Practical introduction to Singularity containers

November 11, 2021

Tom Harrop
tharrop@unimelb.edu.au

melbournebioinformatics.org.au

@MelBioInf

Reproducibility
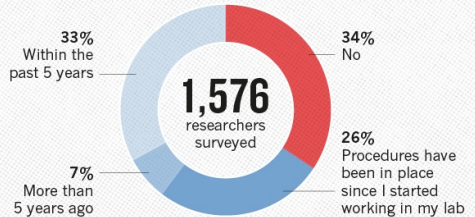
Analysis practices

Guidelines:

1. Don't modify raw data
2. Record the steps
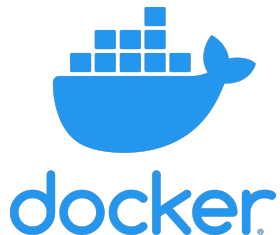   (with version control)
3. Capture the computing environment

- Data management:
  - see Gayle Philip's slides
- Project management, source control:
  - Jessica Chung's slides
- Workflow management:
  - Bobbie Shaban's slides
- Reproducible computing:
  - use containers

Why use containers?
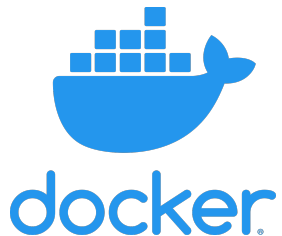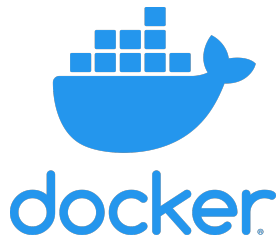

Motivation: Use software without installing it

Why use containers?

Motivation: Use software without installing it

Benefits: Portability, reproducibility

Why use containers?

Motivation:	Use software without installing it
Benefits:	Portability, reproducibility
Drawbacks:	Effort

Use software without installing it

On my computer:

```
# not provided for my OS
guppy_basecaller --version
```

```
guppy_basecaller: command not found
```

Using a container:

```
singularity exec guppy_5.0.16.sif \
    guppy_basecaller --version
```

```
: Guppy Basecalling Software, (C) Oxford Nar
```

Get the version you need

On my computer:

```
# installed by my OS package manager
samtools --version
```

```
samtools 1.11
Using htslib 1.11-4
Copyright (C) 2020 Genome Research Ltd.
```

Using a container:

```
singularity exec samtools_1.12.sif \
    samtools --version
```

```
samtools 1.12
Using htslib 1.12
Copyright (C) 2021 Genome Research Ltd.
```

Avoid tricky software installations

- lots of dependencies
- not all dependencies are listed

Avoid tricky software installations

- lots of dependencies
- not all dependencies are listed
- need specific versions
- package managers don't always help

Avoid tricky software installations

- lots of dependencies
- not all dependencies are listed
- need specific versions
- package managers don't always help
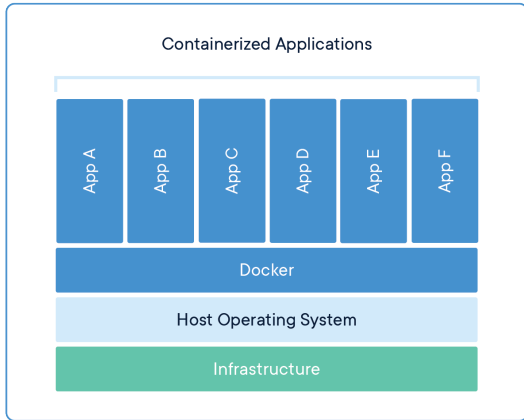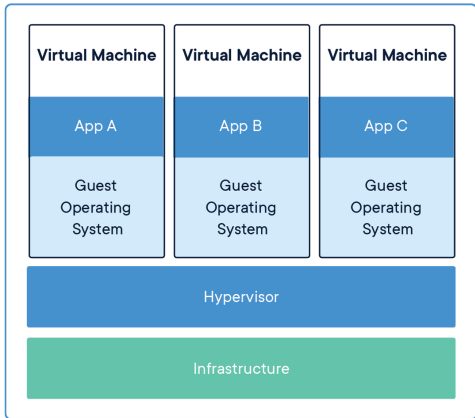
```
singularity run docker://nextgenusfs/funannotate
```

What is a container?
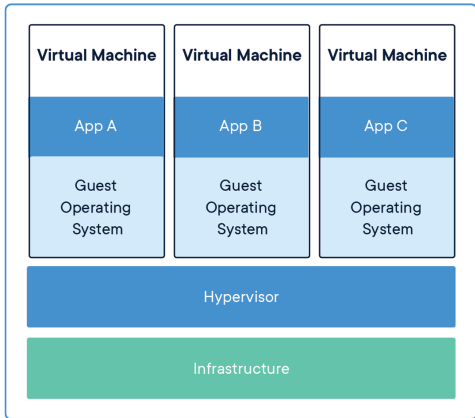
What is a container?

Docker, or Singularity?

- Docker
  - popular on cloud platforms
  - effectively requires root to run*
  - doesn't respect job schedulers*

* may have been fixed

Docker, or Singularity?

- Docker
  - popular on cloud platforms
  - effectively requires root to run*
  - doesn't respect job schedulers*

* may have been fixed

- Singularity
  - runs on HPC
  - popular in research / edu?
  - **supports Docker containers**

Docker, or Singularity?

- Docker
  - popular on cloud platforms
  - effectively requires root to run*
  - doesn't respect job schedulers*

- Singularity
  - runs on HPC
  - popular in research / edu?
  - **supports Docker containers**

* may have been fixed

Suggested workflow: build (or obtain) your containers in Docker format and use Singularity to run them

How to use Singularity

1. Install Singularity
- not too hard to install locally
- HPCs often have it installed
- you can ask the admin to install it

e.g. on Spartan:

```
$ module load singularity/3.7.3
$ singularity --version
singularity version 3.7.3
```

How to use Singularity

1. Install Singularity
- not too hard to install locally
- HPCs often have it installed
- you can ask the admin to install it

e.g. on Spartan:

```
$ module load singularity/3.7.3
$ singularity --version
singularity version 3.7.3
```

2. Run a container
- detailed instructions at
  singularity.hpcng.org
- basic commands is
  ```
  singularity exec {container URL} \
  {command} {args}
  ```
- usually need some options, *e.g.*
  - -B **b**inds paths to make files
    available in the container
  - --nv enables GPU support

Workflow managers support containers
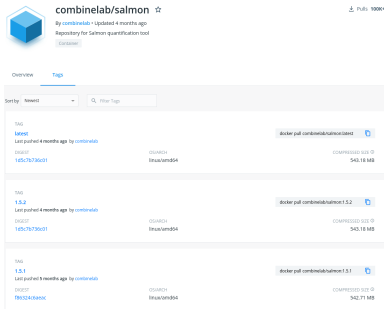
Snakefile:

```
rule trim_adaptors:
    input:          'data/raw_reads.fastq'
    output:         'output/trimmed.fastq'
    container:      'docker://my_repos/trim_adaptors:1.7'
    shell:          'trim_adaptors --raw_reads={input} > {output}'
rule run_assembly:
    input:          'output/trimmed.fastq'
    output:         'output/assembly.fasta'
    container:      'docker://my_repos/tidy_assembler:2.9'
    shell:          'tidy_assembler --reads={input} > {output}'
```

Workflow managers support containers

Snakefile:

```
rule trim_adaptors:
    input:           'data/raw_reads.fastq'
    output:          'output/trimmed.fastq'
    container:       'docker://my_repos/trim_adaptors:1.7'
    shell:           'trim_adaptors --raw_reads={input} > {output}'
rule run_assembly:
    input:            'output/trimmed.fastq'
    output:           'output/assembly.fasta'
    container:        'docker://my_repos/tidy_assembler:2.9'
    shell:            'tidy_assembler --reads={input} > {output}'
```

Run the workflow:

```
snakemake --use-singularity \
    --singularity-args ...
```

Where to find containers

🌐 Some developers provide docker containers

```
singularity pull \
    docker://combinelab/salmon:1.1.0
```

You can search for images on hub.docker.com

Check with the software developer

Build them yourself!

Build them yourself!

Some barriers to container usage

- Building containers yourself can be painful
- Duplication of effort
- Some software **shouldn't** go in a container because of "unfortunate licensing issues"
  - DTU software *e.g.* `rnammer`, `tmhmm`
  - GATech: `GeneMark`
  - GIRInst's `RepBase`
- Getting `Singularity` installed

When to use containers

Modules:

Conda, venv, *etc.*:

Containers:

When to use containers

Modules:

```
module load \
    samtools/1.12
```

- installed by the sysadmin
- properly configured
- easy

Conda, venv, *etc.*:

Containers:

When to use containers

Modules:

```
module load \
    samtools/1.12
```

- installed by the sysadmin
- properly configured
- easy

Conda, venv, *etc.*:

```
conda install -c \
    bioconda samtools
```

- latest packages
- testing in a specific environment

Containers:

When to use containers

Modules:

```
module load \
    samtools/1.12
```

- installed by the sysadmin
- properly configured
- easy

Conda, venv, *etc.*:

```
conda install -c \
    bioconda samtools
```

- latest packages
- testing in a specific environment

Containers:

```
singularity run \
    samtools.sif
```

- reproducibility
- collaboration
- portability
- performance?

Get started with containers

- How I use containers:
    1. **Write Docker recipes** (one tool, one recipe)
    2. Remotely build Docker image store it permanently
    3. Use with Singularity via a workflow manager

- Singularity has good documentation: quick start
- Docker documentation is ... extensive: Best practices for writing Dockerfiles
- Integrating with workflow managers is easy