



University of
St Andrews | FOUNDED
1413 |

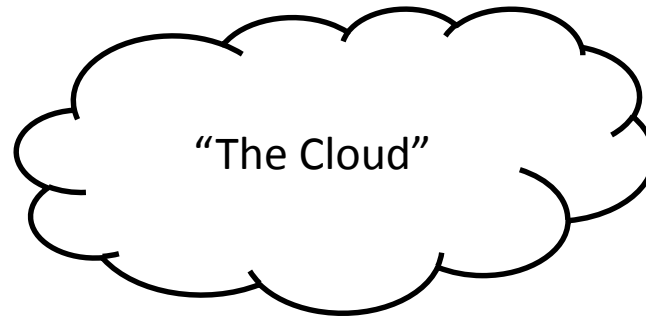
An introduction to Docker

Bert Vandenbroucke

bv7@st-andrews.ac.uk

Problem introduction

Where does your application run?



Problem introduction

Different systems have:

- different operating systems



- different hardware



- different compilers/libraries...



How do you make sure that an application/workflow that runs on your computer will run on another computer?

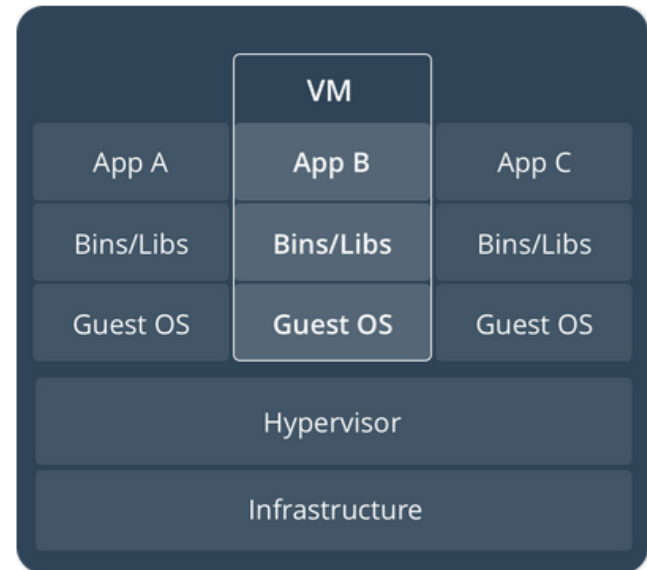
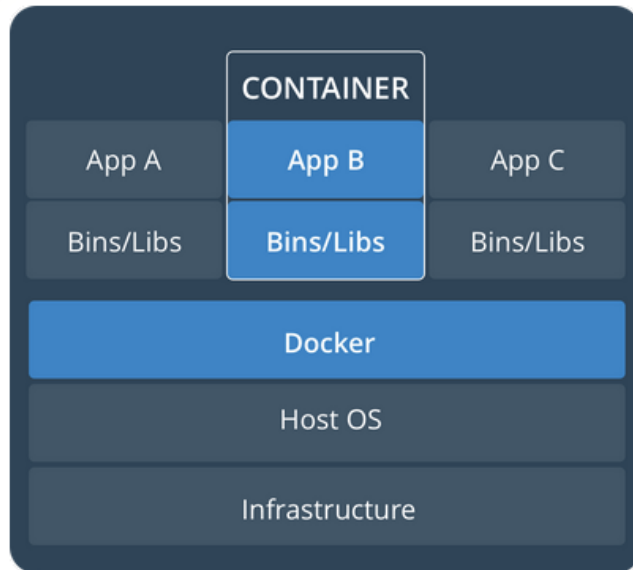
Traditional solution

Test on other systems:

- install a dual boot system, buy a new laptop...
- run another system in VirtualBox (system emulation)
- manually install different compilers, libraries... and figure out how to use/link them correctly

New solution

Container: isolated environment that runs within your operating system



Containers: advantages

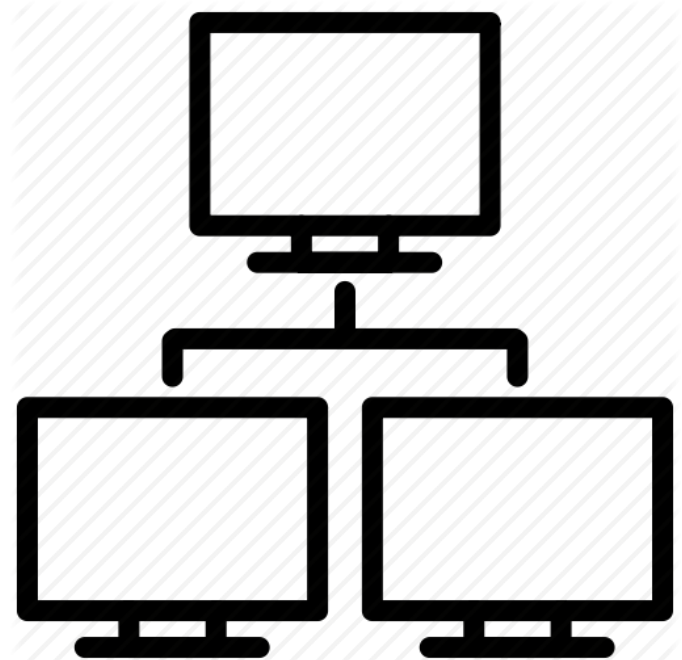
- Lightweight: only relevant libraries and applications, no complete OS
- Portable: can be saved (image) and loaded on another system
- Isolated from host OS
- Large number of images (100,000+) in online container registries on hub.docker.com

Use case 1: remote analysis

Create analysis scripts and pipeline on your own computer

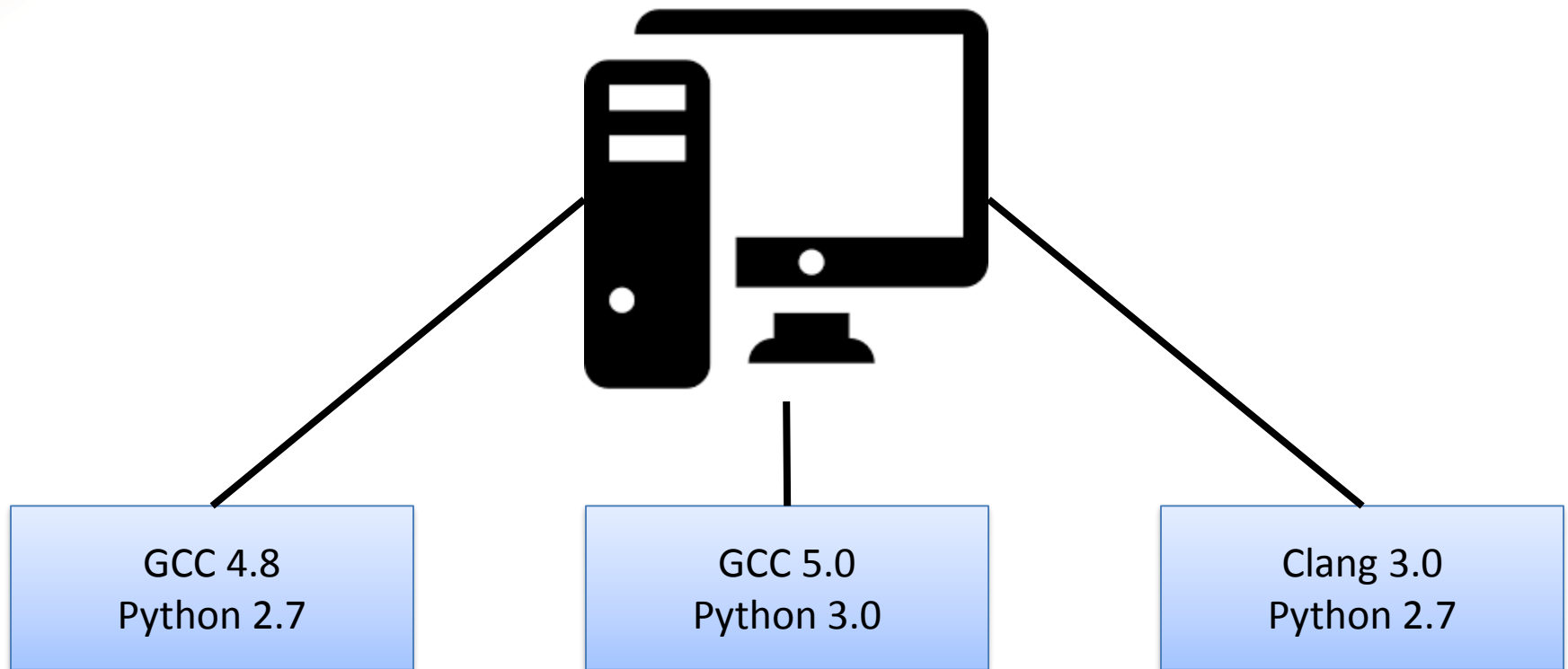


Run the actual analysis in the same environment on a large cluster



Use case 2: code testing

Check that code works with different compilers and libraries on the same computer

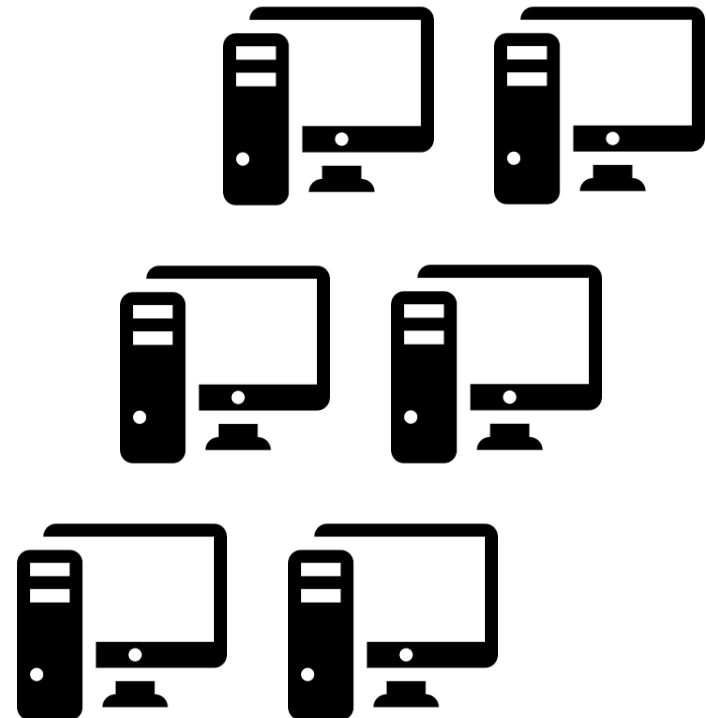


Use case 3: student labs

Create lab scripts
on your own
computer

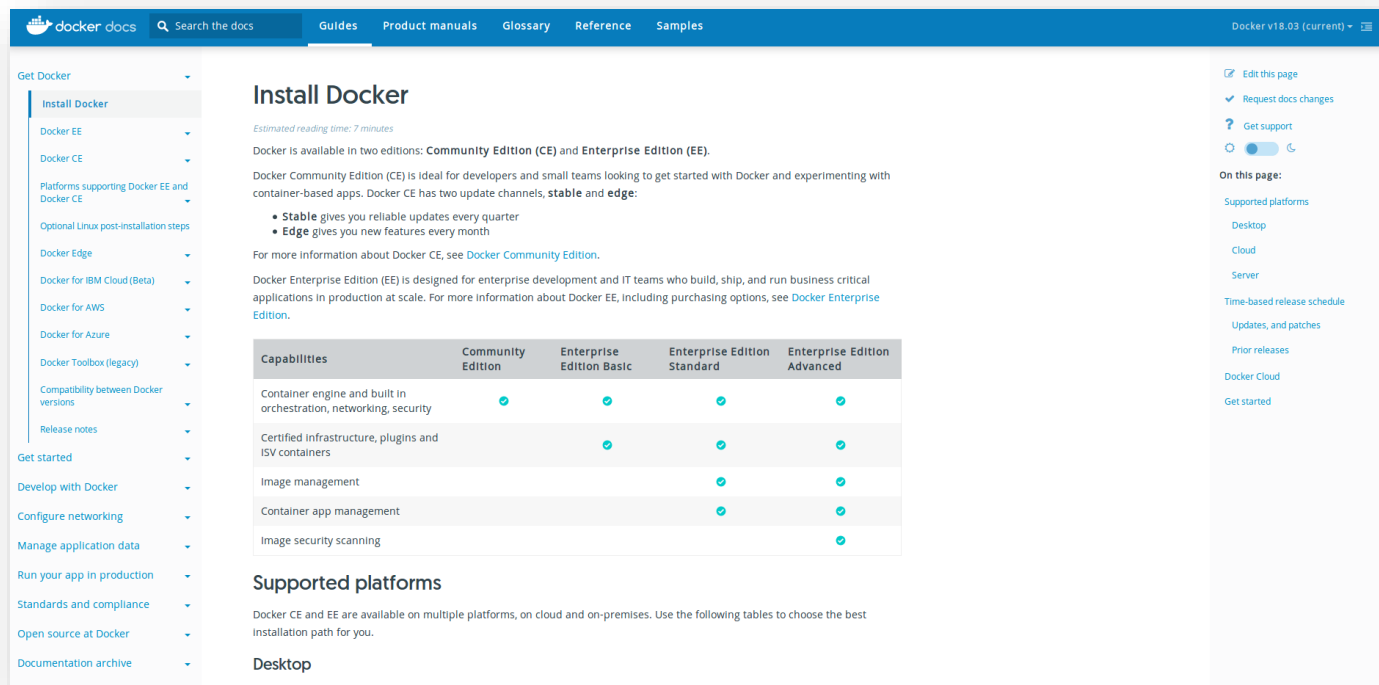


Students do the lab in
the same environment



Docker setup

Detailed instructions for various platforms available: <https://docs.docker.com/install/>



The screenshot shows the Docker documentation website. The main heading is "Install Docker" with an estimated reading time of 7 minutes. The page is divided into sections for Docker Community Edition (CE) and Docker Enterprise Edition (EE). A table compares capabilities across four editions: Community Edition, Enterprise Edition Basic, Enterprise Edition Standard, and Enterprise Edition Advanced. The table shows that CE and EE Basic support container engine and built-in orchestration, networking, and security. EE Standard and EE Advanced support certified infrastructure, plugins, and ISV containers. EE Standard and EE Advanced support image management. EE Advanced supports container app management. EE Advanced also supports image security scanning.

Capabilities	Community Edition	Enterprise Edition Basic	Enterprise Edition Standard	Enterprise Edition Advanced
Container engine and built in orchestration, networking, security	✓	✓	✓	✓
Certified infrastructure, plugins and ISV containers		✓	✓	✓
Image management			✓	✓
Container app management			✓	✓
Image security scanning				✓

Docker overview

Creating a container

Loading and running a container

Saving a container

Docker overview

Creating a container

Loading and running a container

Saving a container

Creating a container

A new container is always based on an existing *base container*:

- contains basic OS binaries
- contains basic libraries
- can contain specific libraries (e.g. specific Python version)
- should be available from an online *registry*

Creating a container

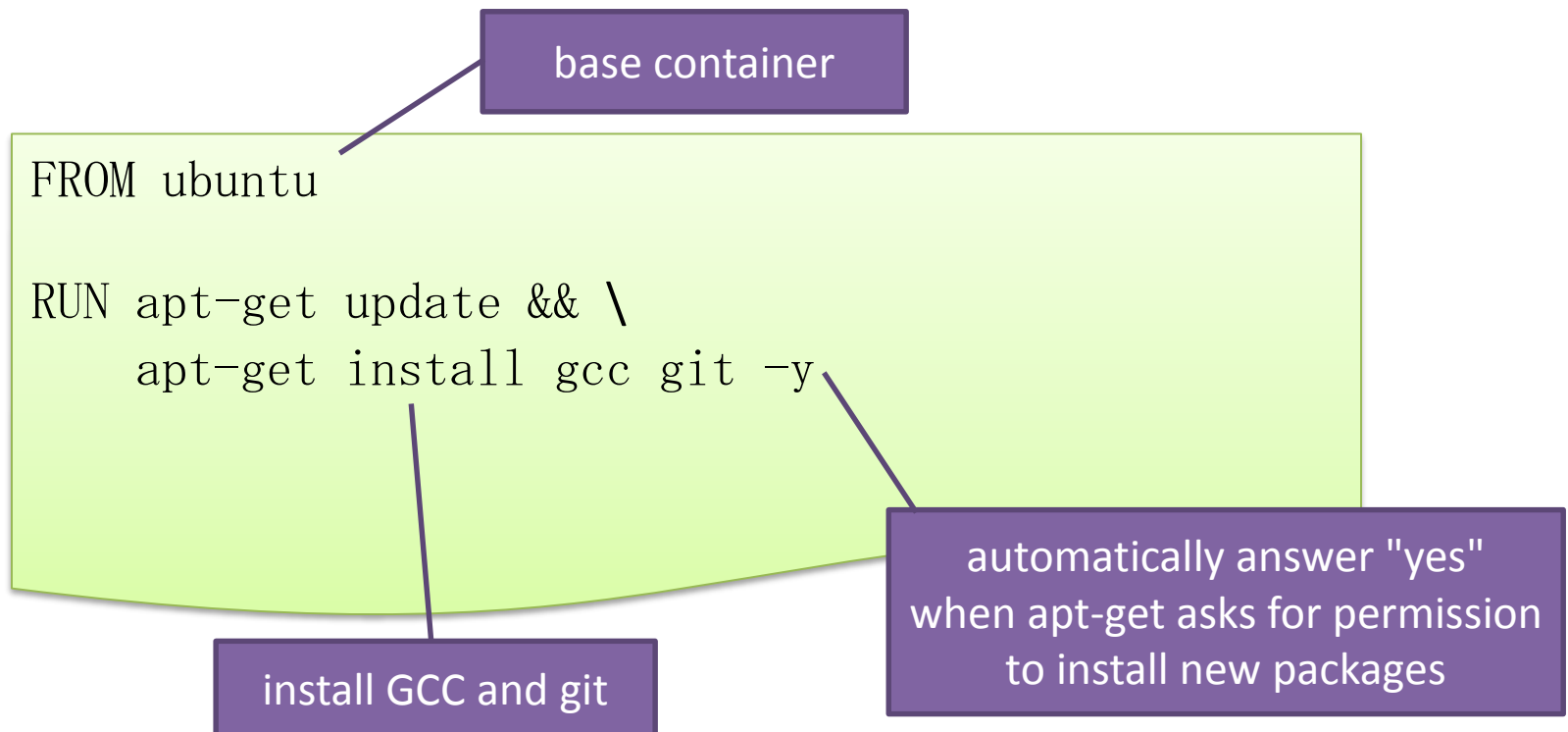
Basic command: `docker build`

Requires a `Dockerfile` to be present:

- specifies a base container
- sets up new libraries and applications
- creates custom folders and files
- copies files from the host to the container

Creating a container

Example Dockerfile that sets up a default code development environment based on Ubuntu



Creating a container

```
Terminal
bv7@apsient:/data/bv7/docker_test/demo$ docker build .
Sending build context to Docker daemon 2.048kB
Step 1/2 : FROM ubuntu
---> f975c5035748
Step 2/2 : RUN apt-get update && apt-get install gcc git -y
---> Using cache
---> 26c860824c1a
Successfully built 26c860824c1a
bv7@apsient:/data/bv7/docker_test/demo$
```

*first run of command will produce more output

Creating a container

Check that image was created

```
Terminal
bv7@apsient:/data/bv7/docker_test/demo$ docker images
REPOSITORY          TAG                 IMAGE ID            CREATED
SIZE
<none>              <none>             26c860824c1a      3 minutes ago
340MB
ubuntu              latest             f975c5035748      3 weeks ago
112MB
swift_exec          latest             8fe40c8179ba      6 weeks ago
505MB
<none>              <none>             e4662b15e1af      6 weeks ago
503MB
swift_repo          latest             861eecad1e26      6 weeks ago
450MB
swift_base          latest             cb21474c8a62      6 weeks ago
383MB
<none>              <none>             1c43bda62963      6 weeks ago
525MB
<none>              <none>             85273bc884fc      6 weeks ago
489MB
<none>              <none>             801b2d307b6e      6 weeks ago
472MB
<none>              <none>             f318a68eee96      6 weeks ago
472MB
```

Docker overview

Creating a container

Loading and running a container

Saving a container

Loading and running a container

Host system: old GCC version

Run an interactive docker container based on the image we created

```
root@c6386563f8ac: /  
bv7@apsient:/data/bv7/docker_test/demo$ gcc --version  
gcc (Ubuntu 4.8.4-2ubuntu1~14.04.4) 4.8.4  
Copyright (C) 2013 Free Software Foundation, Inc.  
This is free software; see the source for copying conditions. There is NO  
warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  
  
bv7@apsient:/data/bv7/docker_test/demo$ docker run -t -i --rm 26c860824c1a bash  
root@c6386563f8ac:/# gcc --version  
gcc (Ubuntu 5.4.0-6ubuntu1~16.04.9) 5.4.0 20160609  
Copyright (C) 2015 Free Software Foundation, Inc.  
This is free software; see the source for copying conditions. There is NO  
warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  
  
root@c6386563f8ac:/#
```

Attach a pseudo-shell to the container

Container: new GCC version

Clean up when we're done

Loading and running a container

List running containers

```
root@c6386563f8ac: /  
bv7@apsient:/data/bv7/docker_test/demo$ docker container ls  
CONTAINER ID        IMAGE               COMMAND             CREATED  
STATUS              PORTS              NAMES                
c6386563f8ac       26c860824c1a      "bash"             5 minutes ago  
Up 5 minutes                xenodochial_ramanujan  
bv7@apsient:/data/bv7/docker_test/demo$ docker exec -it c6386563f8ac bash  
root@c6386563f8ac:/#
```

Connect a second shell to the running container in another terminal window

Docker overview

Creating a container

Loading and running a container

Saving a container

Saving a container

You can create a new image from any *running* container using `docker commit`

This creates a new image that contains all changes made in the container since it was started (and overwrites the existing image)

If you don't save, all changes are lost!

Saving a container

```
root@c6386563f8ac: /
bv7@apsient:/data/bv7/docker_test/demo$ docker commit cc5a903ddead
sha256:ce68c24f665392ff6b895f4edbe15d63bf1c6ec025d4605833cc82d0827049cc
bv7@apsient:/data/bv7/docker_test/demo$ docker images
```

REPOSITORY	TAG	IMAGE ID	CREATED
<none> 341MB	<none>	ce68c24f6653	3 seconds ago
<none> 341MB	<none>	664cc5456023	2 minutes ago
ubuntu 112MB	latest	f975c5035748	3 weeks ago
swift_exec 505MB	latest	8fe40c8179ba	6 weeks ago
<none> 503MB	<none>	e4662b15e1af	6 weeks ago
swift_repo 450MB	latest	861eecad1e26	6 weeks ago
swift_base 383MB	latest	cb21474c8a62	6 weeks ago
<none> 525MB	<none>	1c43bda62963	6 weeks ago
<none> 489MB	<none>	85273bc884fc	6 weeks ago
<none>	<none>	801b2d307b6e	6 weeks ago

Saving a container

Images can be converted into tar files using `docker save`

These tar files can be copied to other systems to run remote containers (using `docker load`)

Alternatively, you can publish your container in an online registry using `docker push`

Summary

- Containers are a lightweight alternative for system emulation
- Easy to use
- Container images can be ported to other systems/hardware...

- EXTRA: container support in workflow management systems (see previous talk)

More information

A huge body of documentation can be found on
<https://docs.docker.com/>

But the easiest way to learn Docker is using it!