

Winter Institute in Data Science and Big Data

Containers, Cloud Computing, and Code Reproducibility: Docker, Kubernetes, and Code Ocean

Le Bao

Massive Data Institute, Georgetown University

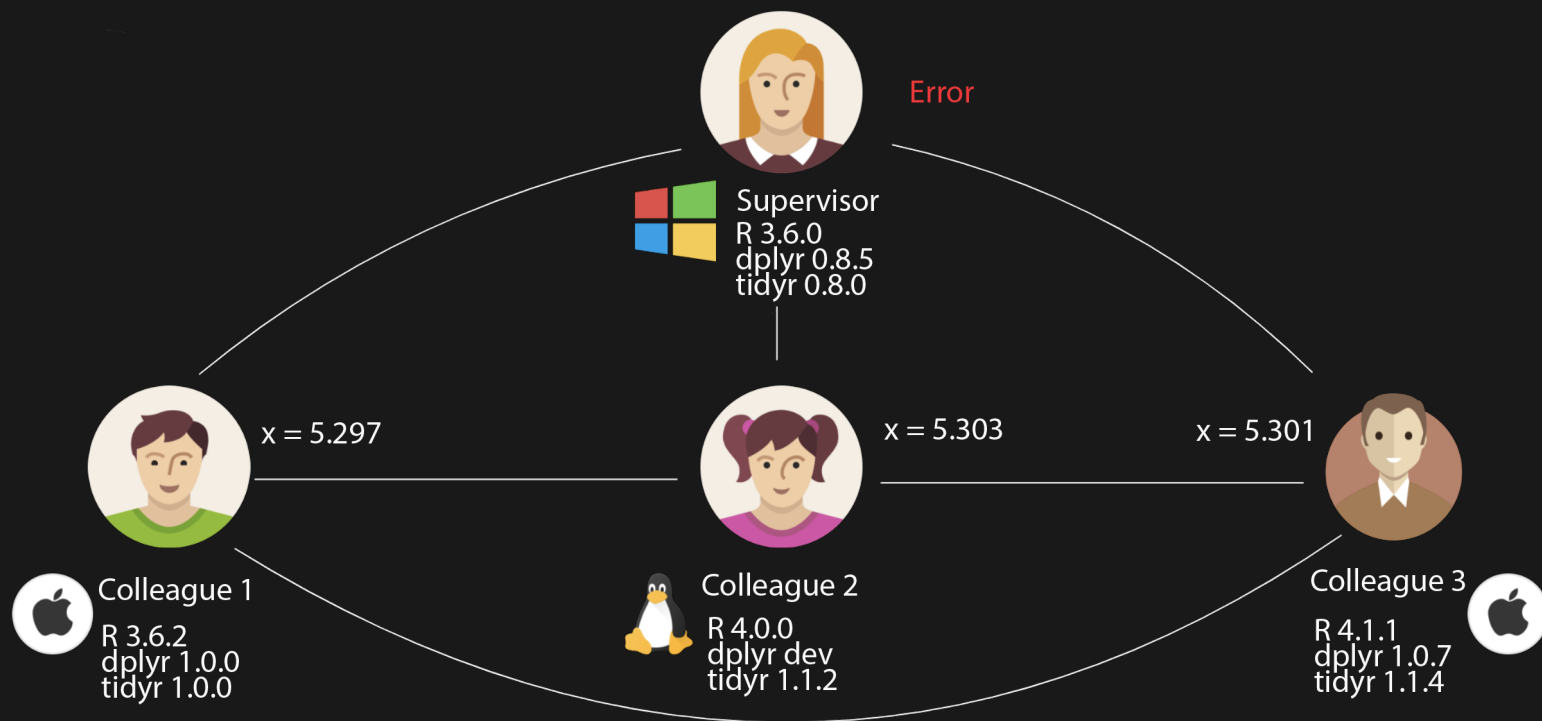
January 10th, 2023

Why?

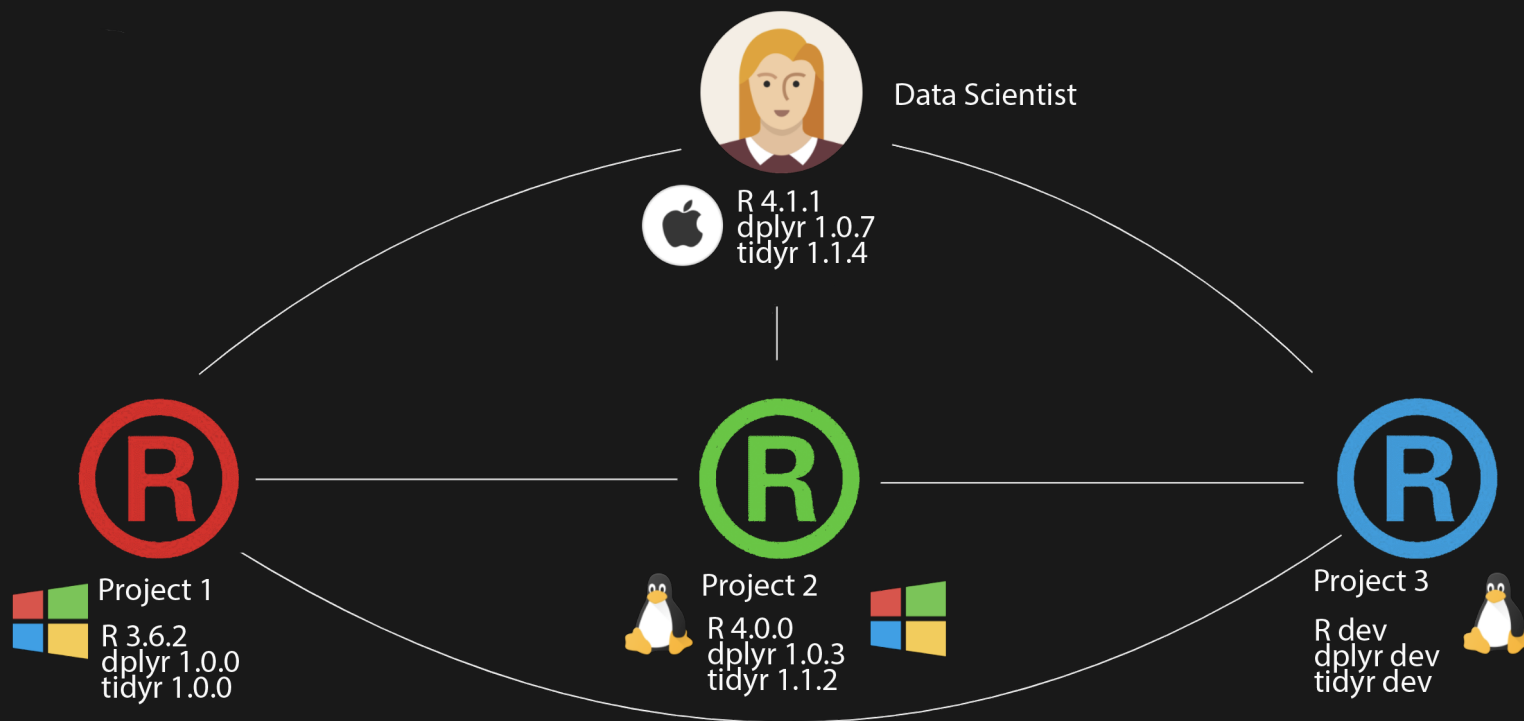


- Docker and Kubernetes are the cutting-edge tools for tech and tech-related industries and scientific research.
- "The share of jobs containing Docker as a skill on Indeed increased by 9,538% from 2014 to 2019."

The Problem of Reproducibility



The Problem of Reproducibility



The Problem of Reproducibility

- Our computing tools are increasingly powerful, diverse, cloud-based.
 - iPhone 6 is 32,600 times faster than the Apollo Guidance Computer (APC).
 - Supercomputer is now accessible to everyone through cloud computing.
- Our work is required to be more open, transparent, and collaborative.
 - Reproducible research, open-sourced projects, etc.
- Our data becomes bigger, higher dimensional, multimodal.
 - Big data, image/voice data, etc.
- Our analysis needs to be fast, instant, and real-time.
 - Real-time analytic, the pandemic, OpenTable & the State of the Industry
- ...
- **Our computing environment is increasingly complex and convoluted.**

Computing Environment

- Computing environment:
 - Hardware
 - Software
 - Operating system
 - System dependencies
 - R/python: versions, packages/libraries (last time)
- Goals:
 - Control the whole computing environment.
 - Configure the environments as we want.
 - Make the environment reproducible.
 - Share the code (that may require specific environment).

Today

- Container
- Docker
 - Run `R` / `python` with Docker
- Kubernetes
- Cloud computing and Code Ocean
 - Both front- and back-ends

What is a Container?



What is a Container?

- A standard unit of software that packages up code and all its dependencies
 - Operating system: linux (most common), Windows, Mac OS, etc. and data center, cloud, serverless.
 - System-level dependencies
 - Software packages: R, python, TensorFlow, MySQL, etc.
 - Software dependencies: tidyverse, NumPy, PyTorch
 - Including everything needed to run code

Why Container?

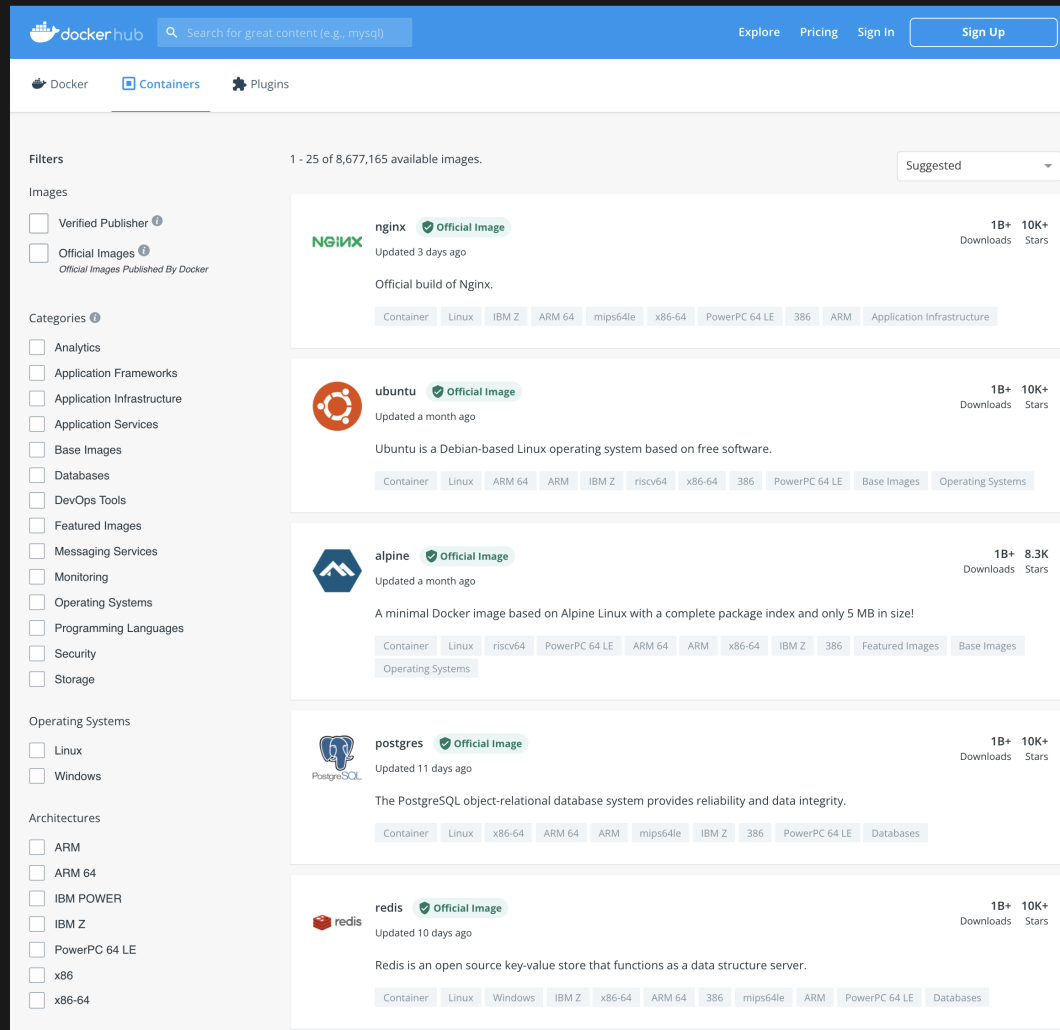
- Lightweight
 - Standalone and standard implementation
 - Isolated from host system
 - Portable and shareable
 - Secure
-
- Container allows us to deploy, replicate, move, and back up a workload in one streamlined way

Docker

- Docker is a platform for building, configuring, and delivering containers
- Docker (2013-) is the *de facto* industry standard for containers (1970-)
- Features:
 - Improved and seamless portability from laptop to any desktop, data center and cloud environment.
 - Collaboration
 - HPC and cloud application
 - Isolated, transparent, and reproducible implementation
 - Open-sourced, and community optimized
 - Docker Hub
 - User-created images
 - Versioned
 - Almost all the versions of R, python, etc.
 - Layered
 - Each additional layer will built upon the existing ones

Docker Image

- A static, read-only template for creating containers.



The screenshot displays the Docker Hub interface. At the top, there's a navigation bar with the Docker Hub logo, a search bar, and links for Explore, Pricing, Sign In, and Sign Up. Below the navigation bar, there are tabs for Docker, Containers (selected), and Plugins. The main content area shows a list of Docker images. On the left, there are filters for Images (Verified Publisher, Official Images), Categories (Analytics, Application Frameworks, etc.), Operating Systems (Linux, Windows), and Architectures (ARM, ARM 64, etc.). The list of images includes:

- nginx** (Official Image): Updated 3 days ago. 1B+ Downloads, 10K+ Stars. Official build of Nginx. Architectures: Container, Linux, IBM Z, ARM 64, mips64le, x86-64, PowerPC 64 LE, 386, ARM, Application Infrastructure.
- ubuntu** (Official Image): Updated a month ago. 1B+ Downloads, 10K+ Stars. Ubuntu is a Debian-based Linux operating system based on free software. Architectures: Container, Linux, ARM 64, ARM, IBM Z, riscv64, x86-64, 386, PowerPC 64 LE, Base Images, Operating Systems.
- alpine** (Official Image): Updated a month ago. 1B+ Downloads, 8.3K Stars. A minimal Docker image based on Alpine Linux with a complete package index and only 5 MB in size! Architectures: Container, Linux, riscv64, PowerPC 64 LE, ARM 64, ARM, x86-64, IBM Z, 386, Featured Images, Base Images, Operating Systems.
- postgres** (Official Image): Updated 11 days ago. 1B+ Downloads, 10K+ Stars. The PostgreSQL object-relational database system provides reliability and data integrity. Architectures: Container, Linux, x86-64, ARM 64, ARM, mips64le, IBM Z, 386, PowerPC 64 LE, Databases.
- redis** (Official Image): Updated 10 days ago. 1B+ Downloads, 10K+ Stars. Redis is an open source key-value store that functions as a data structure server. Architectures: Container, Linux, Windows, IBM Z, x86-64, ARM 64, 386, mips64le, ARM, PowerPC 64 LE, Databases.

Use Docker to Run R

- Rocker project: <https://www.rocker-project.org/>

Image	Description
r-base	Current R version
r-devel	R-devel added side-by-side onto r-base
r-ver	Specify R version
rstudio	Adds rstudio
tidyverse	Adds tidyverse & devtools
verse	Adds tex & publishing-related packages
geospatial	Adds geospatial libraries

Run **R** Using Pre-built Images

- Command:

```
docker run -it --rm rocker/r-base
```

- `docker run`: run processes based on an image
- General form: `docker run [OPTIONS] IMAGE[:TAG|@DIGEST] [COMMAND] [ARG ...]`
- `-it`: interactive session
- `-rm`: automatically remove container once stopped
- `rocker/r-base`: pull latest r-base from rocker repository

Run Using Pre-built Images

- Pull a specific version using `rocker/r-ver:4.2.0`

```
docker run -it --rm rocker/r-ver:4.2.0
```

- Run an RStudio server

```
docker run --rm -p 8888:8787 -e PASSWORD='mypassword' rocker/rstudio:4.2.0
```

For apple silicon machines:

```
docker run --rm -p 8888:8787 -e PASSWORD='mypassword' rocker/rstudio:latest-daily
```

- `-p`: publish a port
- `-e`: set environment variable

Using Docker to Run python

- Lots of images with different support and configurations

```
docker run -it --rm python
docker run -it --rm python:3.10.4
```


Build Your Own **R** Docker Image

Dockerfile: instructions for assembling and configuring an Docker image.

```
FROM rocker/r-ver:4.2.0

# System dependencies
RUN apt-get update && apt-get install -y curl libz-dev

# R packages
RUN Rscript -e 'install.packages("MASS")'
RUN install2.r readr dplyr ggplot2 forcats

## Copy files
RUN mkdir docker-demo
COPY data docker-demo/data
COPY code docker-demo/code
RUN mkdir docker-demo/output
#ADD . docker-demo

## Set working directory
WORKDIR docker-demo
```

Build Your Own **R** Docker Image

- Build image

```
docker build -t demo:r .
```

- **-t**: name tag for the image
- **.**: root directory (where the Dockerfile is)
- Run a container using the image

```
docker run -i -t demo:r /bin/bash
```

- **-i**: interactive
- **-t**: name tag of the image
- **/bin/bash**: start a bash session

Build Your Own `python` Docker Image

```
FROM python:3.10.4

# python libraries
RUN pip install -U bs4

## Copy files
RUN mkdir docker-demo
RUN mkdir docker-demo/data
ADD . docker-demo

## Set working directory
WORKDIR docker-demo
```

- Build image

```
docker build -t demo:python .
```

- Run a container using the image

```
docker run -i -t demo:python /bin/bash
```

Managing Docker Container

- List and commit current containers

```
docker ps -l  
docker commit [CONTAINER ID] [NAME]
```

- Build a Docker image

```
docker images  
docker create [IMAGE ID]
```

- Extract files

```
docker cp [ID]:[Container PATH] [Local PATH]
```

- Remove containers and images

```
docker rm -f [Container ID]  
docker image rm [Image ID]  
docker system prune -all
```

Exercise

- Verify Docker Installation
 - Open Terminal (Mac/Linux) or Command Prompt/PowerShell (Windows)
 - Run `docker run hello-world`
- Run an R or python container with a specific version using `docker run`.
- Run R and python container using Dockerfile.
 - Go to `/docker-demo-python` and build a `demo-python` container using the provided Dockerfile.
 - *Feel free to use your own project.
 - Run the container and test the code script in `/docker-demo-python/code`
 - Extract the output file using `docker cp`
 - Follow the same procedure for `/docker-demo-r`

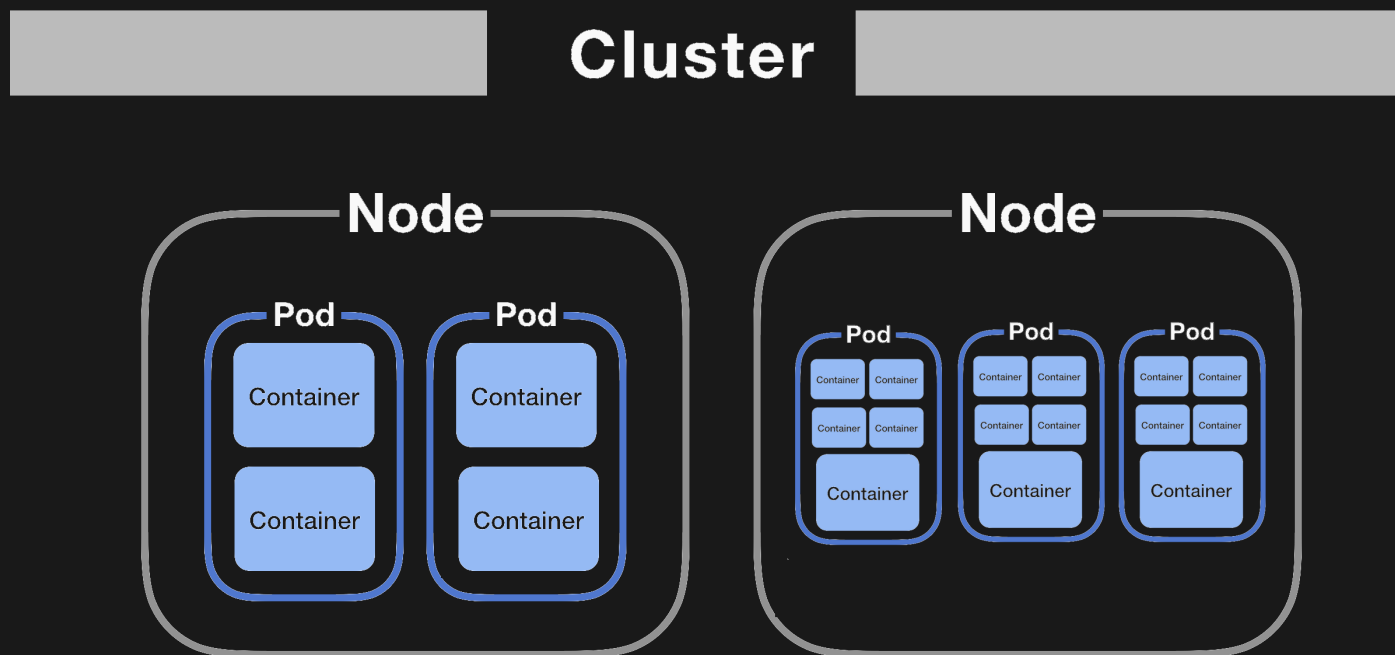
Kubernetes

- What is Kubernetes (aka. K8s)



What is Kubernetes (aka. K8s)

- Container management
 - Deployment, scaling, scheduling, etc.
- Works with Docker, Containerd, and CRI-O, etc.



Code Ocean

- An integrative, collaborative platform for computational research
 - Develop, collaborate, share, and publish code using a web browser (without the need for much specialized knowledge)
 - Similar tools: Digital Ocean, Vultr, Kamatera, Google Cloud/Amazon Web Service, etc.
- Both:
 - a *product* of Docker and K8s *and*
 - an *application* of cloud computing
- **Capsule**
 - Container (using images supplied by CO)
 - Cloud computing + environment + code + (optional) data
- The backend of Code Ocean
 - AWS computing instance: 16 cores, 120 GB of memory
 - Docker for configuring computing environment (user-accessible)
 - K8s for allocating and scheduling resources (not user-accessible)

Exercise

- You can use the example code or create your own capsule for your project.
- Create a Code Ocean account at <https://codeocean.com>
 - `.edu` account comes with 10 hours computing time
- Create a new capsule
- Add `R` (4.1.0) as the base environment
- Install `python` support by adding `python3-pip:latest` to `apt-get`
- Install system dependency using `apt-get`: `libudunits2-dev`, `libgdal-dev`, `libgeos-dev`, `libproj-dev`, `libfontconfig1-dev`.
- Add packages/libraries along with their specific versions
 - Install `beautifulsoup4:4.11.1`, `requests:latest` via `pip3`
 - Install `dplyr:1.0.9`, `tidyr:1.2.0`, `ggplot2:3.4.0`, `sf:latest` via R (CRAN) and `fiftystater` via Github (`wmurphyrd/fiftystater`)
- Upload files
 - Upload code scripts to `/code`
 - Upload data to `/data`
 - The only runtime writable folder is `/results`

(continued on next page ...)

Exercise (cont'd)

- Create a `run` script for running the code and set as file to run

```
#!/usr/bin/env bash
set -ex
mkdir -p ../results/data # make a dir for saving scraped data
mkdir -p ../results/output # make a dir for saving output figs
python3 -u election-2020.py "$@" # running python script
Rscript election-map-2020.R "$@" # running R script for analysis
```

- Edit metadata and readme
- Commit the changes
- Execute a Reproducible Run