Docker for DS

W W H

What Why How

Why What How

Why

Why

~~Why~~ Problems Docker Solves

~~Why~~ Problems Docker Solves

- Reproducibility.

~~Why~~ Problems Docker Solves

- Reproducibility.
- Scalable computational backend.

~~Why~~ Problems Docker Solves

- Reproducibility.
- Scalable computational backend.
- Persistent computational environment.

~~Why~~ Problems Docker Solves

- Reproducibility.
- Scalable computational backend.
- Persistent computational environment.
- Shared dependency management.

~~Why~~ Problems Docker Solves

- Reproducibility.
- Scalable computational backend.
- Persistent computational environment.
- Shared dependency management.
- Lack of appropriate permissions on a server.

Reproducibility

Reproducibility

- The container used to run the code for the model can be restarted in the exact state it was left, and code re-run.

Reproducibility

- The container used to run the code for the model can be restarted in the exact state it was left, and code re-run.
- This complements the Makefile, by ensuring that the build process does not fail due to dependencies.

Scalable computational backend

Scalable computational backend

- Running code our macbooks works well, and is very flexible, but is limited by the hardware on our machines. Docker has to potential to scale to the size of the machine it is run on.

Persistent computational environment

Persistent computational environment

- Often models will train overnight, or code otherwise needs to keep running (displaying results in a dashboard, for example).

Persistent computational environment

- Often models will train overnight, or code otherwise needs to keep running (displaying results in a dashboard, for example).

- Relying on the Caffeine app doesn't always work (laptops sometimes just reset, we want to take them home, etc).

Shared dependency management

Shared dependency management

- We each set up our laptops ourselves, often requiring substantial work (see https://gist.github.com/andyreagan/32c737fd315e70ccb3b4a4387a466c85) and doing so inconsistently.

Shared dependency management

- We each set up our laptops ourselves, often requiring substantial work (see https://gist.github.com/andyreagan/32c737fd315e70ccb3b4a4387a466c85) and doing so inconsistently.
- If we can share installations of common tools, that would reduce the workload for individual data scientists or maintaining personal setups.

Lack of permissions

Lack of permissions

- Happens both on some servers, and on managed macs!

Lack of permissions

- Happens both on some servers, and on managed macs!

- This can be fixed, but is nonetheless a real life hurdle.

Problems Docker Creates

Problems Docker Creates

Problems Docker Creates

- Performance overhead?

Problems Docker Creates

- Performance overhead?
- Maintenance of docker environment itself.

Performance overhead

- This is a bit unknown (and yes, on both sides). Docker running inside a VM locally, or on a server, has access to share memory and compute (I've seen all N cores get used locally training a model).

Maintenance of docker environment itself

- Will this be less work than managing an individual setup? It is surely at least a bit harder to set up Docker (using the RUN commands, etc) than a local environment.

Alternatives (when *not* to use Docker)

- Python virtualenv.
- Julia user home package installation.
- Packrat?

What

What is Docker?

magic

What is Docker?

~~magic~~

What is Docker?

What is Docker?

plumbing!

What is Docker?

What is Docker?

- RedHat <-> linux as Docker <-> runC

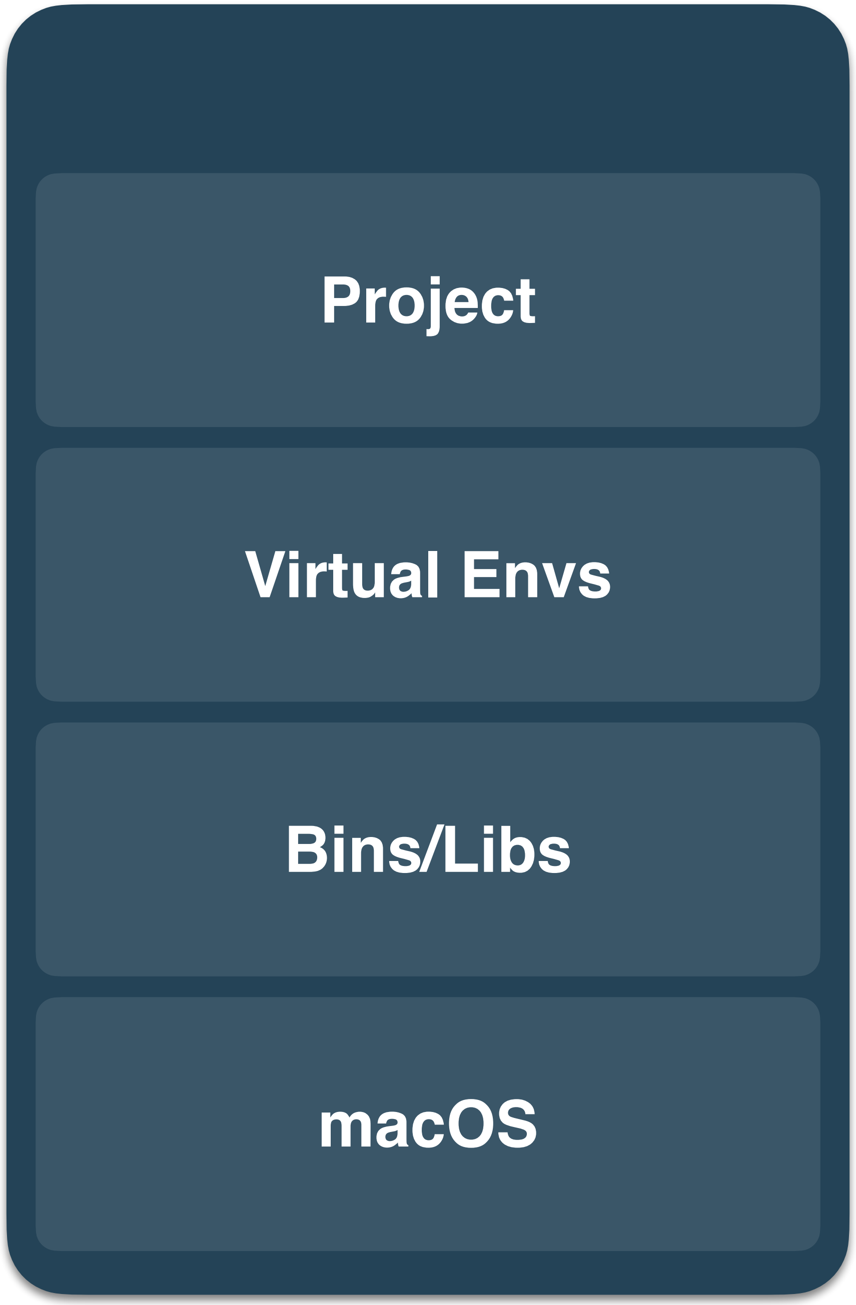
What is Docker?

- RedHat <-> linux as   Docker <-> runC
- Underlying tech ~~LXC~~ runC/libcontainer.

What is Docker?

- RedHat <-> linux as Docker <-> runC
- Underlying tech ~~LXC~~ runC/libcontainer.
- ~~AuFS~~ OverlayFS file system driver

Your laptop



Project

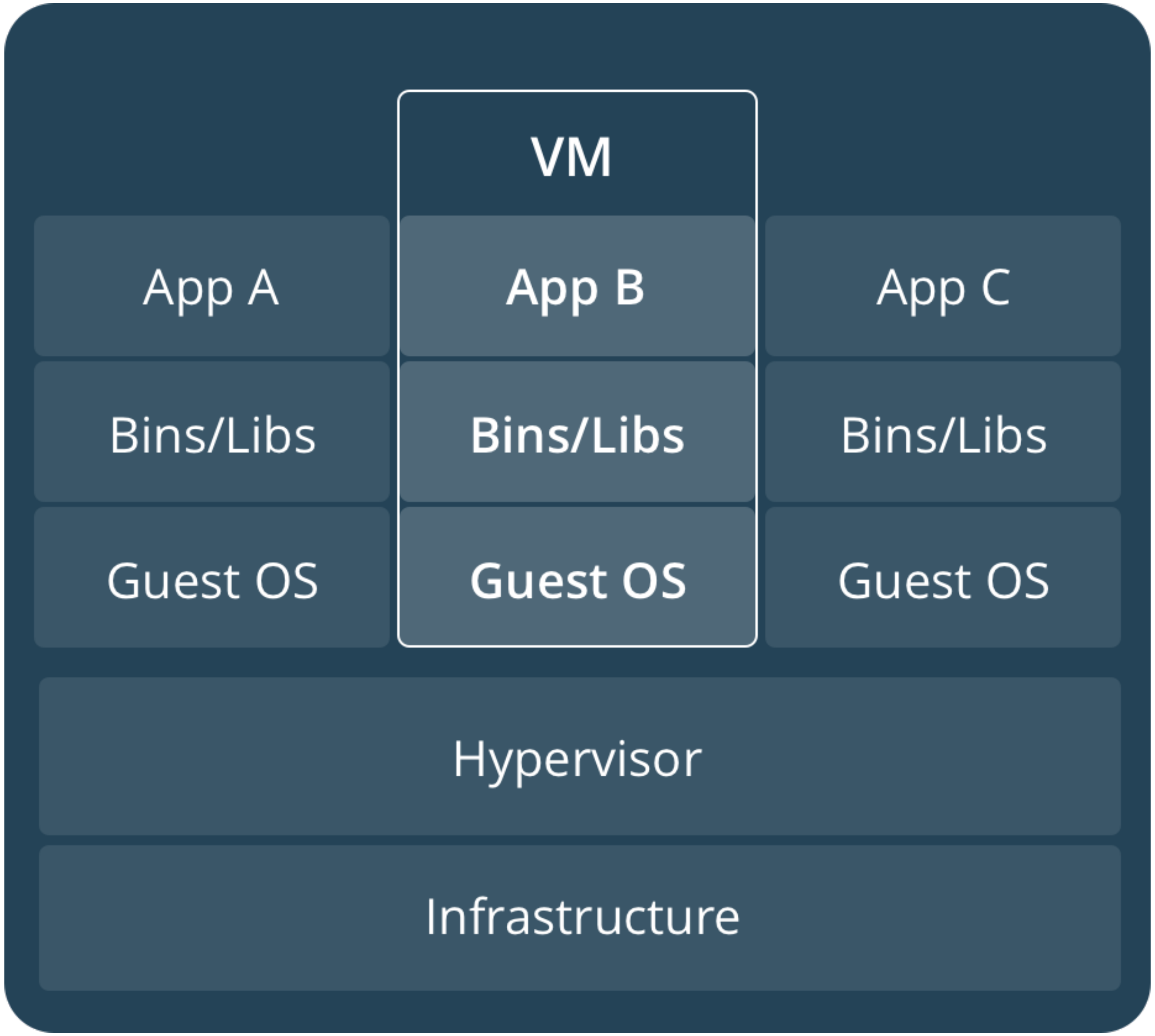
Virtual Envs

Bins/Libs

macOS

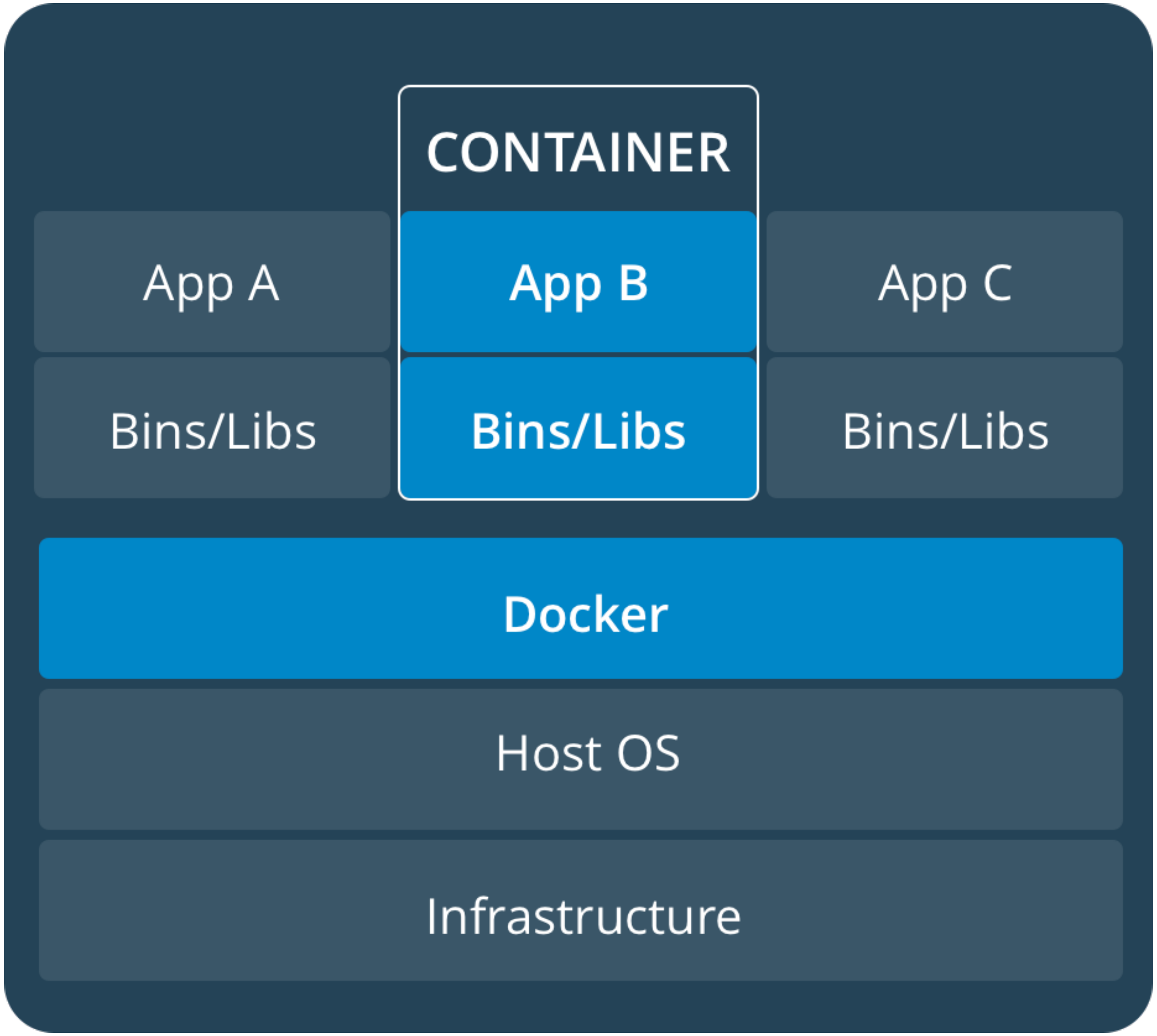
Your laptop

Your laptop

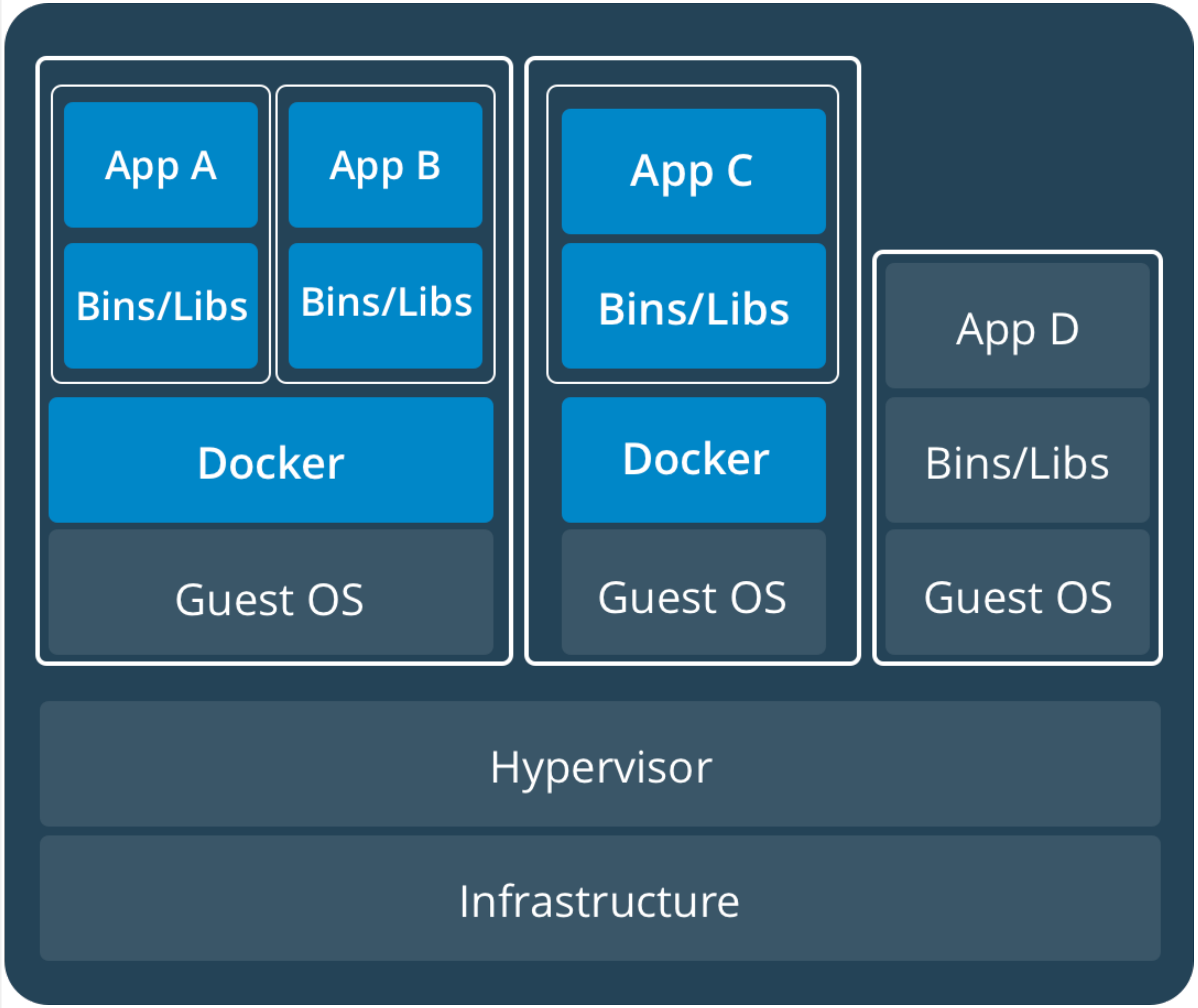


VM
App A
App B
App C
Bins/Libs
Bins/Libs
Bins/Libs
Guest OS
Guest OS
Guest OS
Hypervisor
Infrastructure

with containers

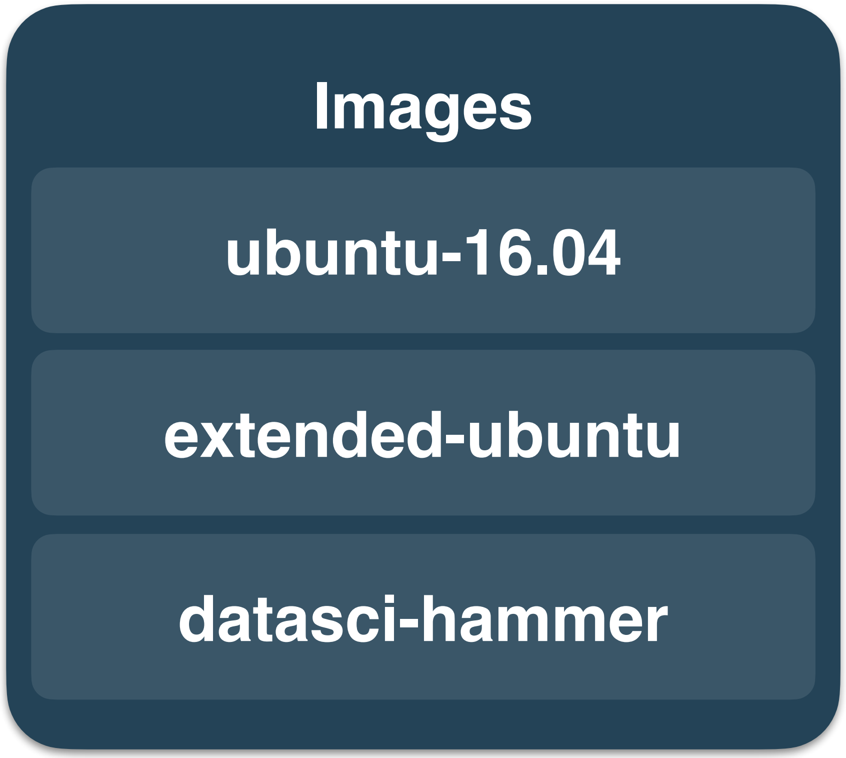


on your laptop



Two key concepts

Two key concepts



Images

ubuntu-16.04

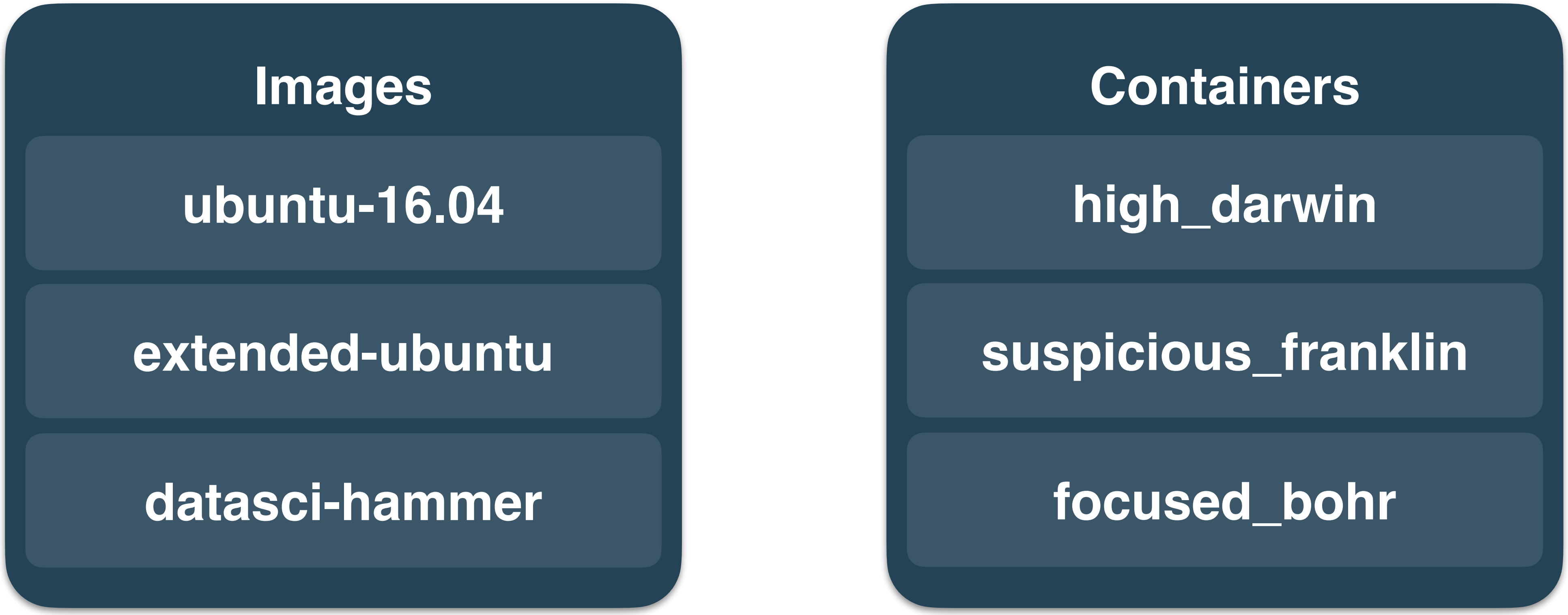
extended-ubuntu

datasci-hammer

Two key concepts

Images	Containers
ubuntu-16.04	sick_einstein
extended-ubuntu	reverent_fermat
datasci-hammer	naughty_heisenberg

Two key concepts



Images

- ubuntu-16.04
- extended-ubuntu
- datasci-hammer

Containers

- high_darwin
- suspicious_franklin
- focused_bohr

docker images

Two key concepts

Images

- ubuntu-16.04
- extended-ubuntu
- datasci-hammer

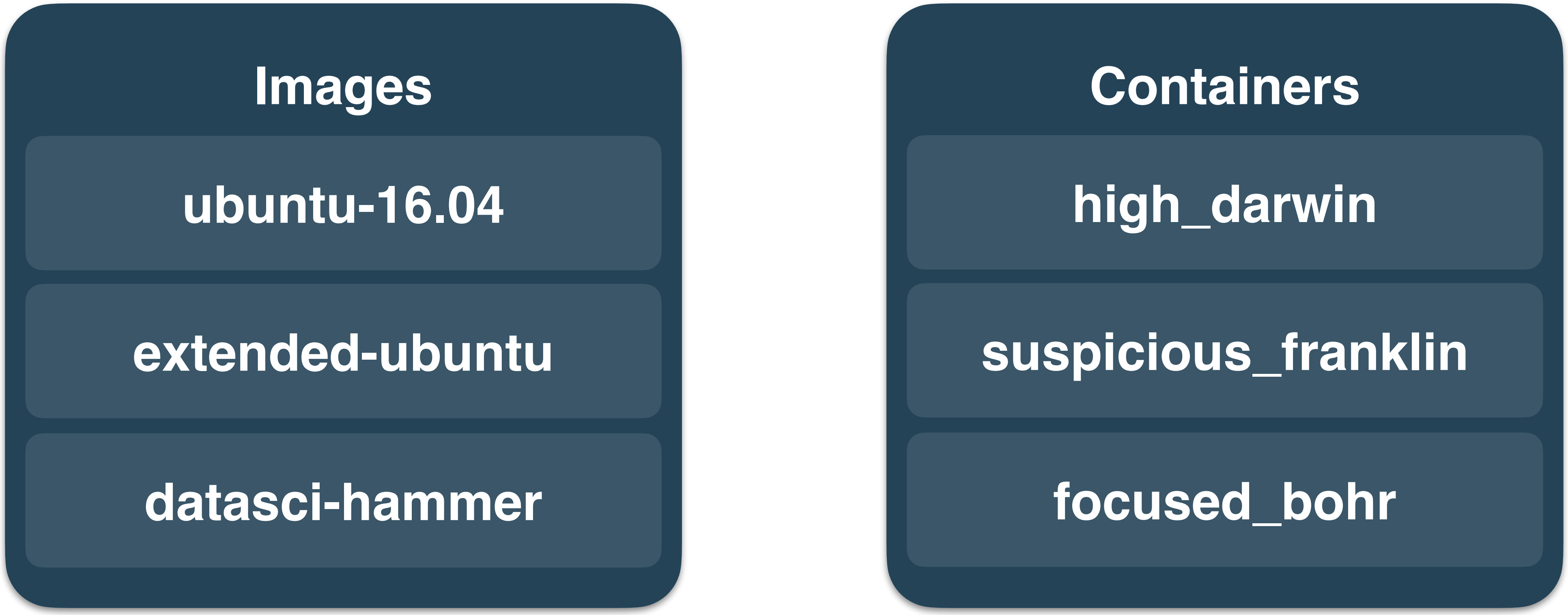
Containers

- high_darwin
- suspicious_franklin
- focused_bohr

docker images

docker ps [-a]

Two key concepts



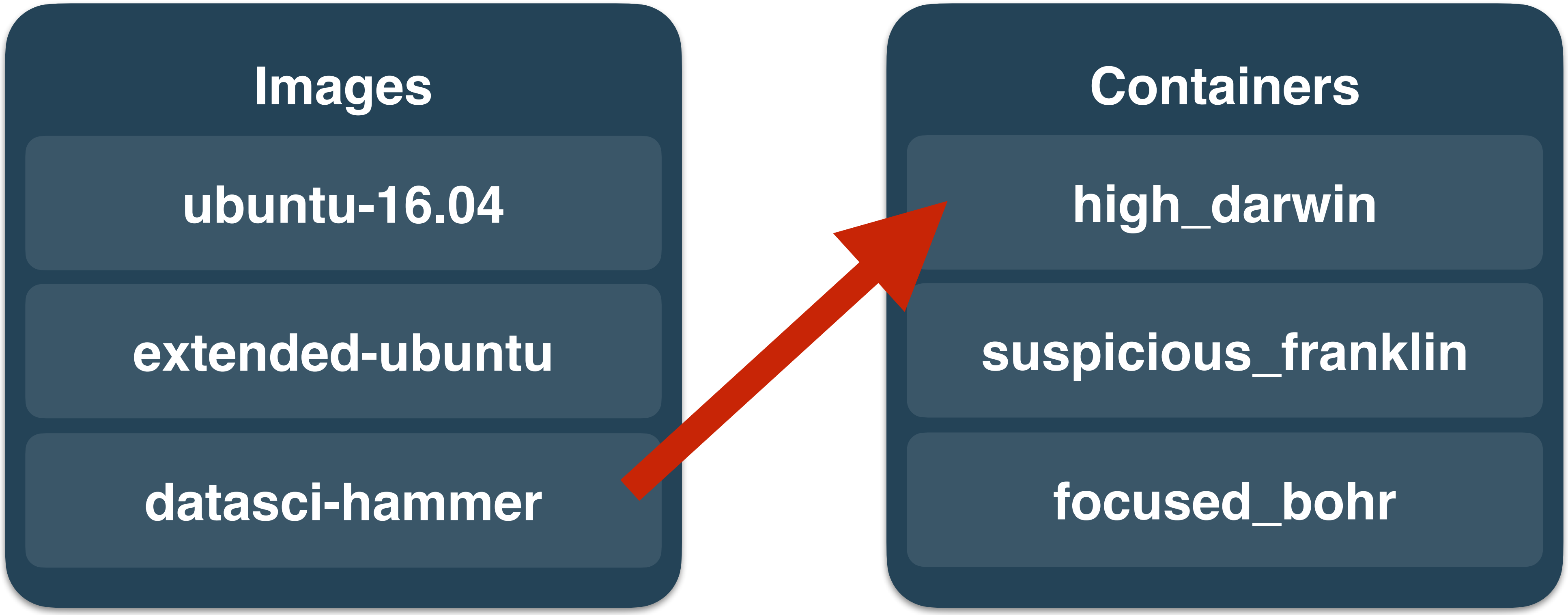
Images

- ubuntu-16.04
- extended-ubuntu
- datasci-hammer

Containers

- high_darwin
- suspicious_franklin
- focused_bohr

Two key concepts



Images

ubuntu-16.04

extended-ubuntu

datasci-hammer

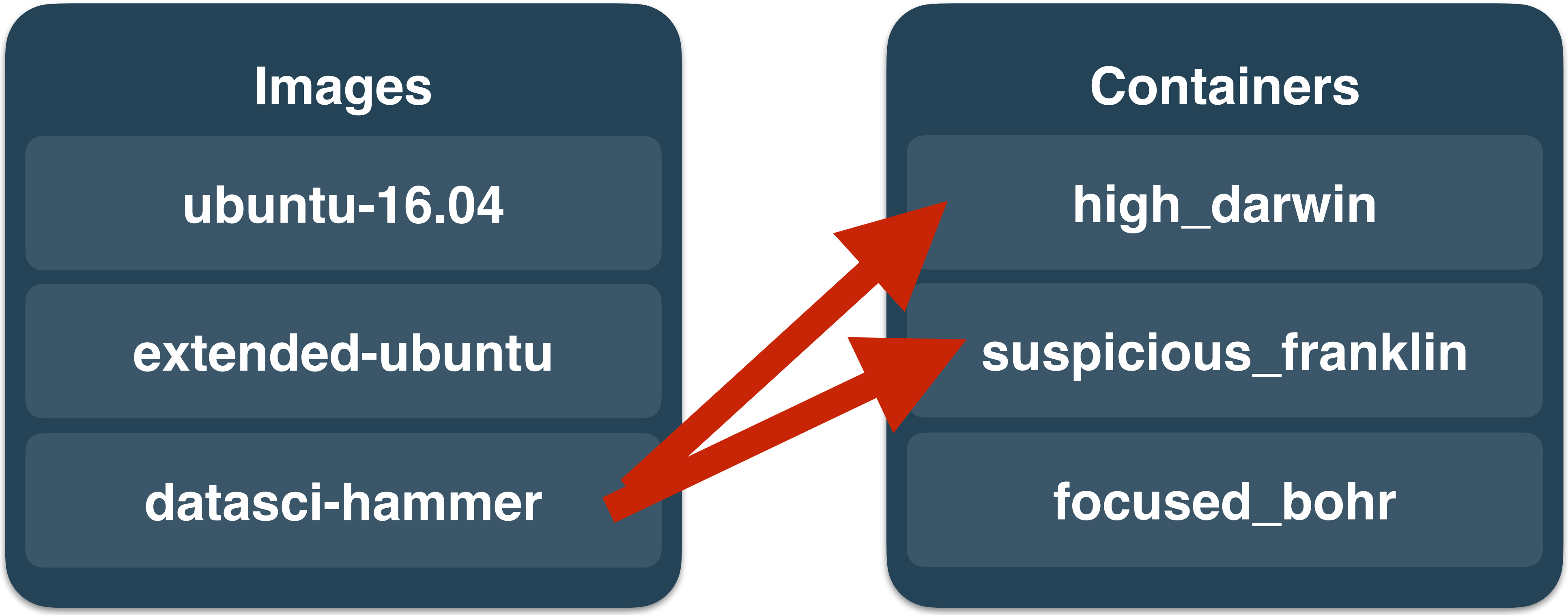
Containers

high_darwin

suspicious_franklin

focused_bohr

Two key concepts



Images

ubuntu-16.04

extended-ubuntu

datasci-hammer

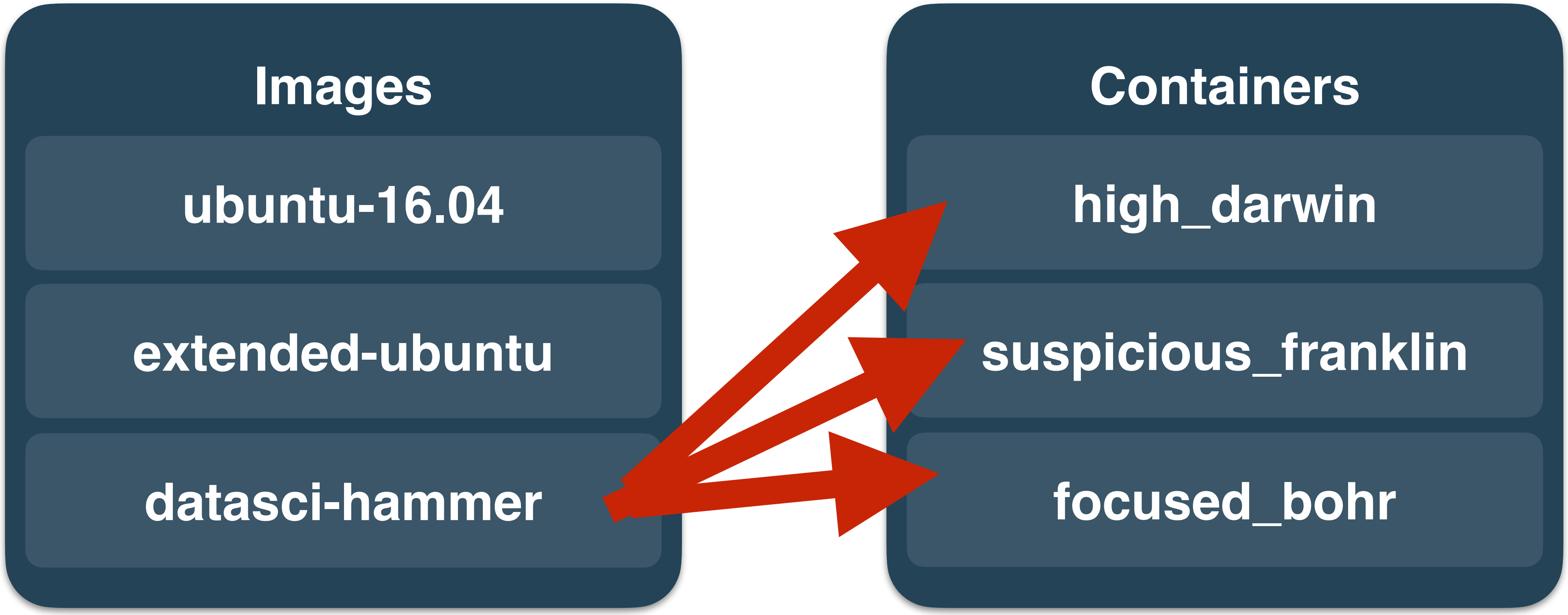
Containers

high_darwin

suspicious_franklin

focused_bohr

Two key concepts



Images

ubuntu-16.04

extended-ubuntu

datasci-hammer

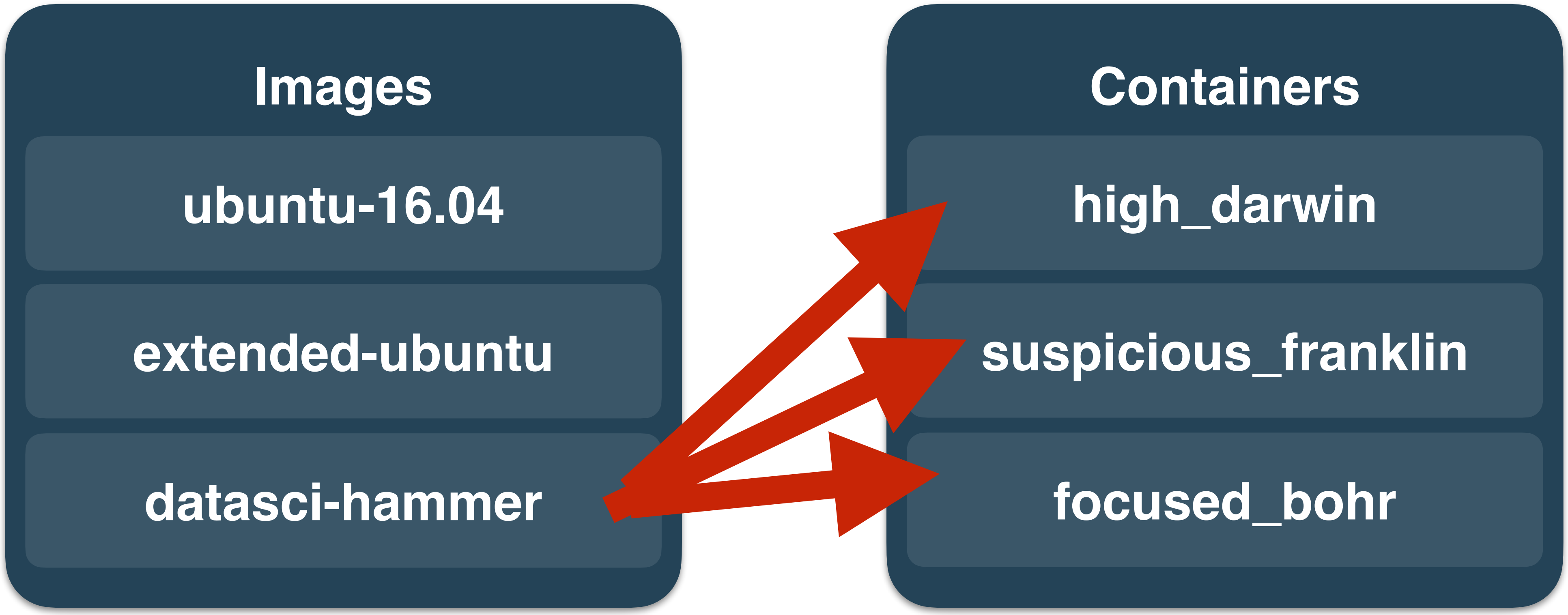
Containers

high_darwin

suspicious_franklin

focused_bohr

Two key concepts



Images

ubuntu-16.04

extended-ubuntu

datasci-hammer

Containers

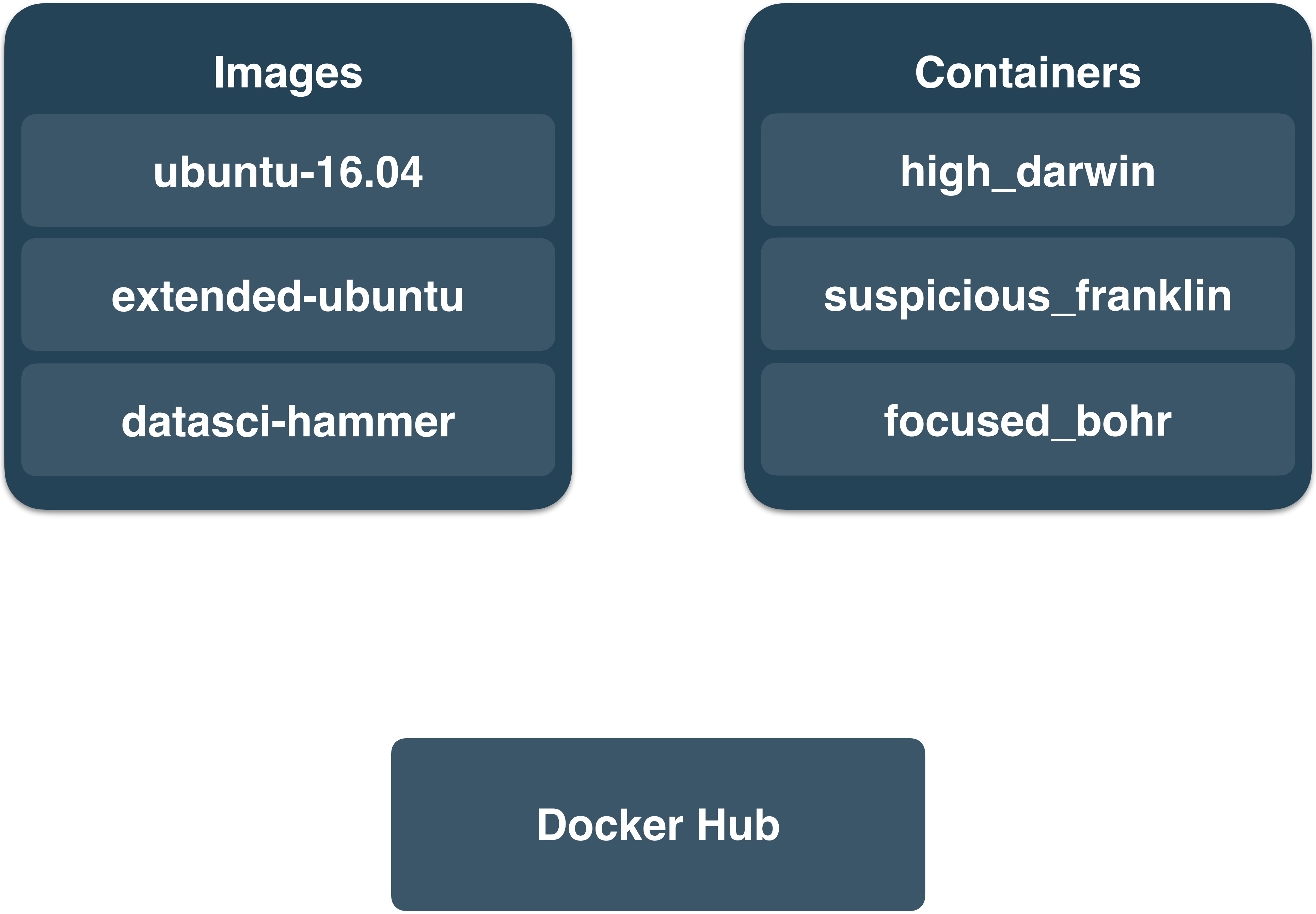
high_darwin

suspicious_franklin

focused_bohr

```
docker run datasci-hammer
```

Two key concepts



Images

ubuntu-16.04

extended-ubuntu

datasci-hammer

Containers

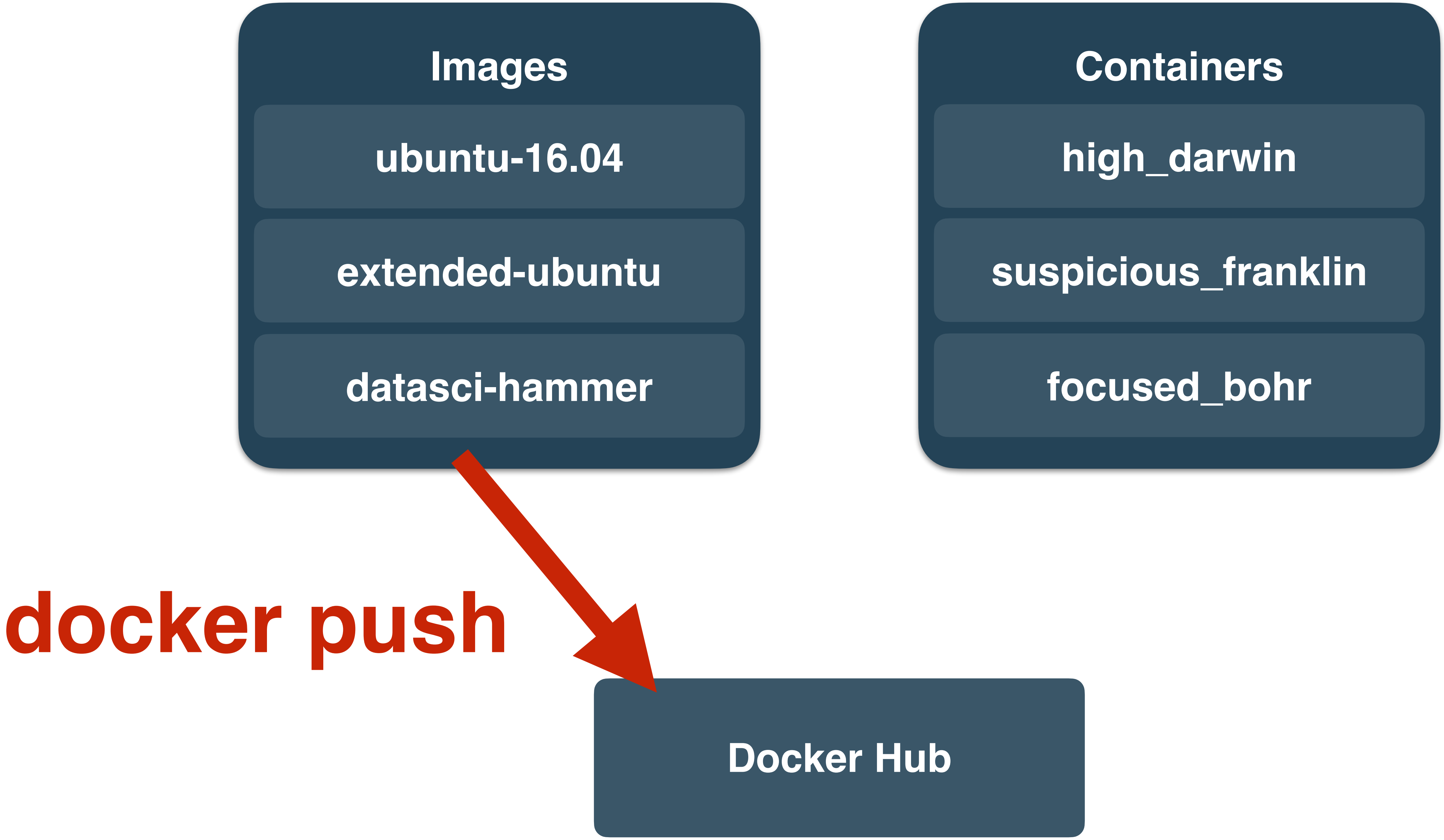
high_darwin

suspicious_franklin

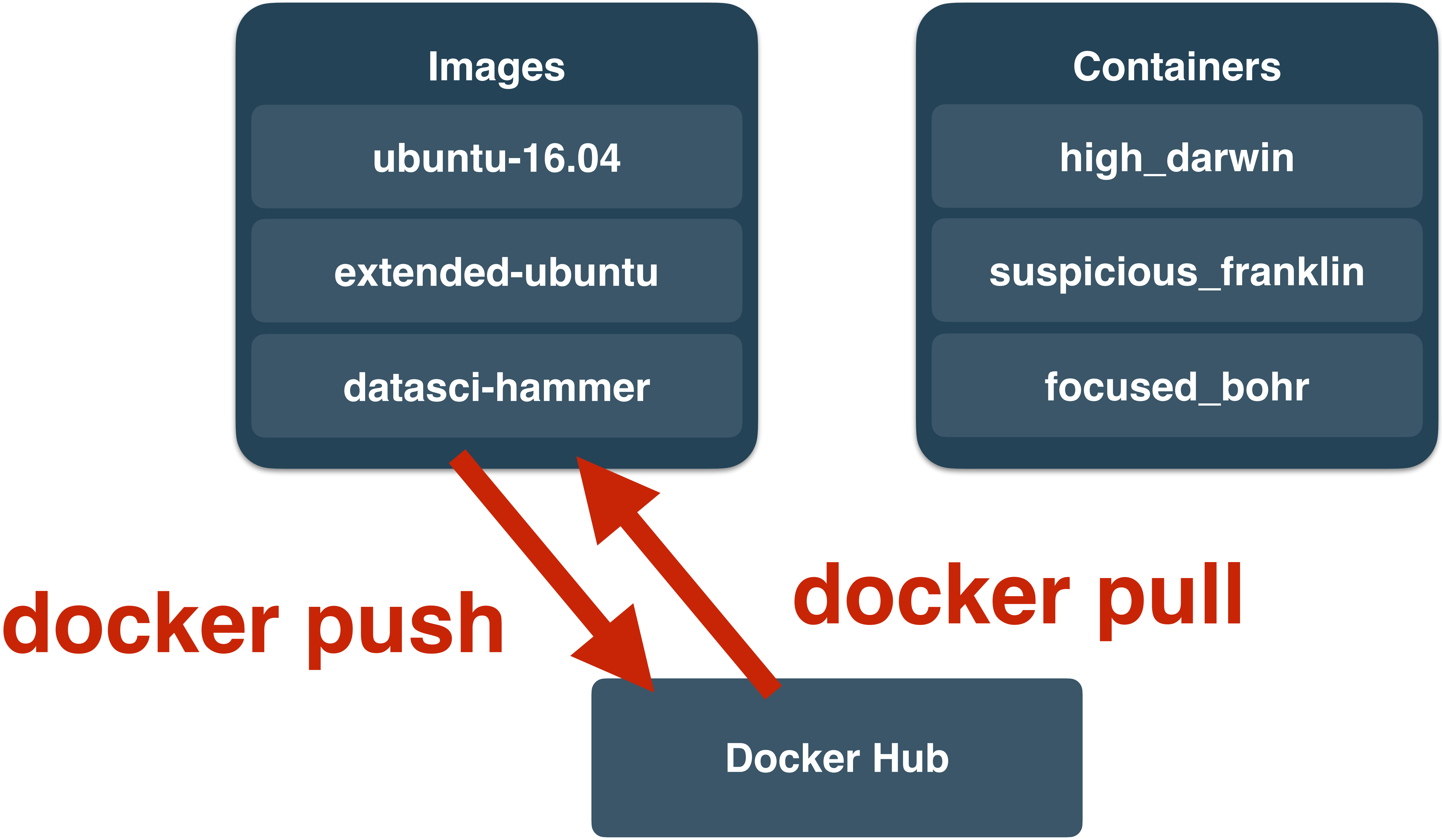
focused_bohr

Docker Hub

Two key concepts



Two key concepts



Images

- ubuntu-16.04
- extended-ubuntu
- datasci-hammer

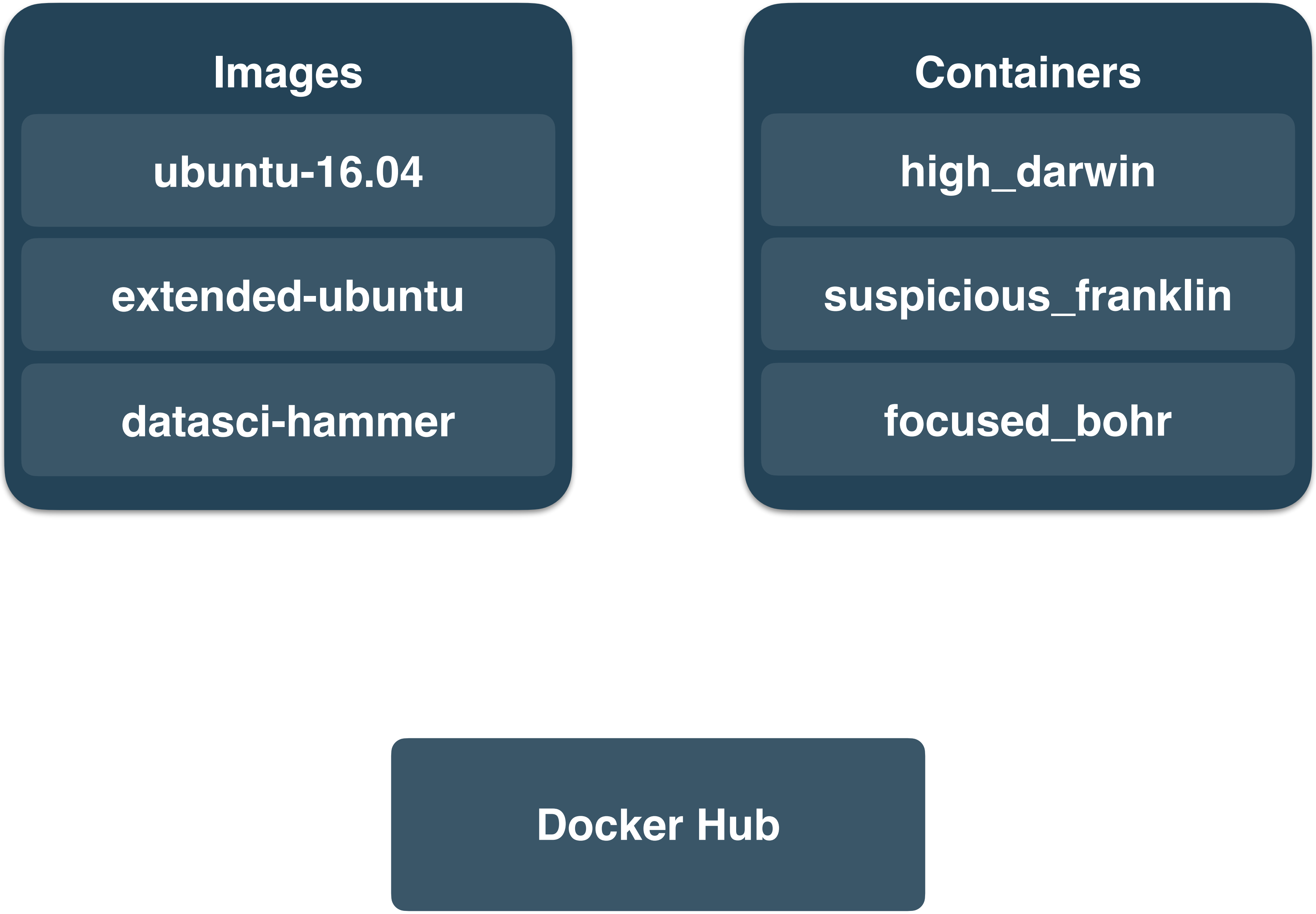
Containers

- high_darwin
- suspicious_franklin
- focused_bohr

docker push

docker pull

Docker Hub



Images

ubuntu-16.04

extended-ubuntu

datasci-hammer

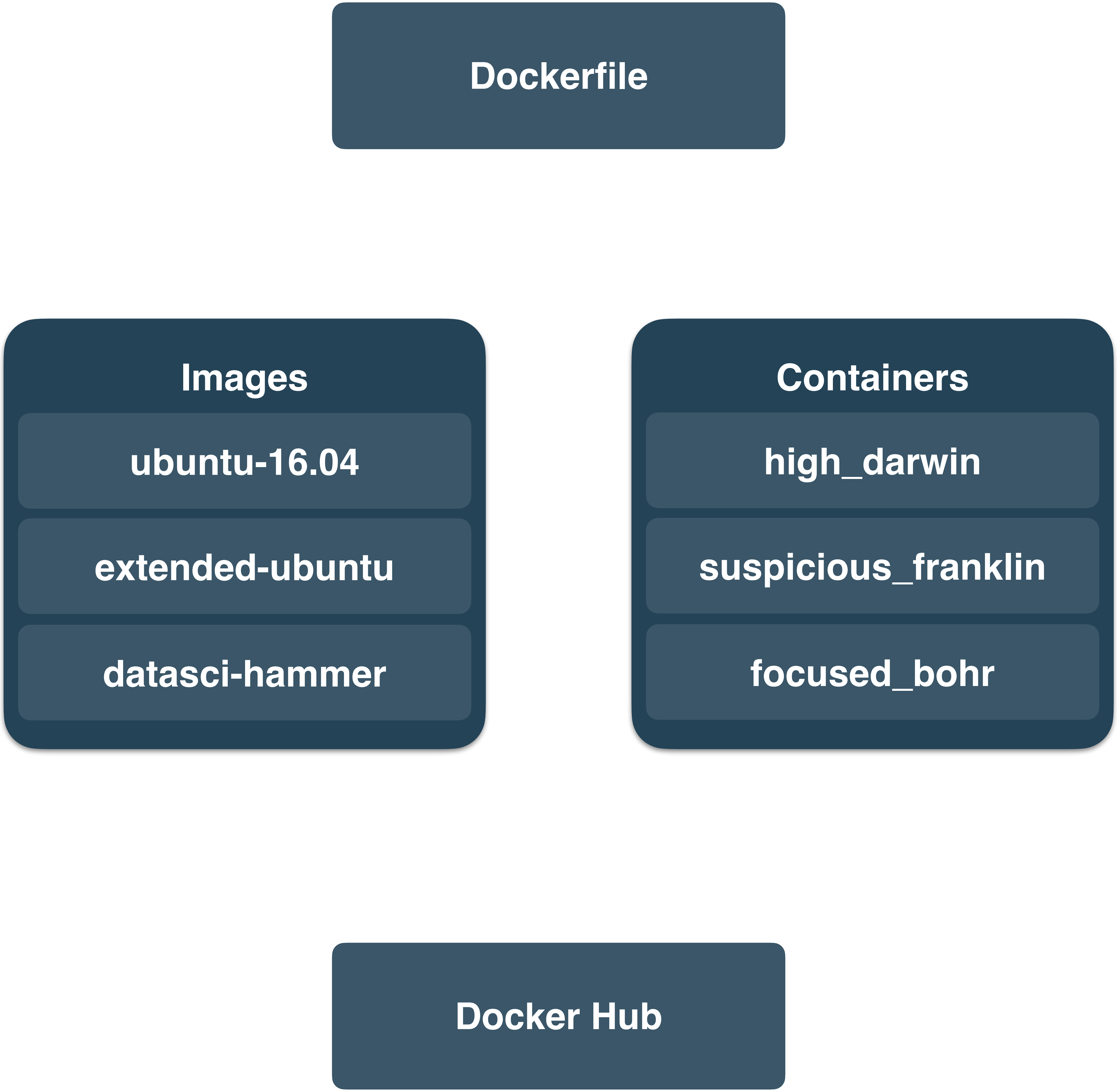
Containers

high_darwin

suspicious_franklin

focused_bohr

Docker Hub



Dockerfile

Images

ubuntu-16.04

extended-ubuntu

datasci-hammer

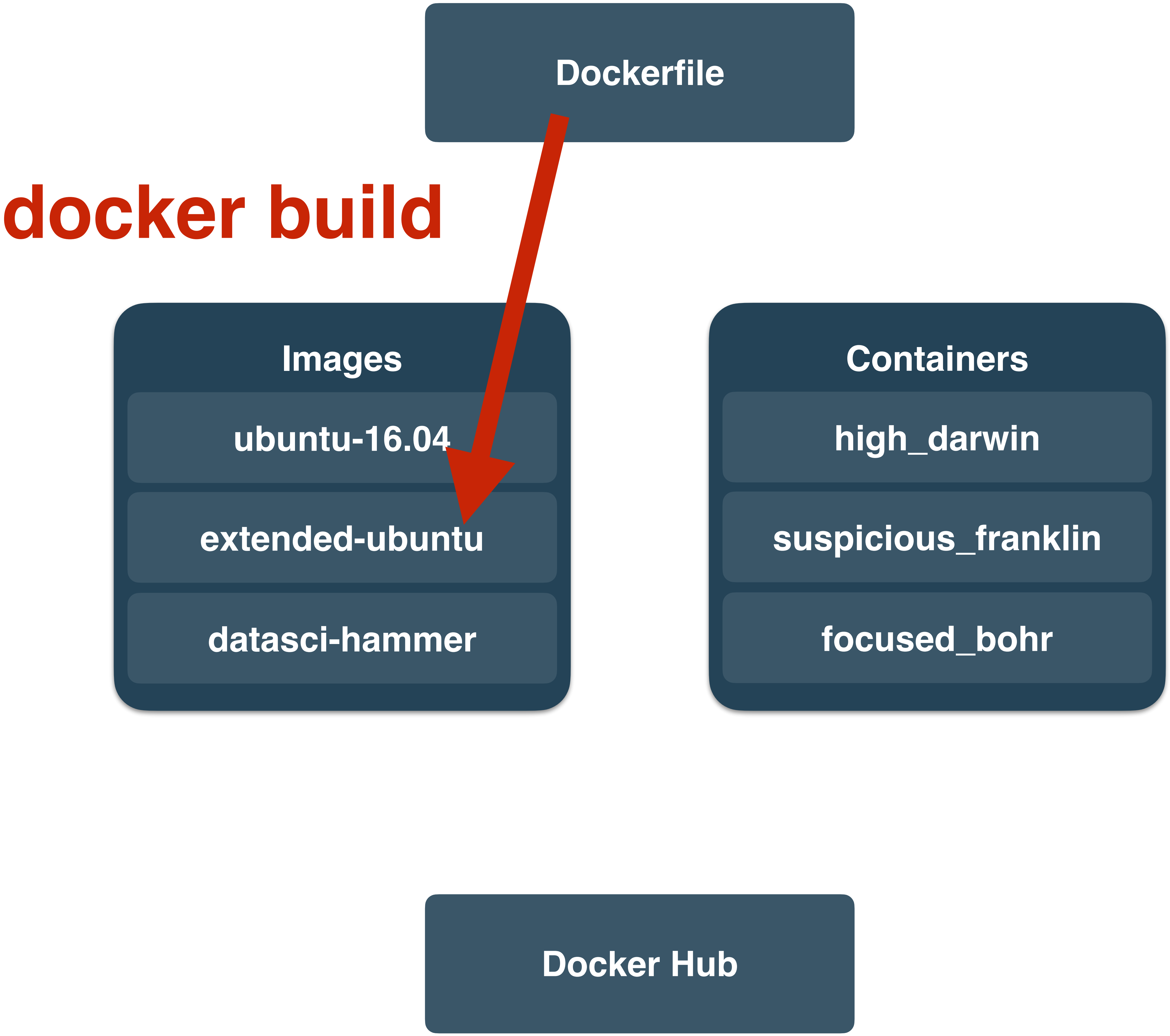
Containers

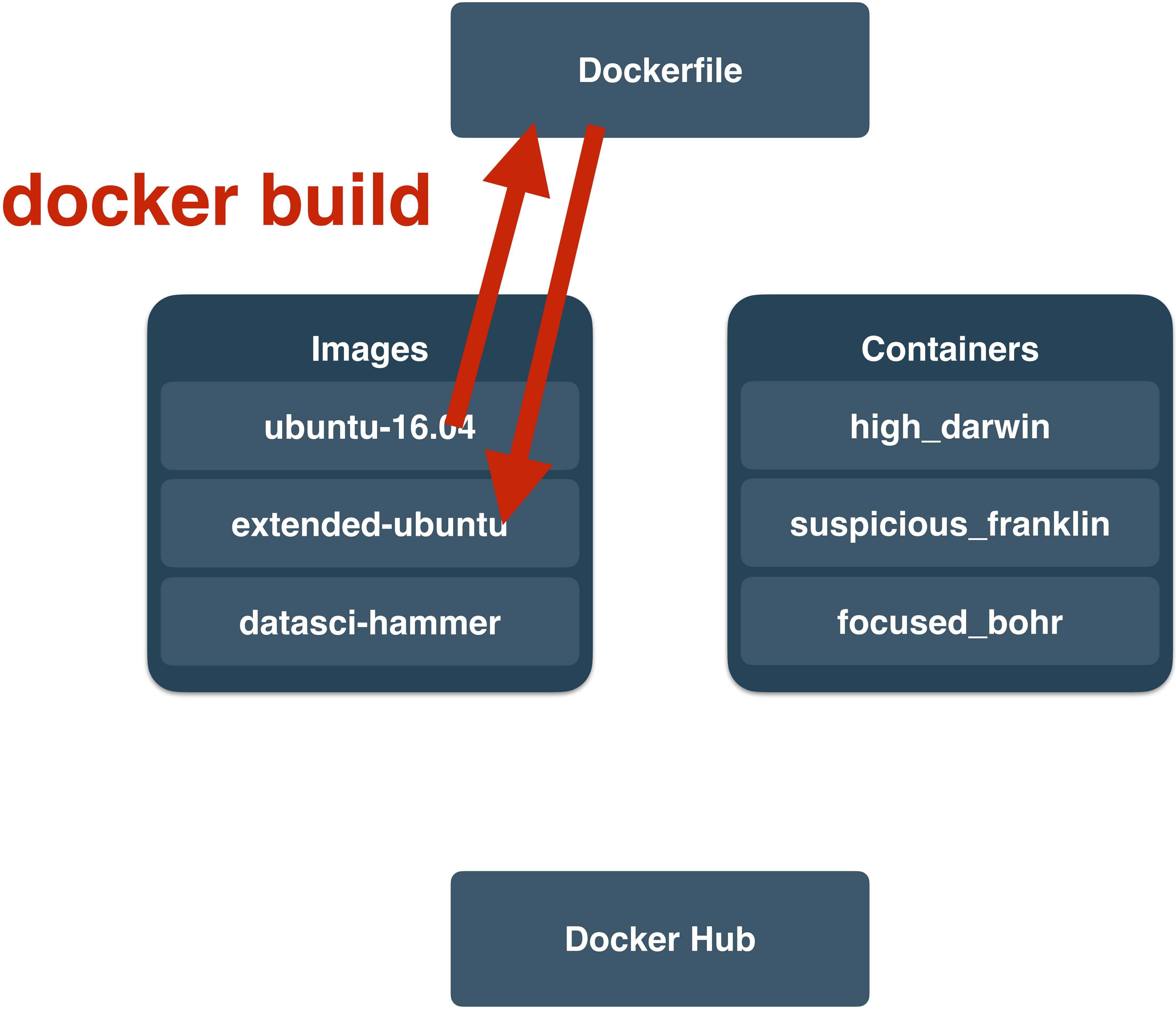
high_darwin

suspicious_franklin

focused_bohr

Docker Hub


Dockerfile
docker build
Images
ubuntu-16.04
extended-ubuntu
datasci-hammer
Containers
high_darwin
suspicious_franklin
focused_bohr
Docker Hub



docker build

Dockerfile

Images

ubuntu-16.04

extended-ubuntu

datasci-hammer

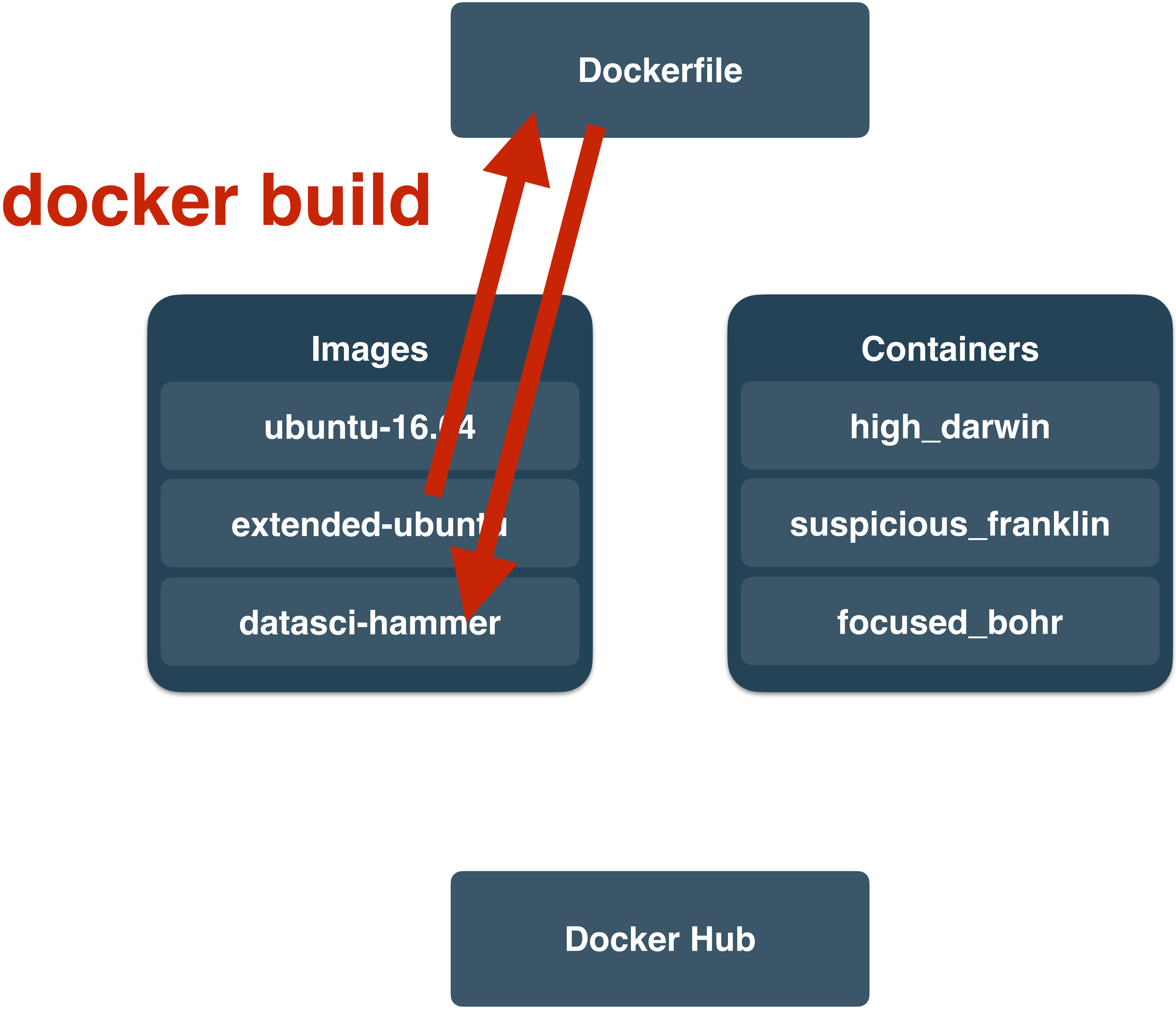
Containers

high_darwin

suspicious_franklin

focused_bohr

Docker Hub



docker build

Dockerfile

Images

ubuntu-16.04

extended-ubuntu

datasci-hammer

Containers

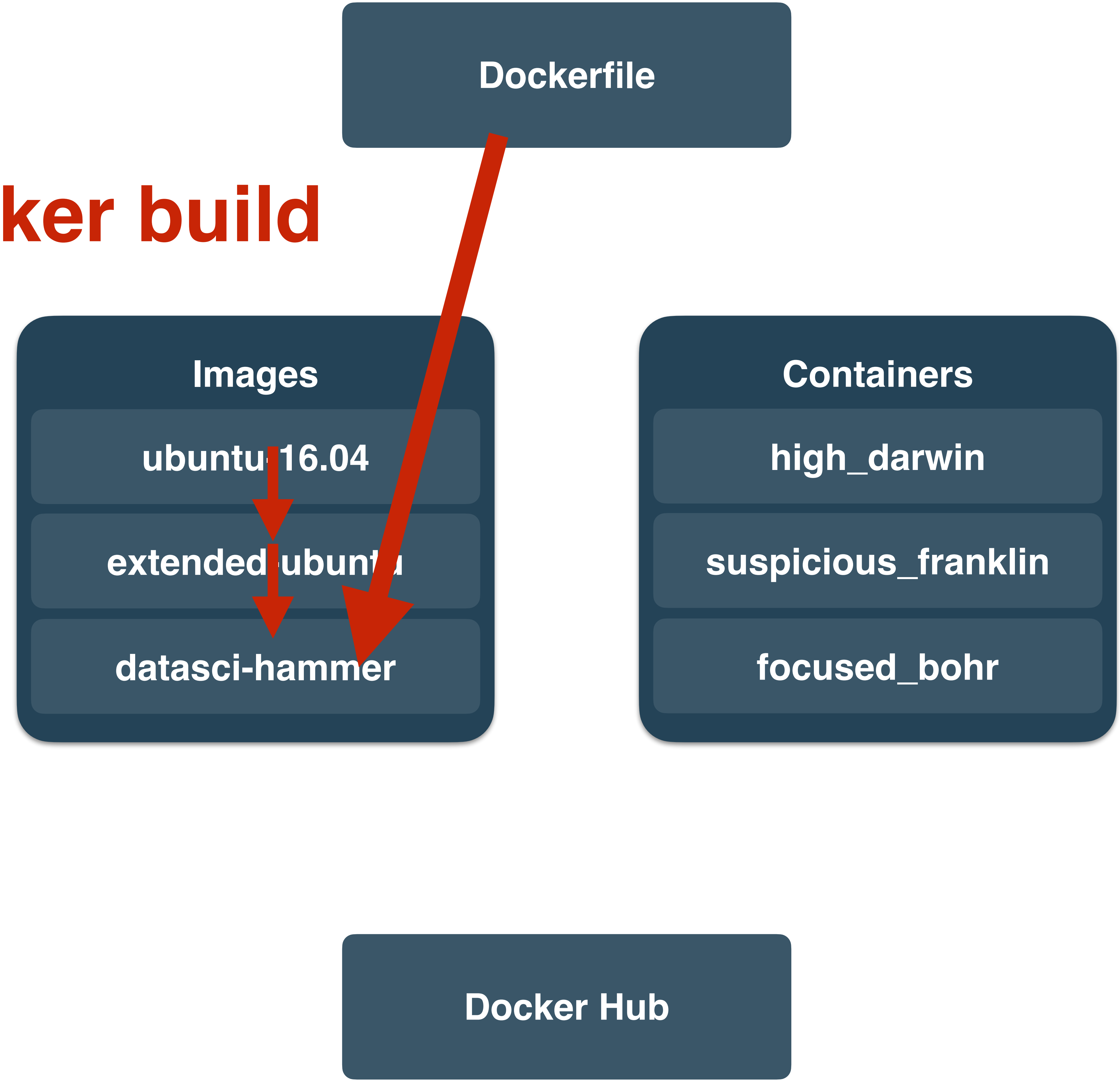
high_darwin

suspicious_franklin

focused_bohr

Docker Hub

docker build



Dockerfile

Images

ubuntu:16.04

extended-ubuntu

datasci-hammer

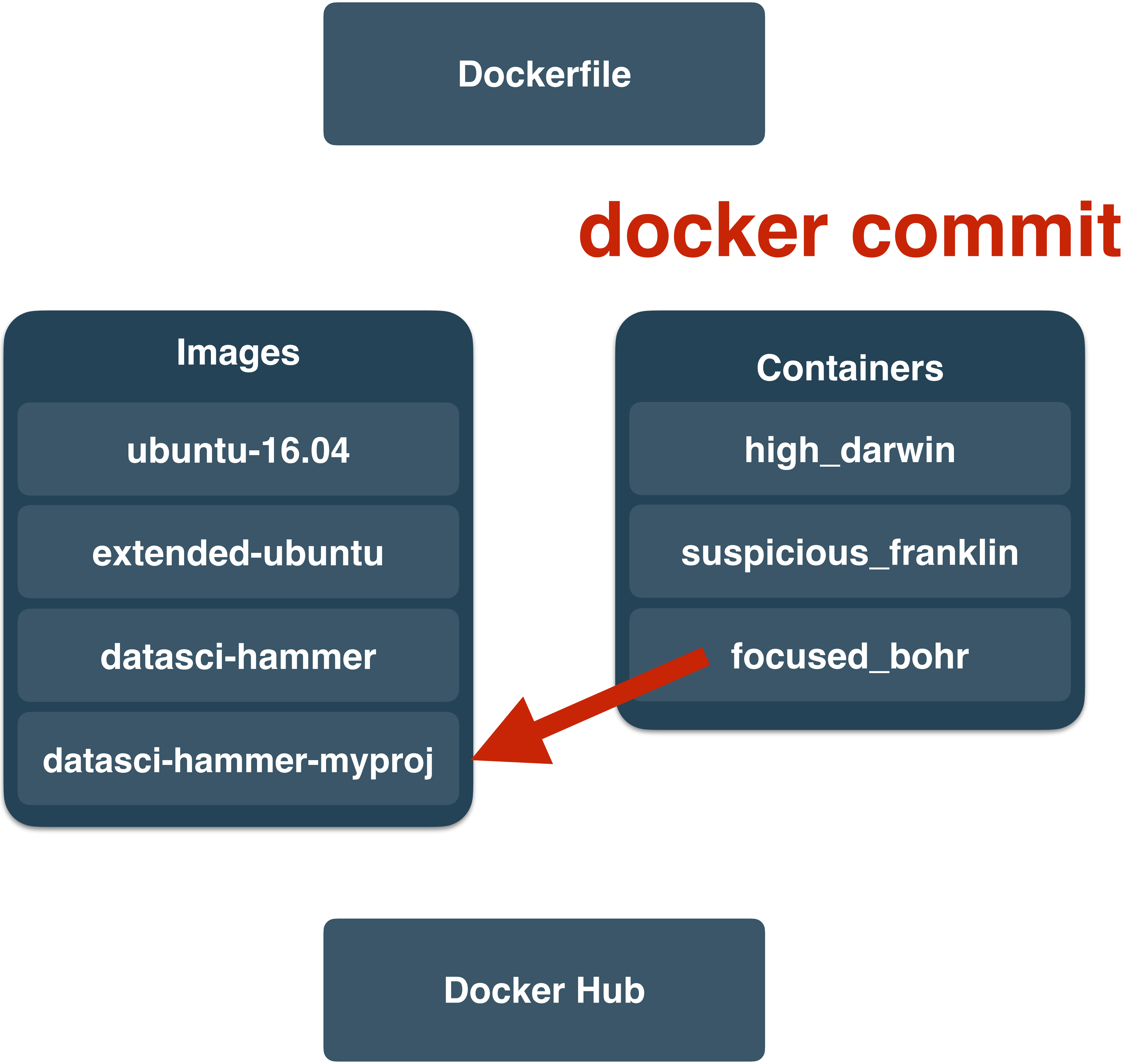
Containers

high_darwin

suspicious_franklin

focused_bohr

Docker Hub



Dockerfile

docker commit

Images

ubuntu-16.04

extended-ubuntu

datasci-hammer

datasci-hammer-myproj

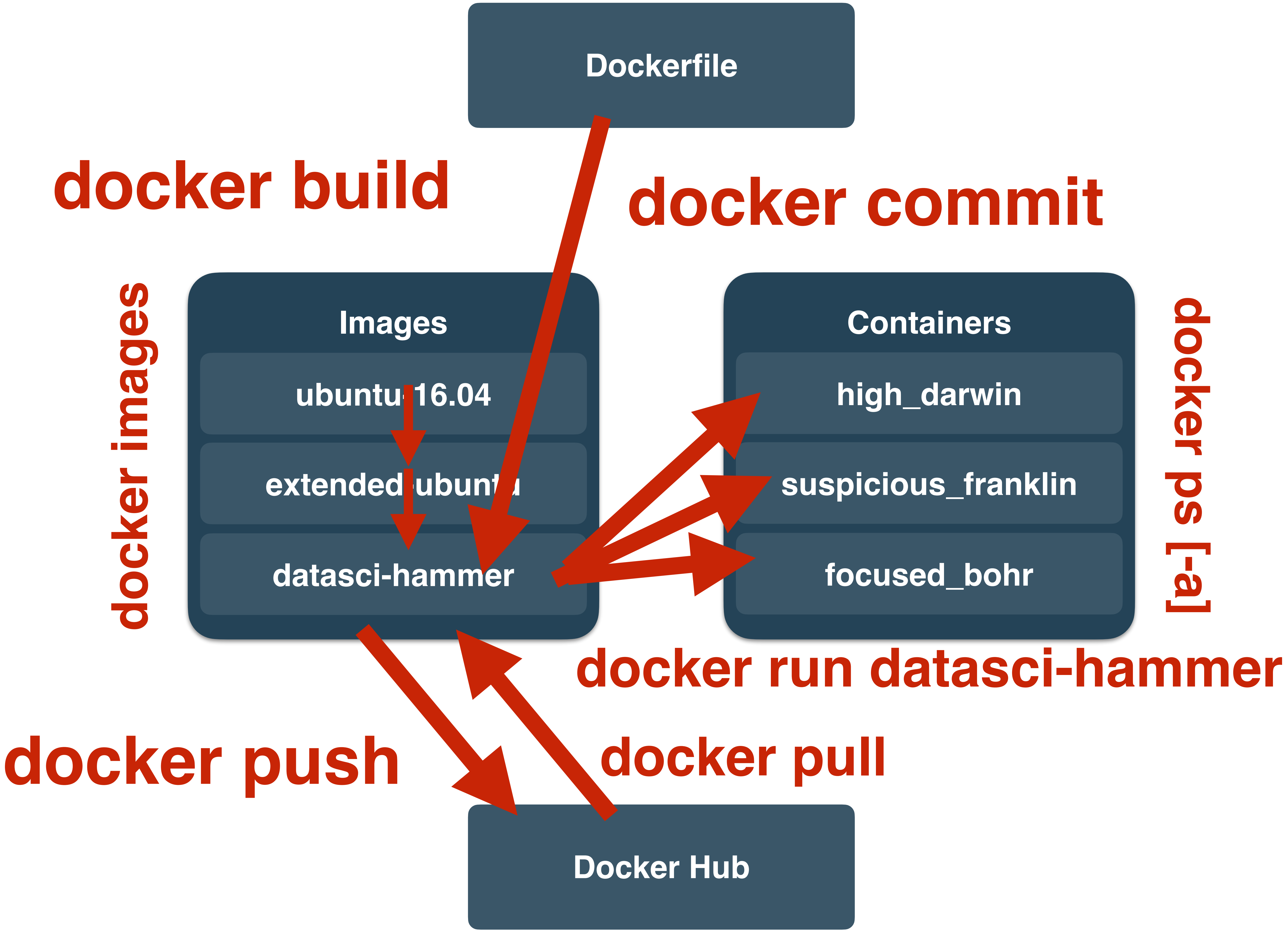
Containers

high_darwin

suspicious_franklin

focused_bohr

Docker Hub



Dockerfile

docker build

docker commit

docker images

Images

ubuntu:16.04

extended-ubuntu

datasci-hammer

Containers

high_darwin

suspicious_franklin

focused_bohr

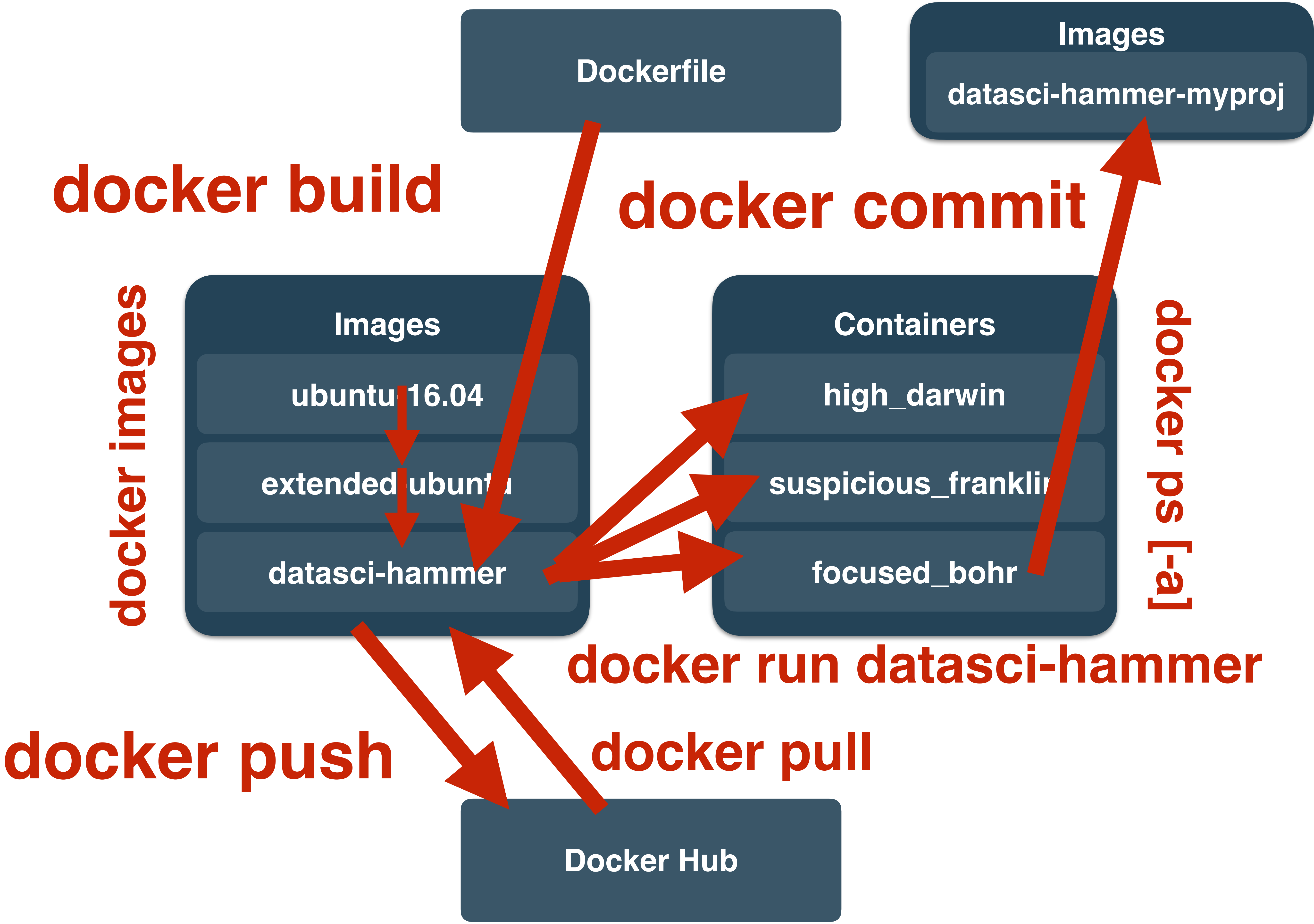
docker ps [-a]

docker run datasci-hammer

docker push

docker pull

Docker Hub



Dockerfile

Images

datasci-hammer-myproj

docker build

docker commit

docker images

Images

ubuntu:16.04

extended-ubuntu

datasci-hammer

Containers

high_darwin

suspicious_franklin

focused_bohr

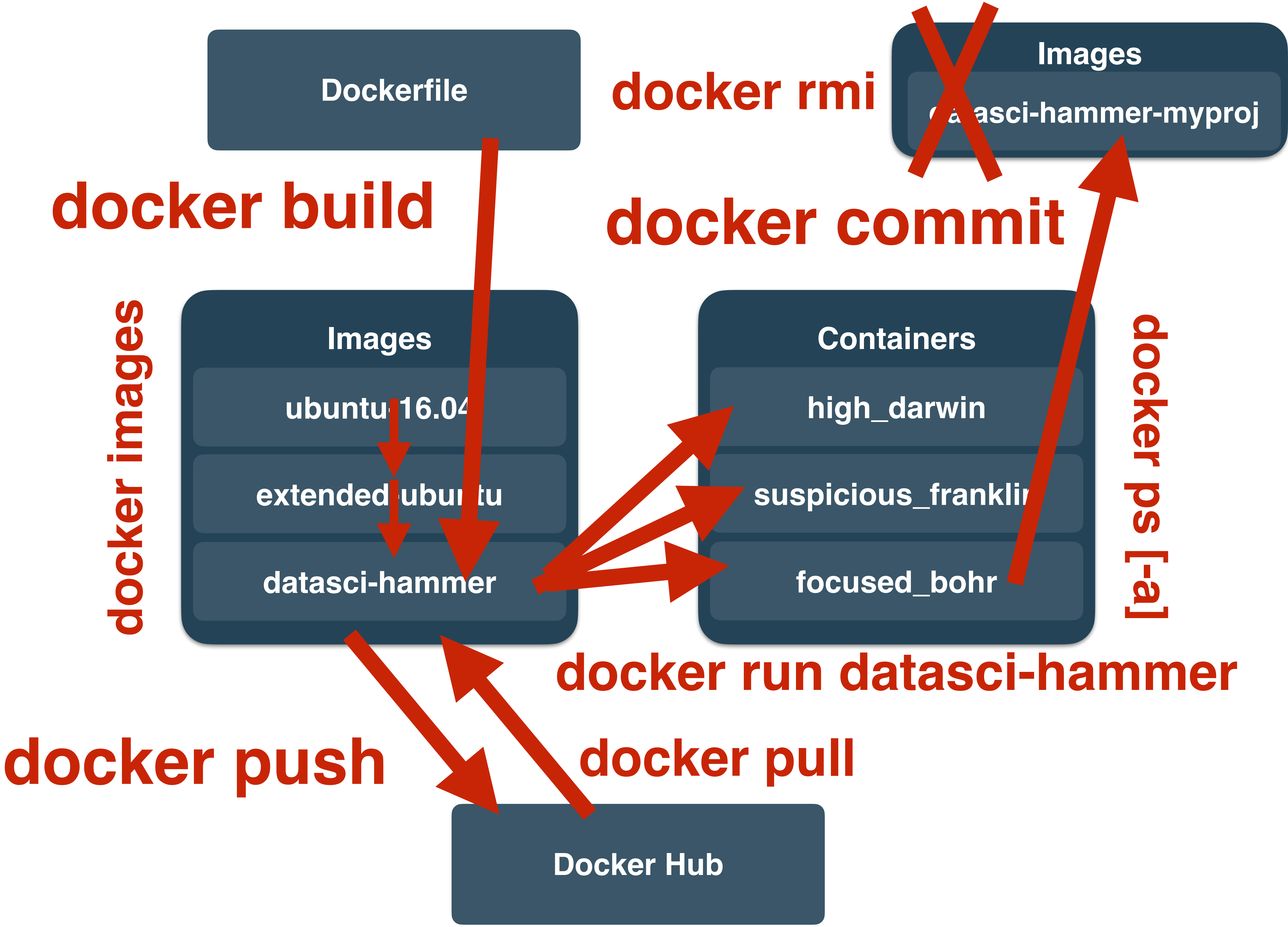
docker ps [-a]

docker run datasci-hammer

docker push

docker pull

Docker Hub



Dockerfile

docker rmi

Images

datasci-hammer-myproj

docker build

docker commit

docker images

Images

ubuntu:16.04

extended-ubuntu

datasci-hammer

Containers

high_darwin

suspicious_franklin

focused_bohr

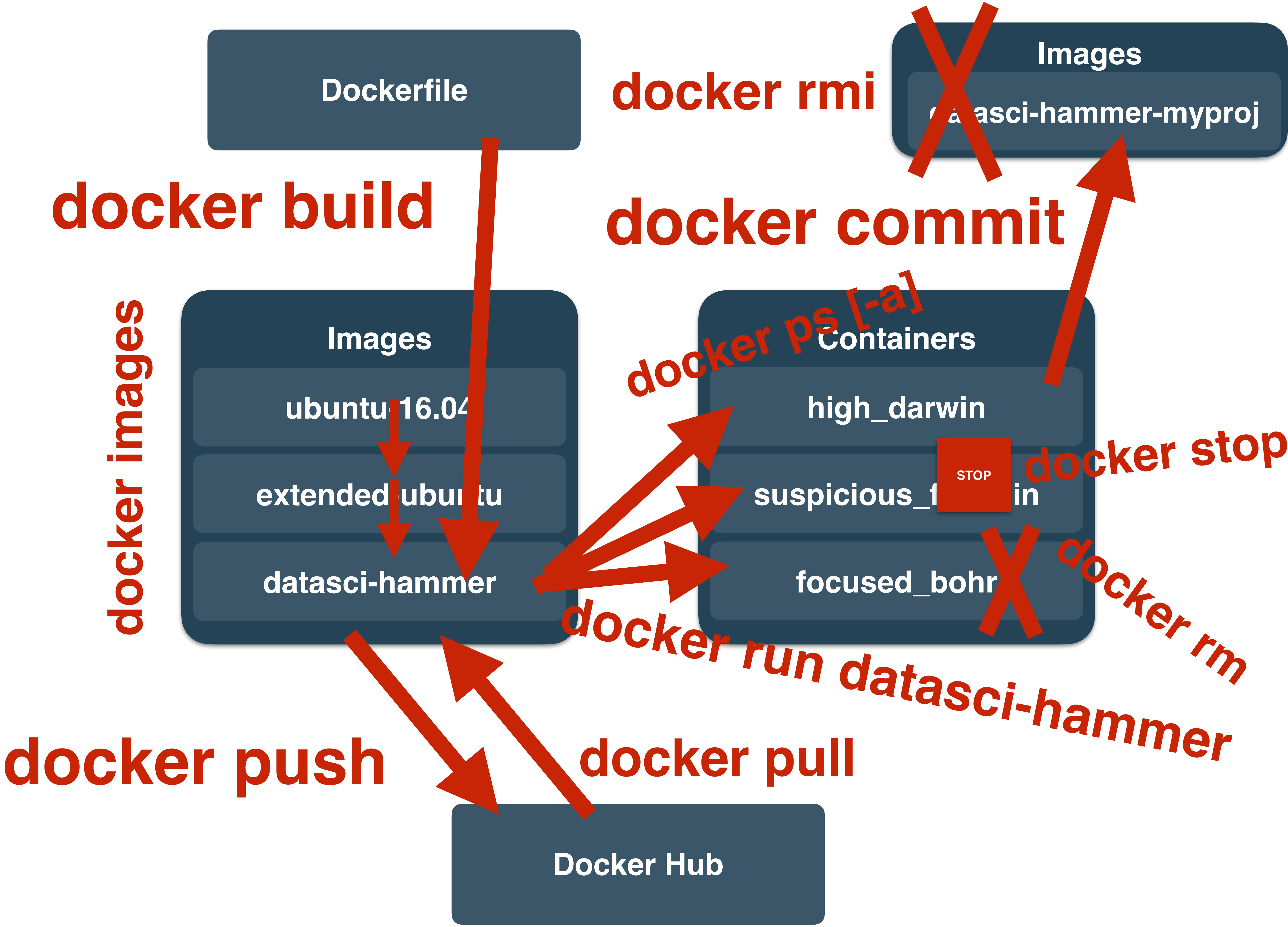
docker ps [-a]

docker run datasci-hammer

docker push

docker pull

Docker Hub



Dockerfile

docker rmi

Images

datasci-hammer-myproj

docker build

docker commit

docker images

Images

ubuntu:16.04

extended-ubuntu

datasci-hammer

docker ps [-a]

Containers

high_darwin

suspicious_f in

STOP

docker stop

focused_bohr

docker rm

docker run datasci-hammer

docker push

docker pull

Docker Hub

How

How to use Docker

In three steps:

1. Build or pull an image.
2. Run a container.
3. Execute code against container.

Build or pull an image

- Pull: easy!
  - This happens automatically with `docker run` or `docker build`
  - `docker pull andyreagan:datasci-hammer-ubuntu`
  - `docker pull centos:latest`

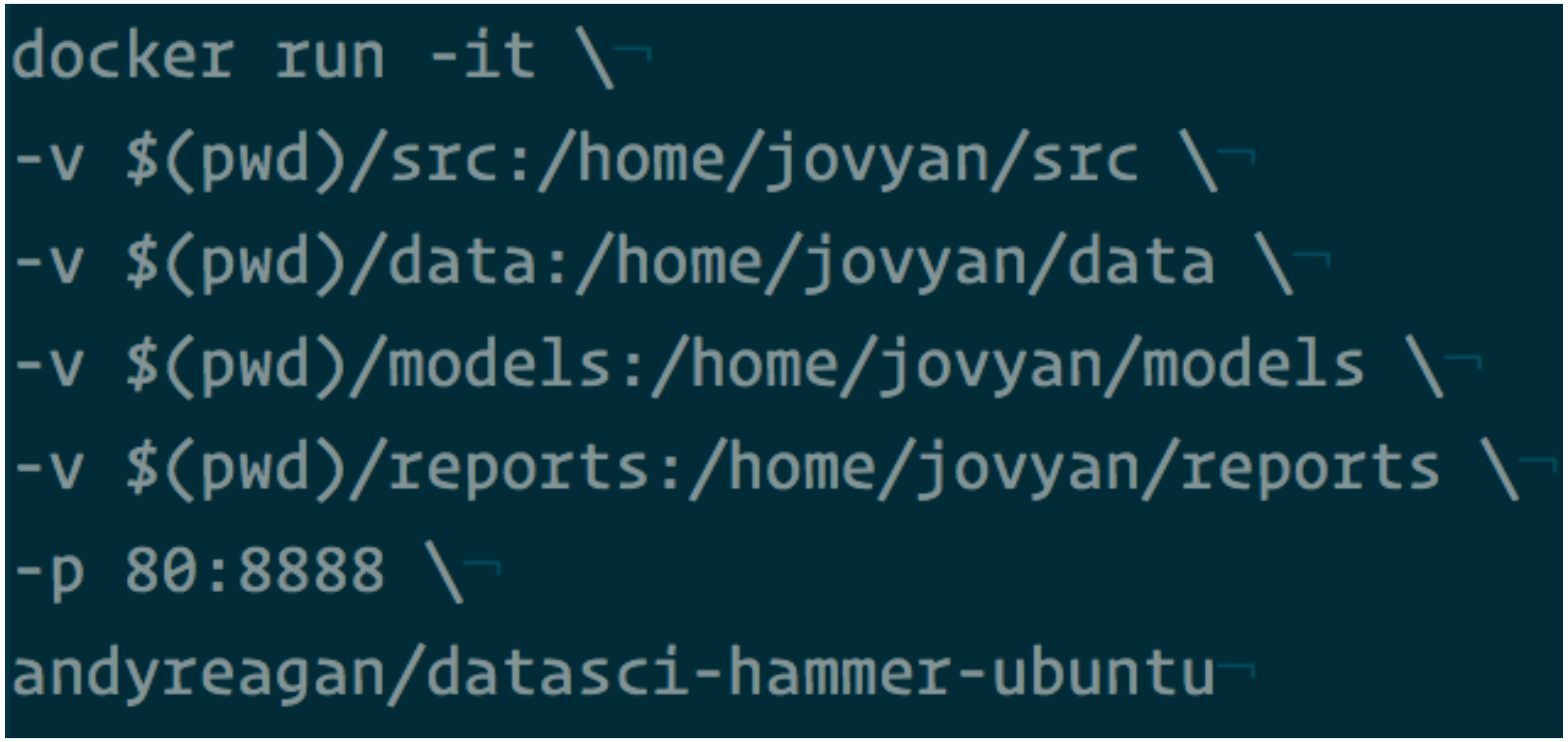
Build or pull an image

- Build: much more work
  - Dockerfile
    - Dockerfile language: RUN, COPY, USER, ADD, ENV...

Run a container

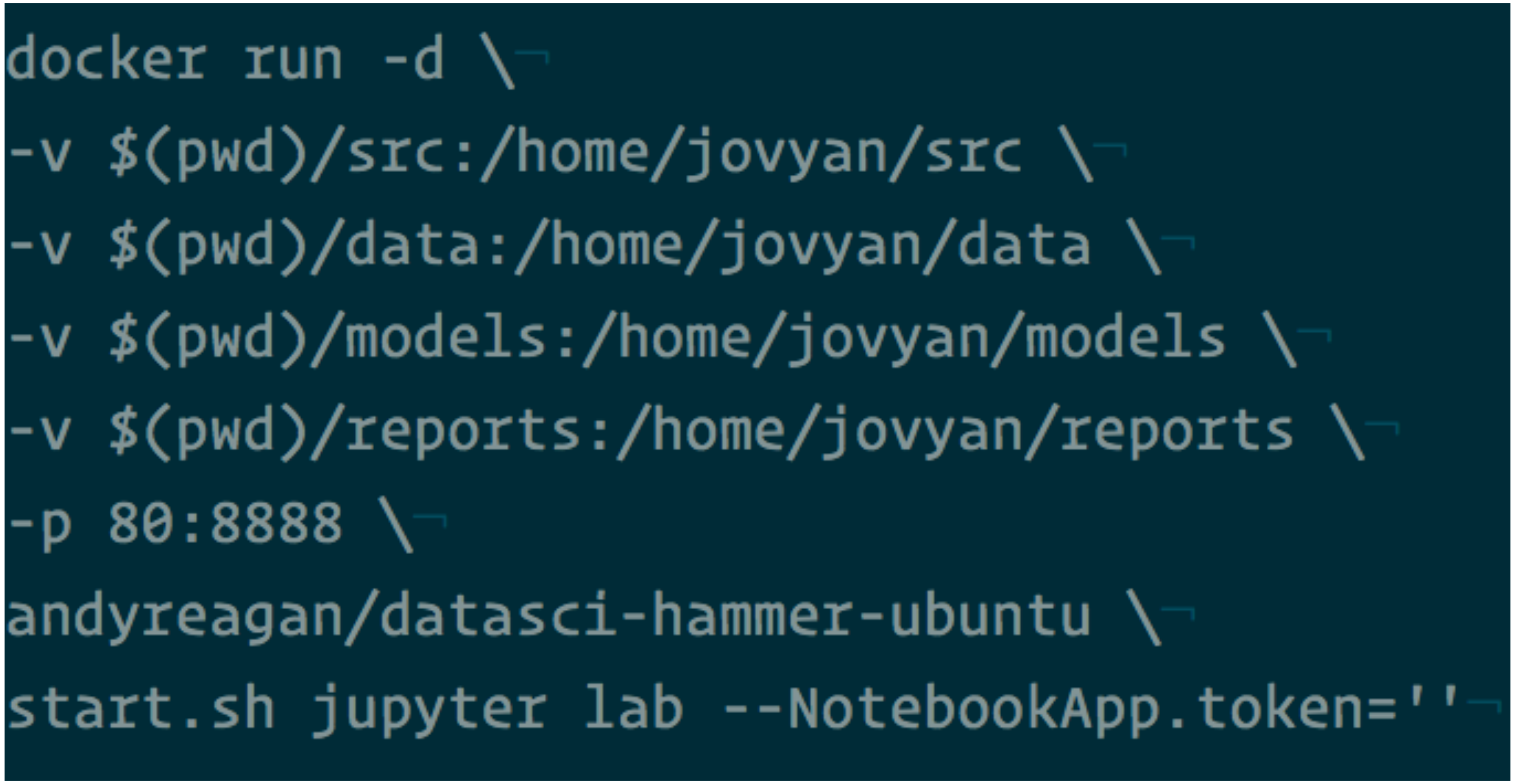
- Single command: `docker run`
- Many options
    - -d: daemon mode
    - -it: interactive mode
    - -v: mount volume
    - -p: map port
    - -e: pass env vars (I don't use this, but others might)

Run a container



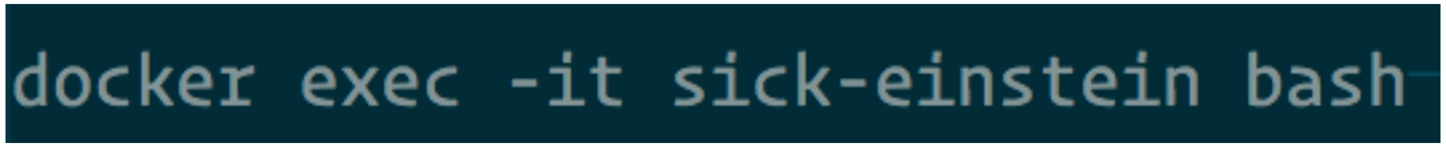
```
docker run -it \
-v $(pwd)/src:/home/jovyan/src \
-v $(pwd)/data:/home/jovyan/data \
-v $(pwd)/models:/home/jovyan/models \
-v $(pwd)/reports:/home/jovyan/reports \
-p 80:8888 \
andyreagan/datasci-hammer-ubuntu
```

Run a container



```
docker run -d \
-v $(pwd)/src:/home/jovyan/src \
-v $(pwd)/data:/home/jovyan/data \
-v $(pwd)/models:/home/jovyan/models \
-v $(pwd)/reports:/home/jovyan/reports \
-p 80:8888 \
andyreagan/datasci-hammer-ubuntu \
start.sh jupyter lab --NotebookApp.token=''
```

Reconnect



```
docker exec -it sick-einstein bash
```

Execute code!

As you wish!

Performance of Docker

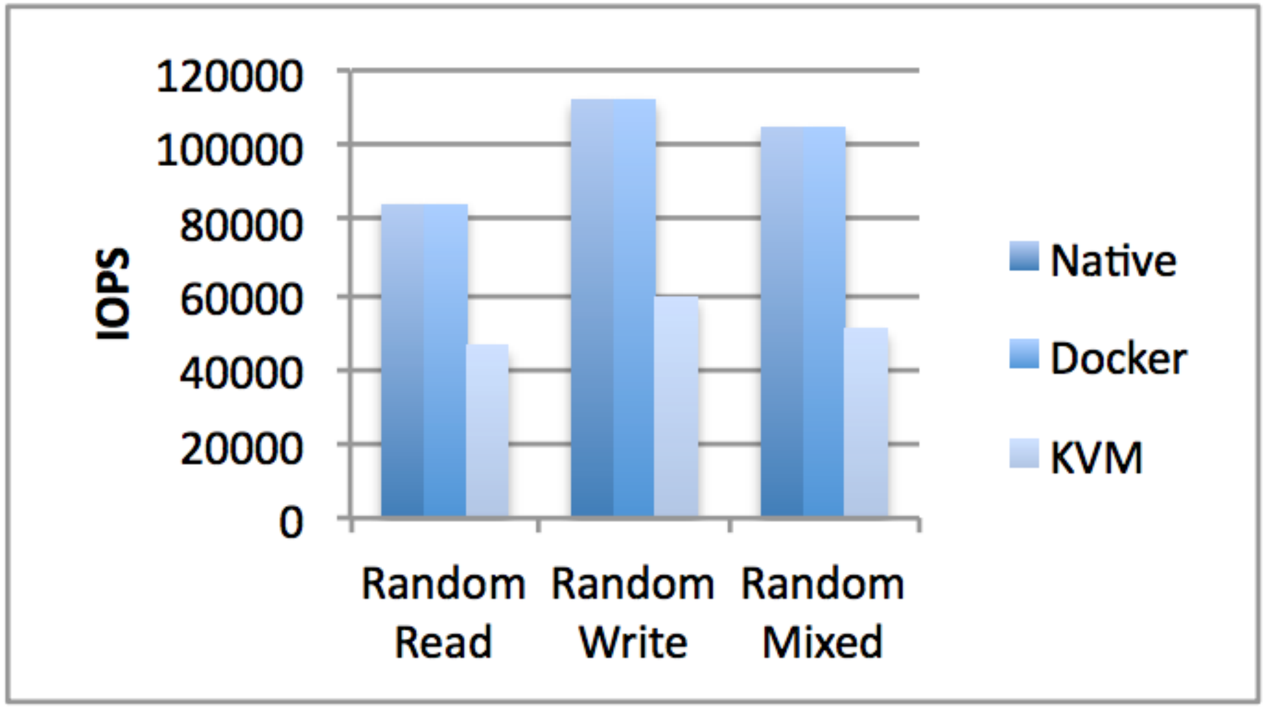


Fig. 6. Random I/O throughput (IOPS).

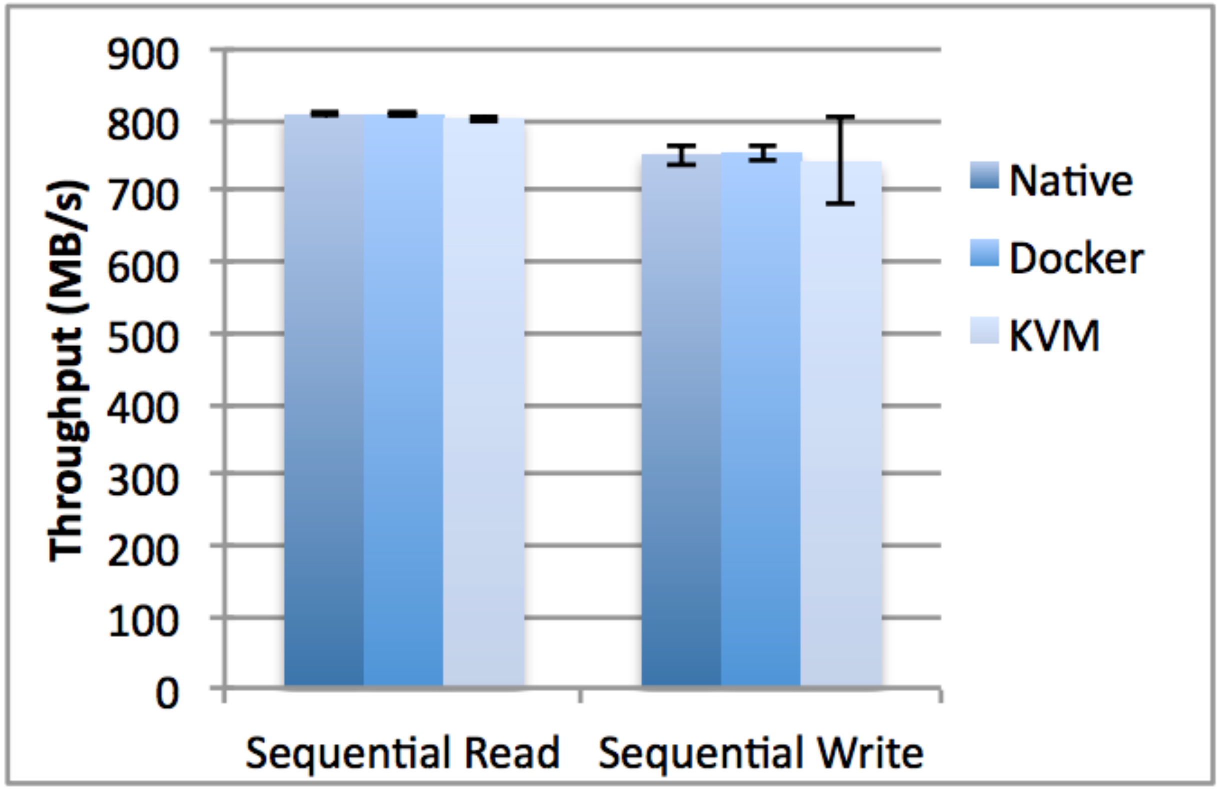


Fig. 5. Sequential I/O throughput (MB/s).

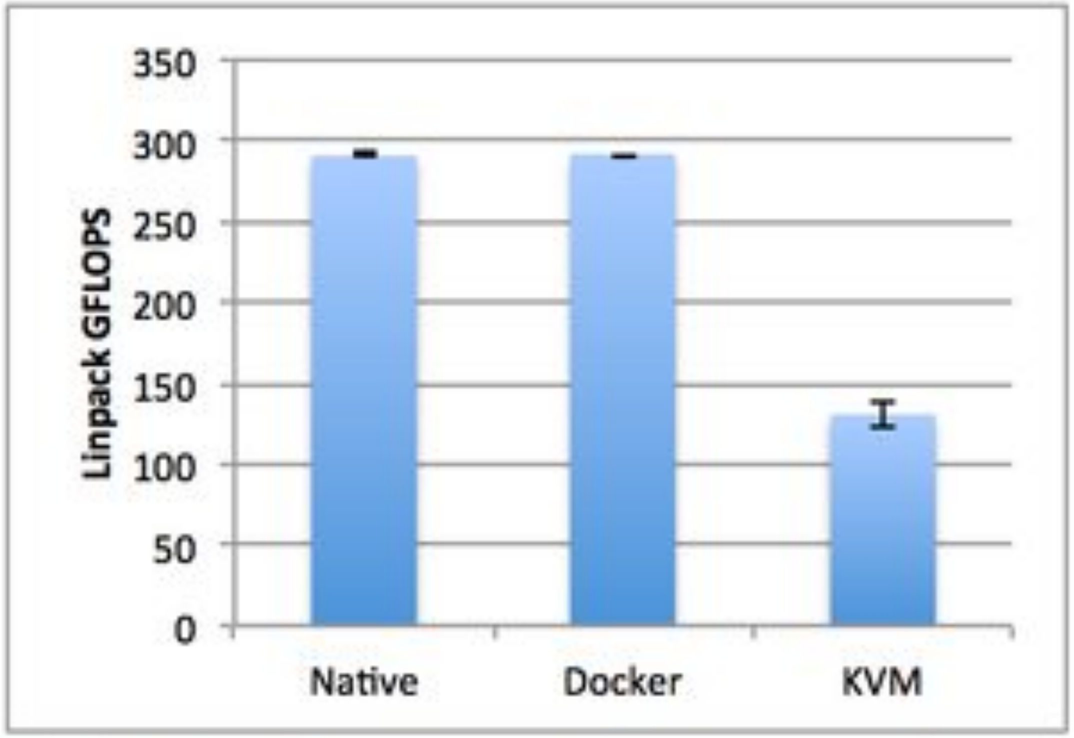


Figure 1. Linpack performance on two sockets (16 cores). Each data point is the arithmetic mean obtained from ten runs. Error bars indicate the standard deviation obtained over all runs.

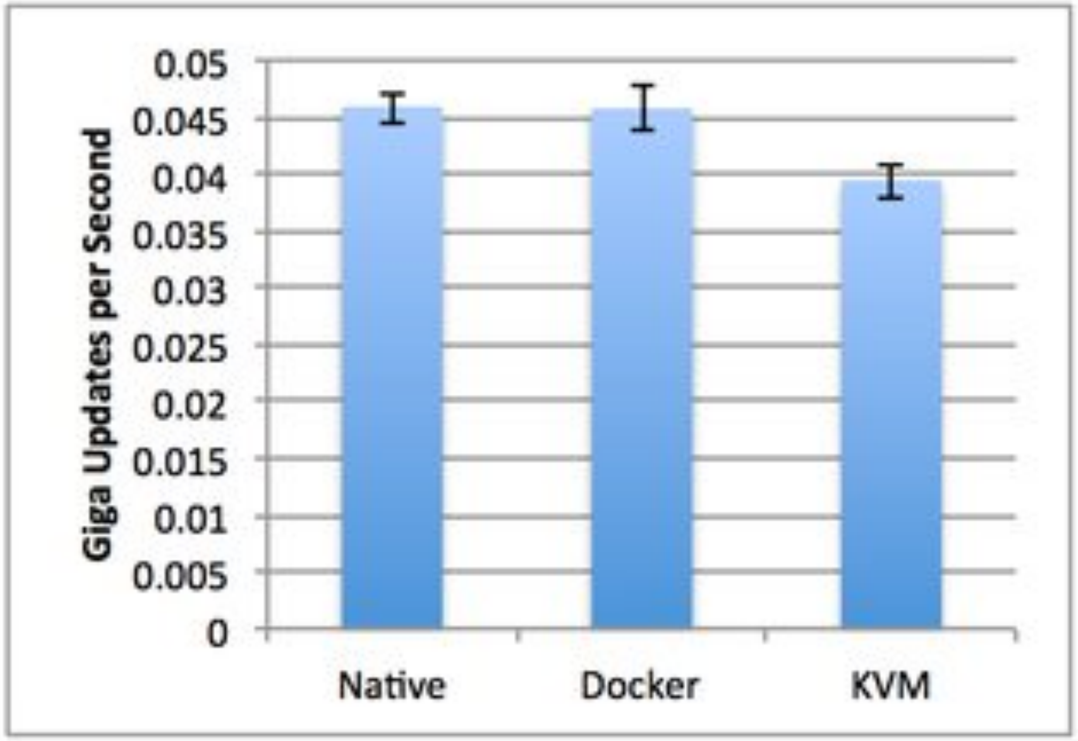


Figure 4. RandomAccess performance on a single socket (8 cores). Each data point is the arithmetic mean obtained from ten runs. Error bars indicate the standard deviation obtained over all runs.

Did we solve these problems?
You tell me!

- Reproducibility.
- Scalable computational backend.
- Persistent computational environment.
- Shared dependency management.
- Lack of appropriate permissions on a server.