and Dockerfiles



Dockerfiles

- steps to run in a container (like installs)
- creates a new image
- docker **build** myimg -t pandas

Registries, Images, Containers, and Dockerfiles



Reproducibility

- Docker files unambiguously describe the setup
- Others can get all the same version numbers

Demos...

Outline

Virtualization

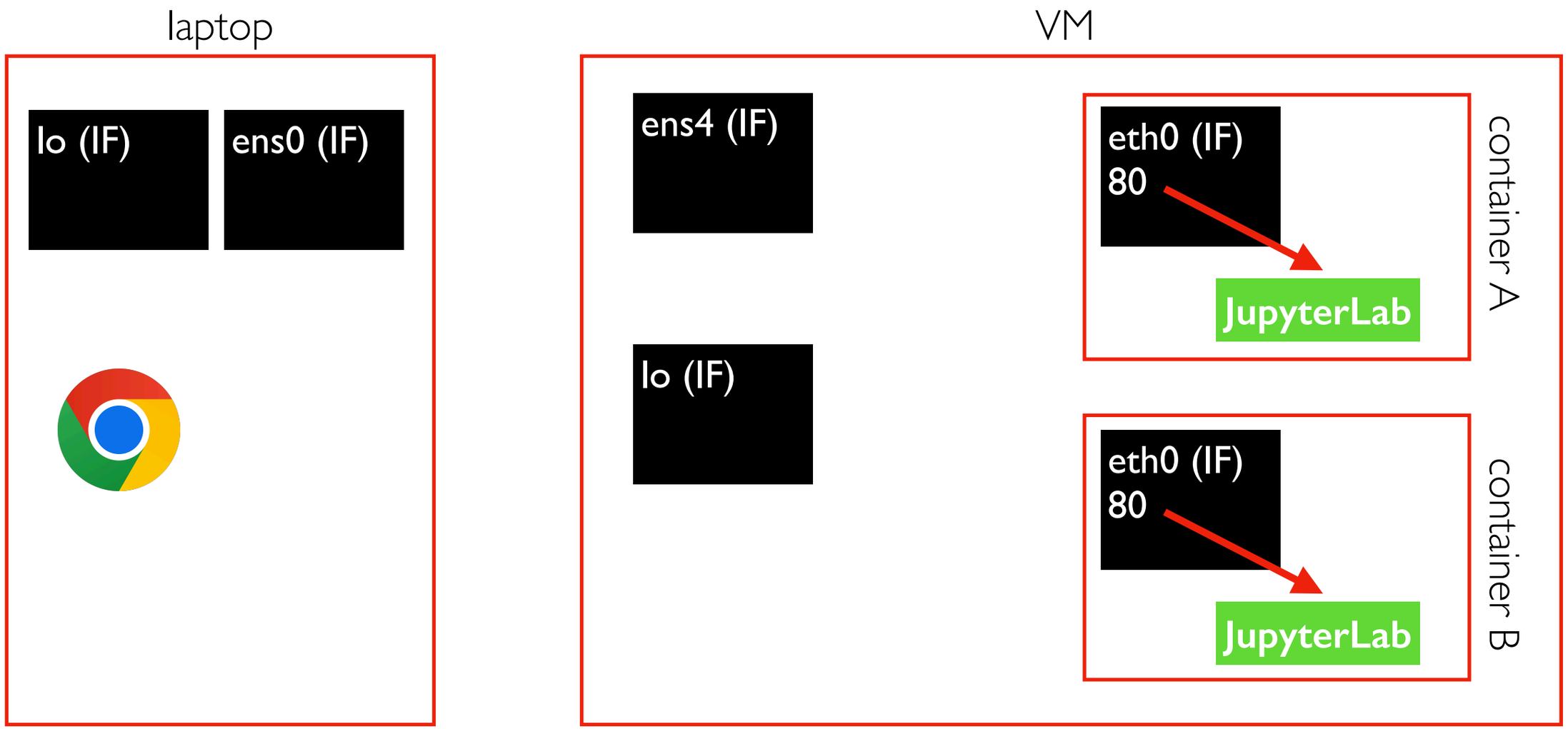
Images, Containers, and Dockerfiles

Ports

Docker Compose

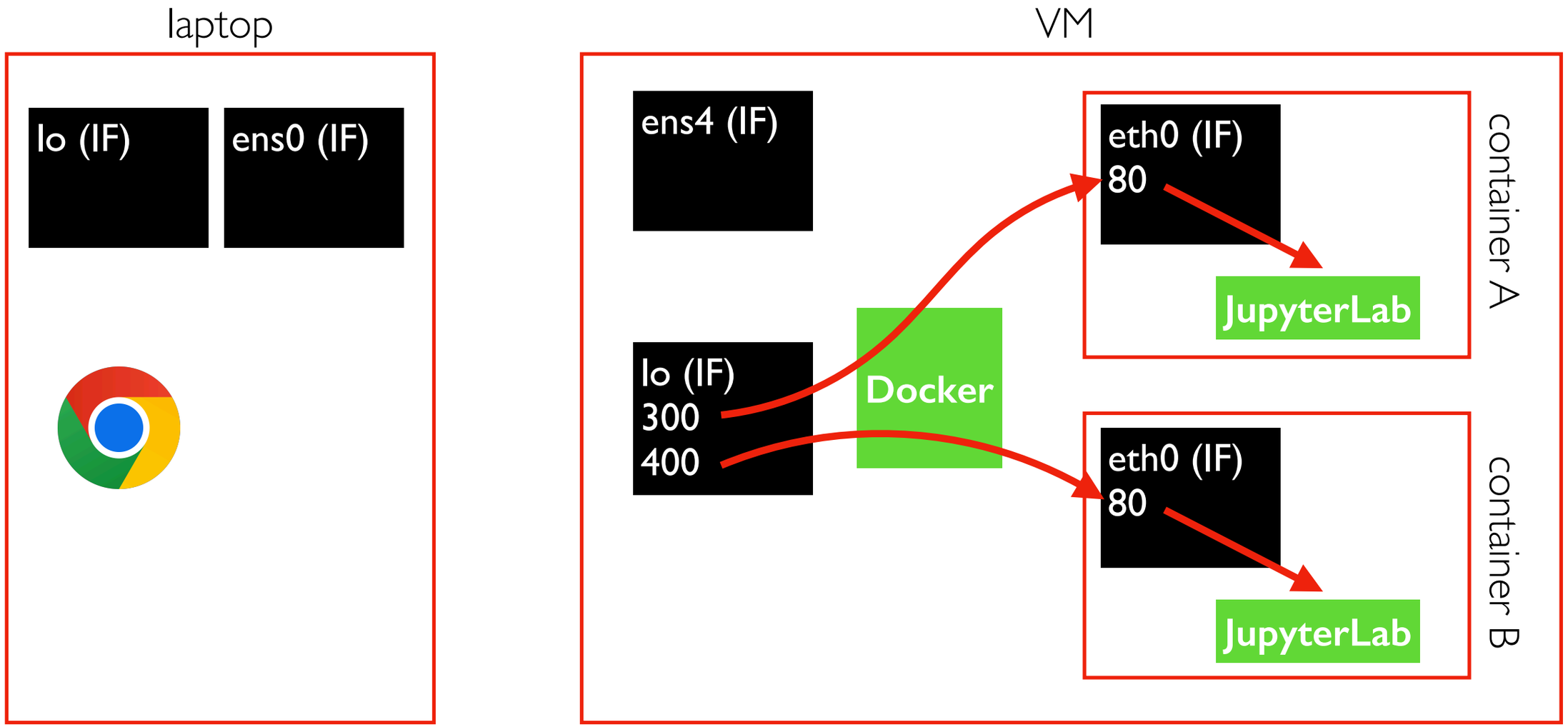
Interfaces (IF) and Ports

both containers have
a virtual port 80



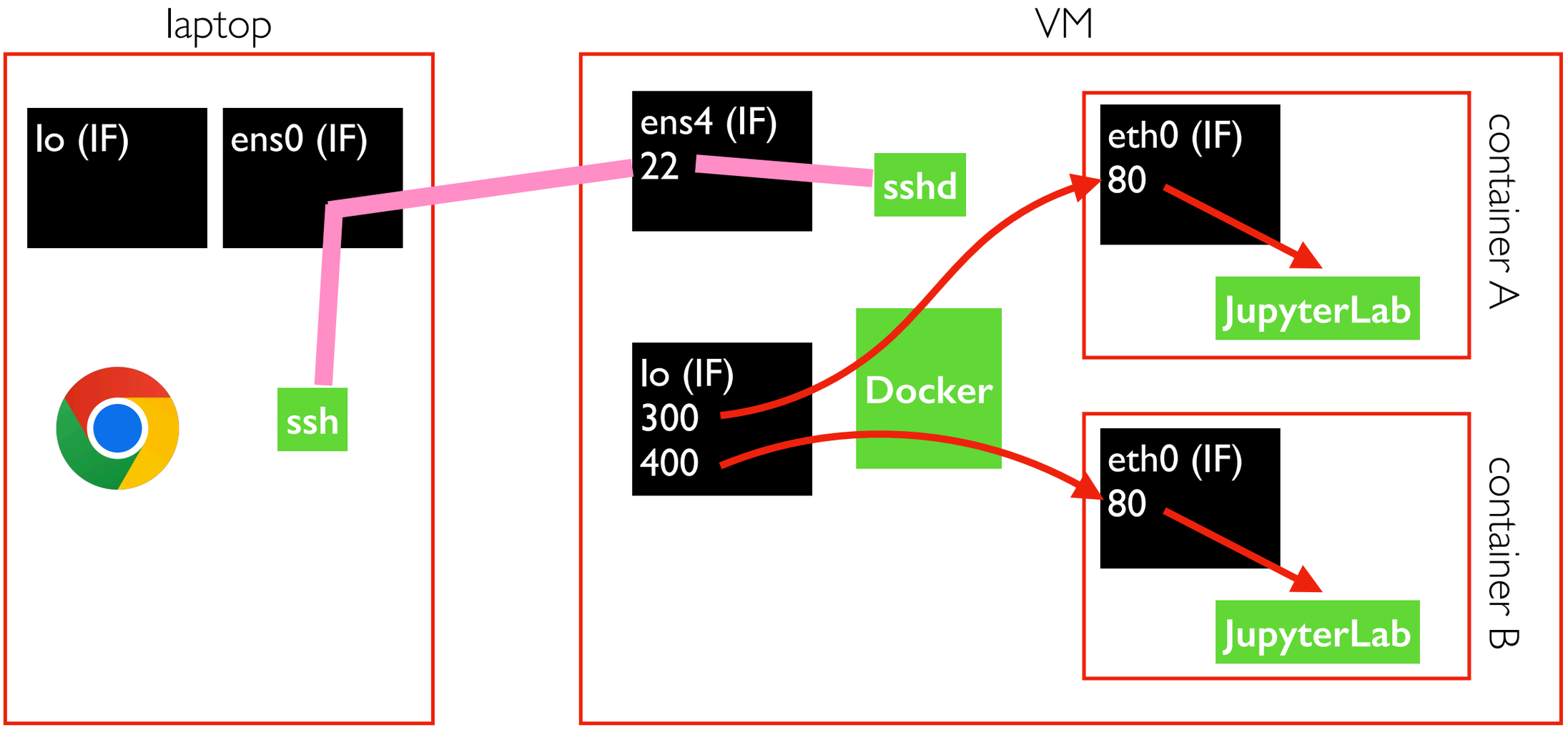

```
docker run -d myimg
docker run -d myimg
```

Interfaces (IF) and Ports




```
docker run -d -p 127.0.0.1:300:80 myimg
docker run -d -p 127.0.0.1:400:80 myimg
```

Interfaces (IF) and Ports

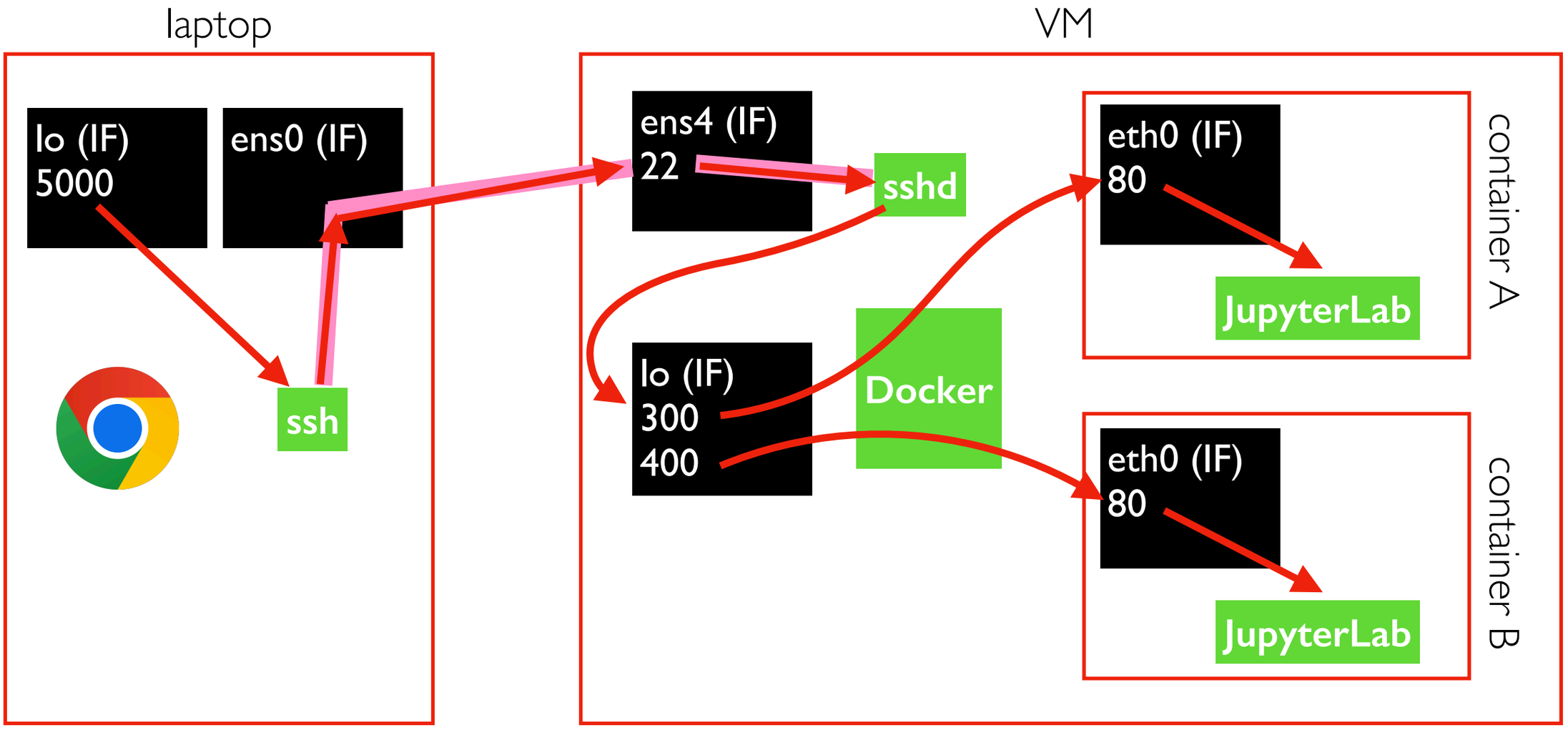


ssh USER@VM

docker run -d **-p** 127.0.0.1:300:80 myimg
docker run -d **-p** 127.0.0.1:400:80 myimg

the SSH connection can be used to send comands and/or forward network traffic

Interfaces (IF) and Ports

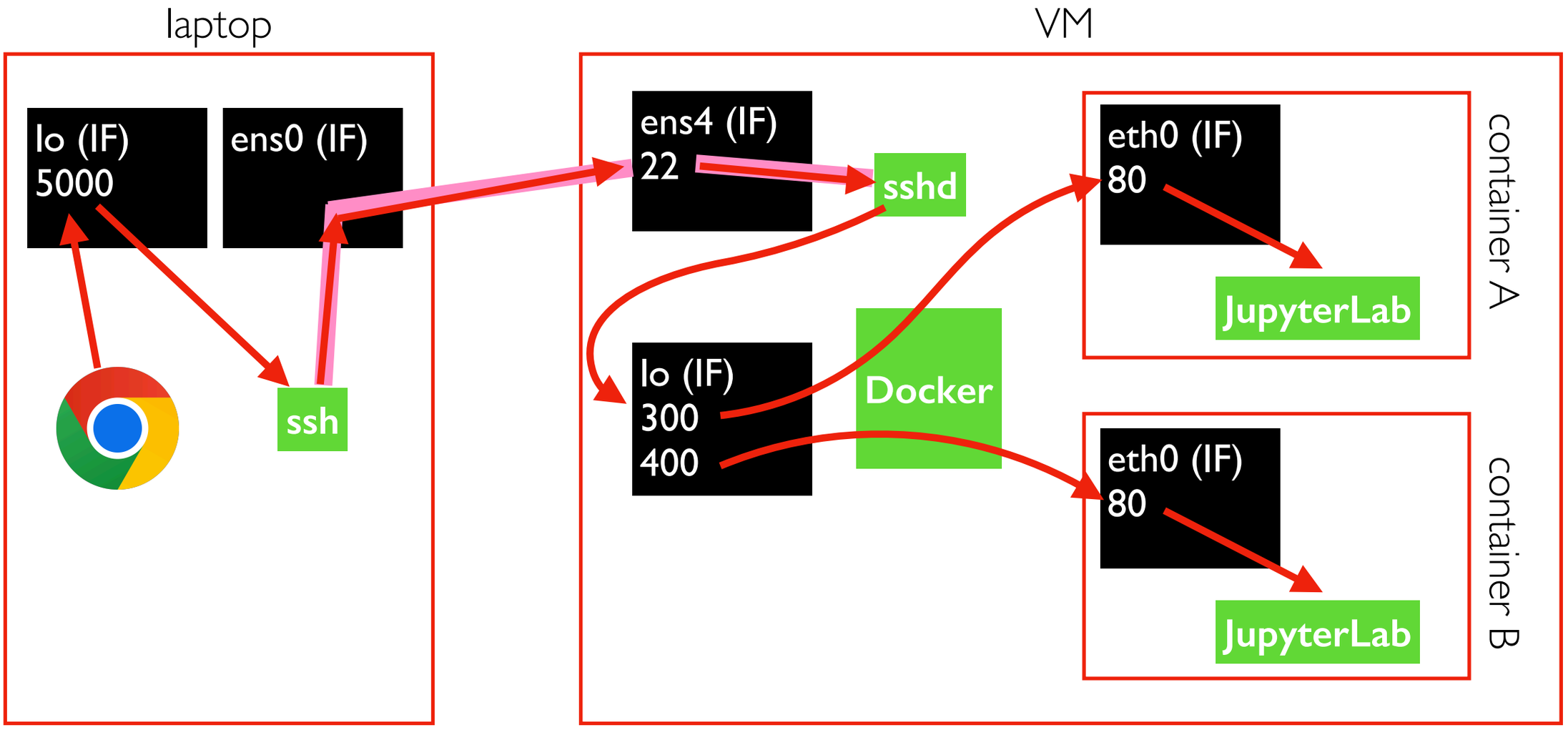


ssh USER@VM **-L** localhost:5000:localhost:300

docker run -d **-p** 127.0.0.1:300:80 myimg
docker run -d **-p** 127.0.0.1:400:80 myimg

the SSH connection can be used to send comands and/or forward network traffic

Interfaces (IF) and Ports



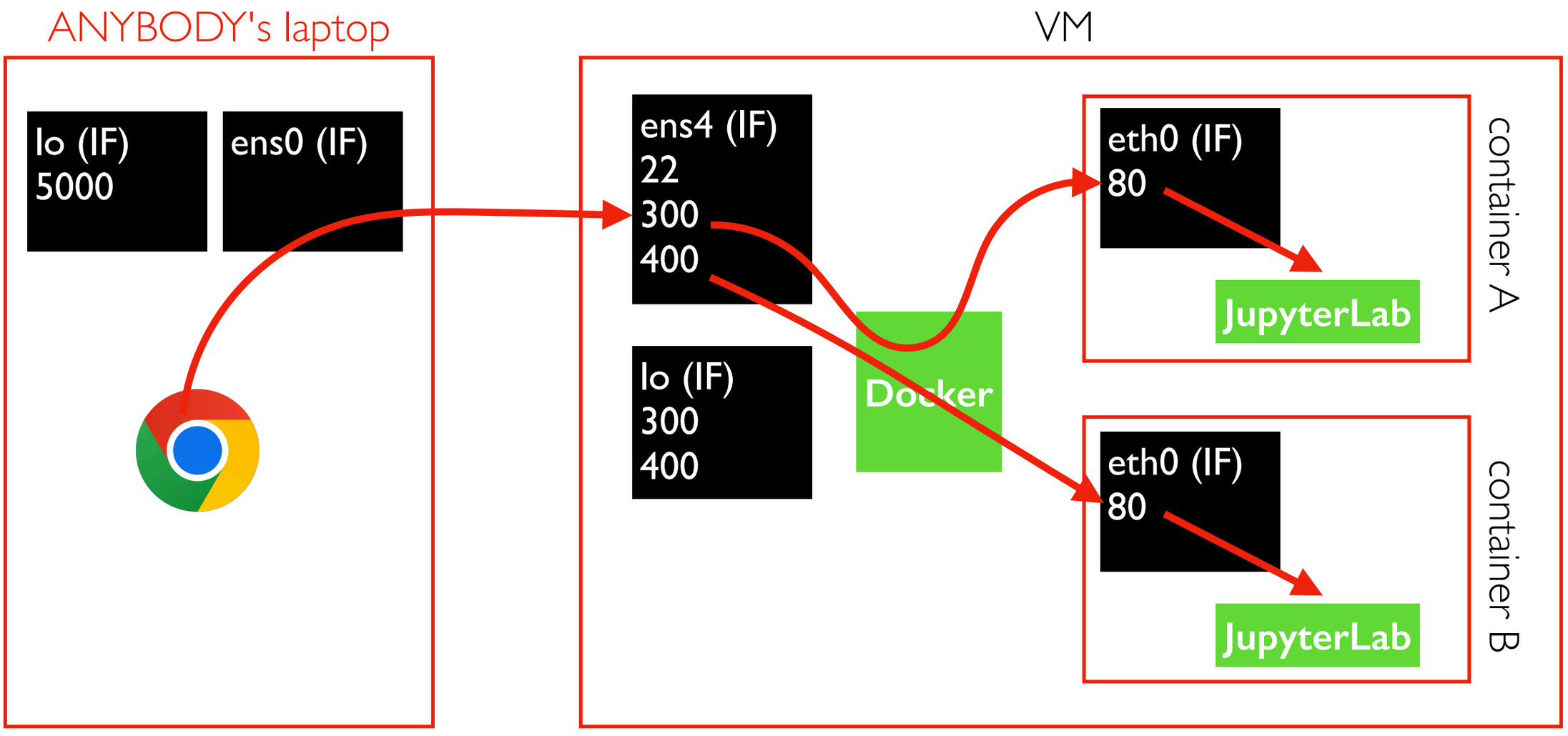
ssh USER@VM **-L** localhost:5000:localhost:300

docker run -d **-p** 127.0.0.1:300:80 myimg
docker run -d **-p** 127.0.0.1:400:80 myimg

http://localhost:5000/lab (in browser)

yay! You can connect to JupyterLab
inside a container running on your VM

Interfaces (IF) and Ports



docker run -d **-p** 300:80 myimg

docker run -d **-p** 0.0.0.0:300:80 myimg

Careful, default is to listen on all ports!

Other security:

- firewall (block port 300)
- password (in JupyterLab)

Demos...

Outline

Virtualization

Images, Containers, and Dockerfiles

Ports

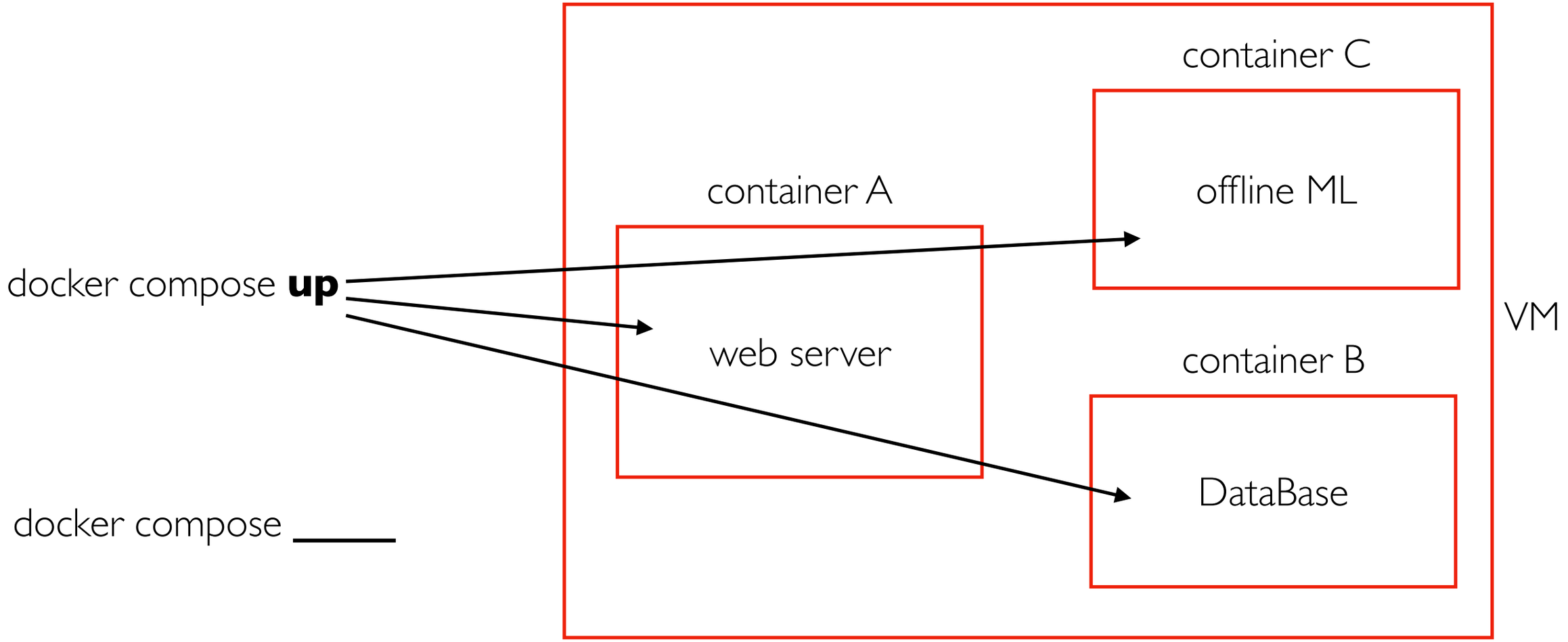
Docker Compose

Container Orchestration

Orchestration lets you deploy many cooperating containers across a cluster of Docker workers.

Kubernetes is the most well known.

Docker compose is a simpler tool that lets you deploy cooperating containers to a single worker.



Demos...