

Containers: Vagrant, Docker, Singularity

Dorota Jarecka: djarecka@mit.edu

Brainhack Global 2017 – Boston

Why do we need containers?

- ▶ Science Reproducibility

Why do we need containers?

- ▶ Science Reproducibility
 - ▶ Each project in a lab depends on complex software environments
 - ▶ operating system
 - ▶ drivers
 - ▶ software dependencies: Python/MATLAB/R + libraries

Why do we need containers?

- ▶ Science Reproducibility
 - ▶ Each project in a lab depends on complex software environments
 - ▶ operating system
 - ▶ drivers
 - ▶ software dependencies: Python/MATLAB/R + libraries
- ▶ We try to avoid
 - ▶ *the computer I used was shut down a year ago, can't rerun the results from my publication...*
 - ▶ *the analysis were run by my student, have no idea where and how...*
 - ▶ etc.

Why do we need containers?

- ▶ Collaboration with your colleagues

Why do we need containers?

- ▶ Collaboration with your colleagues
 - ▶ Sharing your code or using a repository might not be enough

Why do we need containers?

- ▶ Collaboration with your colleagues
 - ▶ Sharing your code or using a repository might not be enough
- ▶ We try to avoid
 - ▶ *well, I forgot to mention that you have to use Clang, gcc never worked for me...*
 - ▶ *don't see any reason why it shouldn't work on Windows... (I actually have no idea about Windows, but won't say it...)*
 - ▶ **it works on my computer...**
 - ▶ etc.

Why do we need containers?

- ▶ Freedom to experiment!

Why do we need containers?

- Freedom to experiment!

```
INSTALL.SH  
#!/bin/bash  
  
pip install "$1" &  
easy_install "$1" &  
brew install "$1" &  
npm install "$1" &  
yum install "$1" & dnf install "$1" &  
docker run "$1" &  
pkg install "$1" &  
apt-get install "$1" &  
sudo apt-get install "$1" &  
steamcmd +app_update "$1" validate &  
git clone https://github.com/"$1"/"$1" &  
cd "$1";./configure;make;make install &  
curl "$1" | bash &
```

Universal Install Script from xkcd: *The failures **usually** don't hurt anything...* **Usually** all your old programs work...

Why do we need containers?

- Freedom to experiment!

```
INSTALL.SH  
#!/bin/bash  
  
pip install "$1" &  
easy_install "$1" &  
brew install "$1" &  
npm install "$1" &  
yum install "$1" & dnf install "$1" &  
docker run "$1" &  
pkg install "$1" &  
apt-get install "$1" &  
sudo apt-get install "$1" &  
steamcmd +app_update "$1" validate &  
git clone https://github.com/"$1"/"$1" &  
cd "$1";./configure;make;make install &  
curl "$1" | bash &
```

Universal Install Script from xkcd: *The failures **usually** don't hurt anything... **Usually** all your old programs work...*

- We try to avoid
 - *I just want to Undo the last five hours of my life...*

Virtual Machines and Container Technologies

- ▶ Main idea: isolate the computing environment
 - ▶ Allow regenerating computing environments
 - ▶ Allow sharing your computing environments

Virtual Machines and Container Technologies

- ▶ Main idea: isolate the computing environment
 - ▶ Allow regenerating computing environments
 - ▶ Allow sharing your computing environments
- ▶ Two types:
 - ▶ Virtual Machines
 - ▶ Containers

Virtual Machines and Container Technologies

- ▶ Main idea: isolate the computing environment
 - ▶ Allow regenerating computing environments
 - ▶ Allow sharing your computing environments
- ▶ Two types:
 - ▶ Virtual Machines
 - ▶ Virtualbox
 - ▶ VMware
 - ▶ AWS, Google Compute, ...
 - ▶ Containers

Virtual Machines and Container Technologies

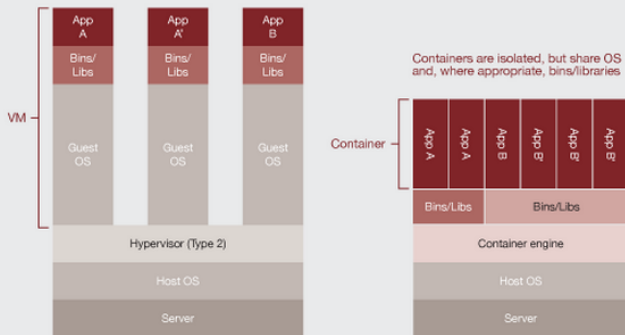
- ▶ Main idea: isolate the computing environment
 - ▶ Allow regenerating computing environments
 - ▶ Allow sharing your computing environments
- ▶ Two types:
 - ▶ Virtual Machines
 - ▶ Virtualbox
 - ▶ VMware
 - ▶ AWS, Google Compute, ...
 - ▶ Containers
 - ▶ Docker
 - ▶ runc
 - ▶ lxc/lxd
 - ▶ Singularity

Virtual Machines and Container Technologies

- ▶ Main idea: isolate the computing environment
 - ▶ Allow regenerating computing environments
 - ▶ Allow sharing your computing environments
- ▶ Two types:
 - ▶ Virtual Machines
 - ▶ Virtualbox
 - ▶ VMware
 - ▶ AWS, Google Compute, ...
 - ▶ Containers
 - ▶ Docker
 - ▶ runc
 - ▶ lxc/lxd
 - ▶ Singularity
- ▶ The details differ (and matter depending on application)

Virtual Machines vs Container

Virtual machines on a Type 2 hypervisor versus application containerization with a shared OS



Vagrant

- ▶ is a command line utility for managing virtual machines
- ▶ provides easy to configure, reproducible, and portable work environments
- ▶ on top of VirtualBox, VMware, AWS, or any other provider (VirtualBox is the default machine)

Vagrant: Creating a computing environments using

- ▶ Using an existing box – ubuntu/trusty64

Vagrant: Creating a computing environments using

- Using an existing box – ubuntu/trusty64

```
$ mkdir vm  
$ cd vm  
$ vagrant init ubuntu/trusty64  
$ vagrant up    # the same as vagrant up --provider virtualbox  
$ vagrant ssh   # or vagrant ssh -c /bin/sh
```

Vagrant: installing new software

- ▶ installing Emacs

Vagrant: installing new software

- ▶ installing Emacs

within the VM:

```
$ sudo apt-get update
$ sudo apt-get install emacs
$ emacs
$ exit
```

- ▶ installing FSL

- ▶ follow the instruction from [NeuroDebian](#) (once it's fixed)

Docker

- ▶ leading software container platform
- ▶ an open-source project
- ▶ bundle libraries and required settings only (not a full operating system)
- ▶ more lightweight to deploy and faster to start up than VM

Docker

- ▶ leading software container platform
 - ▶ an open-source project
 - ▶ bundle libraries and required settings only (not a full operating system)
 - ▶ more lightweight to deploy and faster to start up than VM
-
- ▶ testing your Docker installation:

```
$ docker run hello-world
```

► Using existing images

```
$ docker pull busybox
$ docker images
$ docker run busybox
$ docker run busybox echo "hello from busybox"
$ docker run -it busybox sh
$ docker ps
$ docker rm
$ docker run -it --rm busybox
$ docker run -it --rm -v YourDirectory:/src busybox
```


Docker: Installing software with Dockerfile

- Dockerfile content

```
FROM bids/base_fsl  
RUN apt-get update -y && apt-get install -y r-base
```

Docker: Installing software with Dockerfile

- ▶ Dockerfile content

```
FROM bids/base_fsl  
RUN apt-get update -y && apt-get install -y r-base
```

- ▶ Building a new container

```
$ docker build -t fslR .
```

Docker: Installing software with Dockerfile

- ▶ Dockerfile content

```
FROM bids/base_fsl  
RUN apt-get update -y && apt-get install -y r-base
```

- ▶ Building a new container

```
$ docker build -t fslR .
```

- ▶ Running your new container

```
$ docker run -ti --rm fslR
```

Docker and Nipype

- ▶ Using Nipype official Docker image

Docker and Nipype

- Using Nipype official Docker image

```
$ docker images  
$ docker pull nipype/nipype  
$ docker images  
$ docker run -it --rm nipype/nipype
```

Docker and Nipype

- ▶ Using Nipype official Docker image

```
$ docker images  
$ docker pull nipype/nipype  
$ docker images  
$ docker run -it --rm nipype/nipype
```

- ▶ within the nipype container

```
$ cd /src/nipype/nipype/pipeline/engine/  
$ py.test
```

Docker: Docker Hub

Docker Hub is a cloud-based registry service which allows you to link to code repositories, build your images and share them.

- [Nipype](#)

Docker: Docker Hub

Docker Hub is a cloud-based registry service which allows you to link to code repositories, build your images and share them.

- ▶ Nipype
- ▶ Anaconda - a python distribution

Docker: Docker Hub

Docker Hub is a cloud-based registry service which allows you to link to code repositories, build your images and share them.

- ▶ Nipype
- ▶ Anaconda - a python distribution
- ▶ Miniconda - a package manager

Singularity

- ▶ a container solution created for scientific and application driven workloads.

Singularity

- ▶ a container solution created for scientific and application driven workloads.
- ▶ supports existing and traditional HPC resources

Singularity

- ▶ a container solution created for scientific and application driven workloads.
- ▶ supports existing and traditional HPC resources
- ▶ a user inside a Singularity container is the same user as outside the container

Singularity

- ▶ a container solution created for scientific and application driven workloads.
- ▶ supports existing and traditional HPC resources
- ▶ a user inside a Singularity container is the same user as outside the container
- ▶ can use Vagrant to create containers (you have root privileges on your VM!)

Singularity

- ▶ a container solution created for scientific and application driven workloads.
- ▶ supports existing and traditional HPC resources
- ▶ a user inside a Singularity container is the same user as outside the container
- ▶ can use Vagrant to create containers (you have root privileges on your VM!)
- ▶ can run existing Docker containers

Singularity

- ▶ a container solution created for scientific and application driven workloads.
- ▶ supports existing and traditional HPC resources
- ▶ a user inside a Singularity container is the same user as outside the container
- ▶ can use Vagrant to create containers (you have root privileges on your VM!)
- ▶ can run existing Docker containers
- ▶ Satra's presentation – [Singularity on Openmind](#)