

Using Docker in BioHPC Cloud

Jaroslav Pillardy
Bioinformatics Facility

4/27/2020

What is Docker?

A Linux subsystem to run isolated Linux “machines” called “containers”

Isolated means that programs, users and storage in Docker “machine” are separated from the host system.

It can run any flavor of Linux on any Linux machine

What is Docker?

Programs installed inside Docker container don't need to be compatible with the host, and in fact they can be installed from scratch, regardless of what is installed on the host.

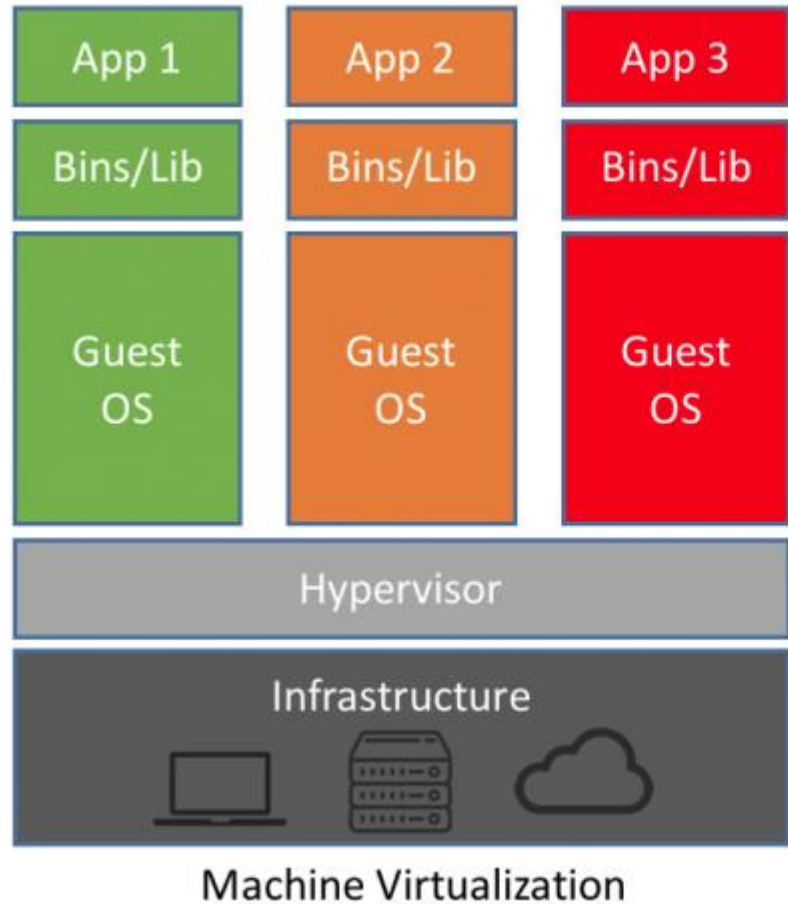
Regular user can become "root" (Linux administrator) inside Docker container, or any user as needed.

Docker components

image - a template that can be loaded into Docker and executed. Image can be stored on a disk as a file or in a specialized Docker repository of images

container - a running instance of Docker image – actual Docker “machine”. Users can execute programs, install software and work in container as in a regular Linux system.

Virtual Machine vs Docker Containers



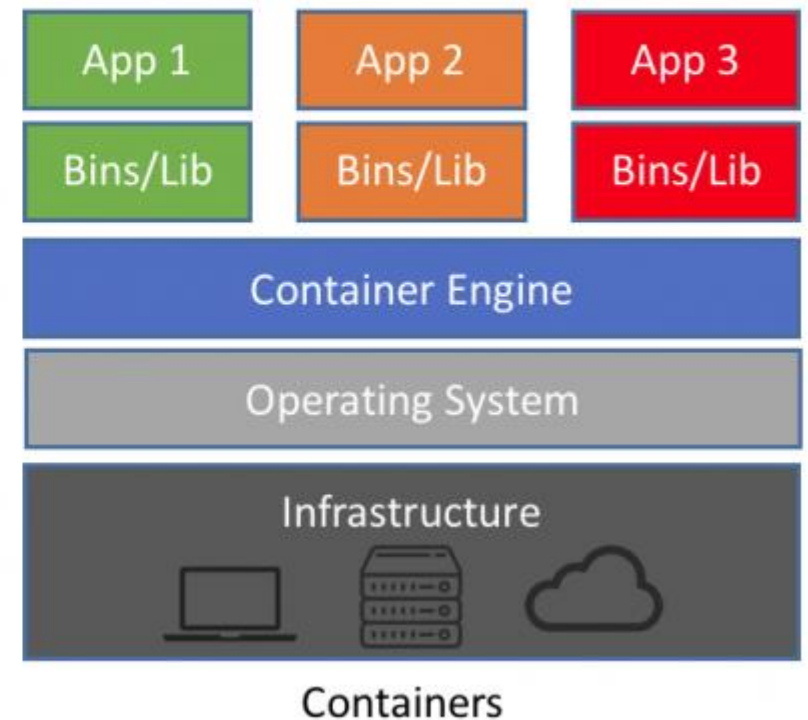
Virtual Machine (VM) is a separated “server” emulating its own hardware and running its own operating system. The only interaction with host OS is via hardware or emulated hardware.

VM can run any OS but is slower and requires extra resources to run. Up to 20-30% overhead.

Virtual Machine vs Docker Containers

Docker shares host OS kernel services and some libraries (read-only). It runs as a process in host operating system. It can access host files directly (optional).

No execution overhead – same speed. Cannot run non-Linux OS.



Docker security

In order to use original Docker user must have “root” access to certain parts of Linux OS.

It is safe for admins to deploy software, but NOT safe for users in multi-user environment like BioHPC

We have developed our own version of Docker on the top of original Docker that addresses security problems at the same time preserving most of Docker features.

BioHPC Docker

Original Docker command is "docker". This command has been replaced by "docker1" command in BioHPC Lab.

Whenever reading a Docker book or website please replace "docker" with "docker1" when you want to run the command on BioHPC Lab machines.

BioHPC Docker

If you run "docker" instead of "docker1" you will get an error. You have to use "docker1"

```
[jarekp@cbsum1c2b014 ~]$ docker ps -a
Cannot connect to the Docker daemon. Is the docker daemon running on this host?
[jarekp@cbsum1c2b014 ~]$
```

```
[jarekp@cbsum1c1b009 ~]$ docker ps -a
Got permission denied while trying to connect to the Docker daemon socket at
unix:///var/run/docker.sock: Get
http://%2Fvar%2Frun%2Fdocker.sock/v1.26/containers/json?all=1: dial unix
/var/run/docker.sock: connect: permission denied
```

BioHPC Docker

You can check docker1 options with “docker1 --help” or “docker1 commandname --help”

```
[jarekp@cbsum1c1b009 ~]$ docker1 --help
```

This is BioHPC Cloud docker1 implementing secure Docker environment.

Some Docker commands have been modified or disabled, but most are unchanged. There are three additional commands in docker1 listed below

clean Deletes sets of containers

claim Changes ownership of all files and dirs in /workdir/labid and /SSD/labid to labid

white Displays set of options that are whitelisted to use with docker1

Special option --noworkdir for run command disables automatic mapping of /workdir and /SSD

Please note you need to put docker command options BEFORE container name where applicable

Docker help page follows.

Usage: docker COMMAND

A self-sufficient runtime for containers

Docker images

Before running any Dockerized application you need to know how to access its Docker image.

- Images are stored in Docker registries (or hubs) and their names and addresses are described in software documentation.

```
docker1 pull docker.io/biohpc/imagename
```

BioHPC image

```
docker1 pull docker.io/imagename
```

public image

- Image can be imported from a file

```
docker1 load -i filename
```

```
docker1 import filename
```

docker.io/ part may be omitted, repositories known to this Docker installation will be queried in order

Docker images: load vs import

- `docker1 load -i filename`

Docker load command creates a container from **saved image**, it imports all the image layers, tags and settings. File for load command must be created with save command.

- `docker1 import filename`

Docker import command creates a container from **saved container**, it creates a simplified image based on the saved container with a single layer and no extra settings (like entry point). File for import command must be created with export command

BioHPC Docker

To run a container:

- Pull or import/load image
- Start container from image

You can also try to start container without pulling first, Docker will pull it if not found locally. All images used on a server are cached in a local registry.

BioHPC Docker

Pull image

docker.io/ part may be omitted,
repositories known to this Docker
installation will be queried in order

```
[jarekp@cbsum1c1b009 ~]$ docker1 pull biohpc/cowsay
Using default tag: latest
Trying to pull repository dtr.cucloud.net/biohpc/cowsay ...
Trying to pull repository docker.io/biohpc/cowsay ...
sha256:c4c5c72fd2e09e35a02199aef774f5338ab6a17f204ed7317f11a58f8d1ad284: Pulling from
docker.io/biohpc/cowsay
9e4b184d5239: Pull complete
4a28c3c49d95: Pull complete
Digest: sha256:c4c5c72fd2e09e35a02199aef774f5338ab6a17f204ed7317f11a58f8d1ad284
Status: Downloaded newer image for docker.io/biohpc/cowsay:latest
```

```
[jarekp@cbsum1c1b009 ~]$ docker1 images
```

REPOSITORY	TAG	IMAGE ID	CREATED	SIZE
docker.io/biohpc/cowsay	latest	195f168235c9	3 years ago	337 MB

List images

BioHPC Docker

Import image

```
[jarekp@cbsum1c2b014 ~]$ docker1 import /programs/docker/images/cowsay.tar  
sha256:da8e563445a8792ae5b161b446e8ef9ca2c76f2bafab58ad88bf0adcbfb5d0b0
```

```
[jarekp@cbsum1c2b014 ~]$ docker1 images
```

REPOSITORY	TAG	IMAGE ID	CREATED	SIZE
biohpc_jarekp/cowsay	latest	da8e563445a8	About a minute ago	319.7 MB
docker.io/biohpc/cowsay	latest	195f168235c9	16 months ago	337.1 MB

```
[jarekp@cbsum1c2b014 ~]$
```

What if we try to load a file created with export?

```
[jarekp@cbsumm15 docker]$ docker1 load -i /programs/docker/images/cowsay.tar  
open /local/docker/tmp/docker-import-755479071/dev/json: no such file or directory  
[jarekp@cbsumm15 docker]$
```

BioHPC Docker

Run Docker container

- **Single command**
Run a command and then container stops.
- **Interactive mode**
Open shell inside container for interactive work. Once you are finished, exit shell and container stops.
- **Background mode**
Start container in the background and connect to it when needed.
Container will keep running.

BioHPC Docker

Single command run

image name command command arguments

```
[jarekp@cbsum1c2b014 ~]$ docker1 run biohpc/cowsay cowsay "This is Docker"
```

```
< This is Docker >
```

```
-----
```

```
  \      ^  ^  
   \    (oo)\_____  
      (__)\\       )\\/\  
         ||----w |  
         ||     ||
```

List containers on this host, -a lists both active and inactive

```
[jarekp@cbsum1c2b014 ~]$ docker1 ps -a
```

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NAMES
1035e0f20e5f	biohpc/cowsay	"cowsay 'This is Dock"	16 seconds ago	Exited (0) 13 seconds ago		jarekp__biohpc_1

Image exited – cannot connect to it anymore

BioHPC Docker

Interactive run

run interactively image name command to run

```
[jarekp@cbsum1c2b014 ~]$ docker1 run -it biohpc/cowsay /bin/bash
```

```
[root@a0017f5faf51 workdir]# pwd
```

```
/workdir
```

```
[root@a0017f5faf51 workdir]# cowsay "hi"
```

```
< hi >
```

```
----
```

```
  \      ^__^
   \      (oo)\_______
      (__)\       )\/\
         ||----w |
         ||     ||
```

```
[root@a0017f5faf51 workdir]# exit
```

```
exit
```

```
[jarekp@cbsum1c2b014 ~]$ docker1 ps -a
```

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NAMES
a0017f5faf51	biohpc_jarekp/cowsay	"/bin/bash"	32 seconds ago	Exited (0) 6 seconds ago		jarekp__biohpc_2
1035e0f20e5f	biohpc/cowsay	"cowsay 'This is Dock'"	6 minutes ago	Exited (0) 6 minutes ago		jarekp__biohpc_1

```
[jarekp@cbsum1c2b014 ~]$
```

we are inside container now – as root!
Default directory is /workdir

Image exited – cannot connect to it anymore

BioHPC Docker

Run in the background

run in background image name command to run

```
[jarekp@cbsum1c2b014 ~]$ docker1 run -d -t biohpc/cowsay /bin/bash  
10af80003f76940b154a176af4a3b3747647763c2a3eb62b27f9e442cad7060f
```

Image running

```
[jarekp@cbsum1c2b014 ~]$ docker1 ps -a
```

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NAMES
10af80003f76	biohpc/cowsay	"/bin/bash"	7 seconds ago	Up 4 seconds		jarekp__biohpc_3
a0017f5faf51	biohpc_jarekp/cowsay	"/bin/bash"	7 minutes ago	Exited (0) 7 minutes ago		jarekp__biohpc_2
1035e0f20e5f	biohpc/cowsay	"cowsay 'This is Dock'"	13 minutes ago	Exited (0) 13 minutes ago		jarekp__biohpc_1

```
[jarekp@cbsum1c2b014 ~]$ docker1 exec 10af80003f76 cowsay "hi"
```

```
< hi >  
-----  
 \      ^__^  
  (oo)\_____  
   (__)\       )\/\  
    ||----w |  
    ||     ||
```

BioHPC Docker

Run in the background

```
[jarekp@cbsum1c2b014 ~]$ docker1 exec -it 10af80003f76 /bin/bash
```

```
[root@10af80003f76 workdir]# ls -al
```

```
total 0
```

```
drwxr-xr-x  2 4965 root   6 May 22 17:26 .
```

```
drwxr-xr-x 18 root root 288 May 22 18:48 ..
```

```
[root@10af80003f76 workdir]# exit
```

```
exit
```

```
[jarekp@cbsum1c2b014 ~]$ docker1 ps -a
```

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NAMES
10af80003f76	biohpc/cowsay	"/bin/bash"	About a minute ago	Up About a minute		jarekp__biohpc_3
a0017f5faf51	biohpc_jarekp/cowsay	"/bin/bash"	9 minutes ago	Exited (0) 8 minutes ago		jarekp__biohpc_2
1035e0f20e5f	biohpc/cowsay	"cowsay 'This is Dock'"	14 minutes ago	Exited (0) 14 minutes ago		jarekp__biohpc_1

```
[jarekp@cbsum1c2b014 ~]$
```

Run Docker container

- **Single command**

Run a command and then container stops.

```
docker1 run image command arguments
```

- **Interactive mode**

Open shell inside container for interactive work. Once you are finished, exit shell and container stops.

```
docker1 run -it image /bin/bash
```

- **Background mode**

Start container in the background and connect to it when needed. Container will keep running.

```
docker1 run -d -t image /bin/bash
```

```
docker1 exec container_id command arguments
```

```
docker1 exec -it container_id /bin/bash
```

BioHPC Docker

stop running container

```
docker1 stop container_id_or_name
```

remove (erase) container

```
docker1 rm container_id_or_name
```

Typically id (e.g. 10af80003f76) or name

(e.g. jarekp__biohpc_1) may be used as parameter for stop, rm, exec

BioHPC Docker – cleaning leftovers

Remove all current user non-running containers

```
docker1 clean
```

Remove all current user containers – running or not

```
docker1 clean all
```

Automatically remove my container after exit

```
docker1 run --rm image command arguments
```

Docker repositories

Pull container from BioHPC repository

```
docker1 pull docker.io/biohpc/imagename
```

Pull container from Docker public repository

```
docker1 pull docker.io/imagename
```

```
docker1 pull imagename
```


BioHPC Docker – volumes and directories

By default `/workdir/labid` is mapped to `/workdir` inside the container.

You can skip that by using `--noworkdir` option (in `docker1 run` command)

BioHPC Docker – volumes and directories

It is possible to map other directories from host machine to the container inner file system.

The host directory to be mapped must be owned by you and it must be under /workdir/labid/, /local/storage/ or /fs/servername/storage/, /SSD/labid (replace labid with your BioHPC user id)

```
docker1 run -d -t -v /workdir/jarekp/data:/data biohpc/cowsay --noworkdir /bin/bash
```

Building images from dockerfiles

Docker images can be built using a list of commands stored in file called *dockerfile*

Dockerfile below will create a CentOS 7 image with basic dev tools (gcc), ssh and wget.

```
FROM centos:7
RUN yum -y install gcc
RUN yum -y install openssh-clients
RUN yum -y install wget
```

Building images from dockerfiles

To build the image you need to use `docker1 build` command, with full path to the dockerfile and dockerfile directory. The dockerfile can only be under `/workdir/labid`. Docker build command points to directory where dockerfile resides, if multiple dockerfiles are present additional `-f` options specifies which one to use.

```
docker1 build -t my_centos7_dev /workdir/jarekp/build
```

```
[jarekp@cbsumm15 build]$ docker1 images
```

REPOSITORY	TAG	IMAGE ID	CREATED	SIZE
biohpc_jarekp/my_centos7_dev	latest	b4ecded706c4	10 seconds ago	530 MB

```
docker1 save -o /workdir/my_centos7_dev.tar biohpc_jarekp/my_centos7_dev
```

Building images from dockerfiles

Dockerfiles seem to be a great way to store images in an easy and human readable way.

In reality there are a lot of problems. Each time you create an image from dockerfile it will install software from repositories, which may not be available anymore, or software versions may have changed, so they don't work together anymore.

If you have a good working image, save it! Dockerfile is a good way to document how it was created.

BioHPC Docker Example: install TopHat

1. Start Ubuntu container in background
2. Connect to the container and install TopHat. Check online “how to install TopHat in Ubuntu”.
3. Verify it runs, save output in /workdir, exit container
4. How about the resulting files? Where are they and how to get them?
5. Save the container as image for future use

BioHPC Docker Example: install TopHat

1. Start Ubuntu container in background

```
docker1 run -d -it docker.io/biohpc/ubuntudev /bin/bash
```

```
[jarekp@cbsum1c2b014 ~]$ docker1 run -d -it docker.io/biohpc/ubuntudev /bin/bash
Unable to find image 'docker.io/biohpc/ubuntudev:latest' locally
Trying to pull repository docker.io/biohpc/ubuntudev ...
sha256:d03a0a1e2247895a3e57aa8cb5cdf1c6253f759b3daba78846af1d825e19cb75: Pulling from
docker.io/biohpc/ubuntudev
```

```
87ad106e166e: Pull complete
```

```
Digest: sha256:d03a0a1e2247895a3e57aa8cb5cdf1c6253f759b3daba78846af1d825e19cb75
```

```
Status: Downloaded newer image for docker.io/biohpc/ubuntudev:latest
```

```
ee4d845bb1633c3fe907fd3a2b217f5285bd3cfa9ca24955e4720c39f4eb8e67
```

```
[jarekp@cbsum1c2b014 ~]$ docker1 ps -a
```

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS
ee4d845bb163	docker.io/biohpc/ubuntudev	"/bin/bash"	15 minutes ago	Up 15 minutes

```
jarekp__biohpc_1
```

```
[jarekp@cbsum1c2b014 ~]$
```

BioHPC Docker Example: install TopHat

2. Connect to the container and install TopHat

```
docker1 exec -it containerid /bin/bash
```

```
apt-get update
```

```
apt-get install tophat
```

```
[jarekp@cbsum1c2b014 ~]$ docker1 exec -it ee4d845bb163 /bin/bash
root@ee4d845bb163:/workdir# apt-get update
Hit:1 http://archive.ubuntu.com/ubuntu bionic InRelease
Get:2 http://security.ubuntu.com/ubuntu bionic-security InRelease [88.7 kB]
Get:3 http://archive.ubuntu.com/ubuntu bionic-updates InRelease [88.7 kB]
Get:4 https://cloud.r-project.org/bin/linux/ubuntu bionic-cran35/ InRelease [3626 B]
[...]
Get:11 http://security.ubuntu.com/ubuntu bionic-security/multiverse amd64 Packages [8213 B]
Get:12 http://archive.ubuntu.com/ubuntu bionic-updates/multiverse amd64 Packages [12.6 kB]
Get:13 http://archive.ubuntu.com/ubuntu bionic-updates/universe amd64 Packages [1372 kB]
Get:14 http://archive.ubuntu.com/ubuntu bionic-updates/restricted amd64 Packages [59.0 kB]
Get:15 http://archive.ubuntu.com/ubuntu bionic-backports/universe amd64 Packages [7674 B]
Get:16 http://archive.ubuntu.com/ubuntu bionic-backports/main amd64 Packages [8286 B]
Fetched 4770 kB in 2s (2952 kB/s)
Reading package lists... Done
root@ee4d845bb163:/workdir#
```


BioHPC Docker Example: install TopHat

2. Connect to the container and install TopHat

```
docker1 exec -it containerid /bin/bash
```

```
apt-get update
```

```
apt-get install tophat
```

```
root@ee4d845bb163:/workdir# apt-get install tophat
```

```
Reading package lists... Done
```

```
Building dependency tree
```

```
Reading state information... Done
```

```
The following packages were automatically installed and are no longer required:
```

```
  dbus-x11 gconf2 libavahi-glib1 libbonobo2-0 libbonobo2-common libcanberra0 libgnome-2-0 libgnome2-common  
libgnomevfs2-0
```

```
  libgnomevfs2-common liborbit-2-0 libtdb1 libvorbisfile3 sound-theme-freedesktop
```

```
[...]
```

```
Processing triggers for libc-bin (2.23-0ubuntu9) ...
```

```
Setting up libboost-thread1.58.0:amd64 (1.58.0+dfsg-5ubuntu3.1) ...
```

```
Setting up libtbb2:amd64 (4.4~20151115-0ubuntu3) ...
```

```
Setting up bowtie2 (2.2.6-2) ...
```

```
Setting up tophat (2.1.0+dfsg-1build1) ...
```

```
Processing triggers for libc-bin (2.23-0ubuntu9) ...
```

```
root@ee4d845bb163:/workdir#
```

BioHPC Docker Example: install TopHat

3. Verify it runs, save output in /workdir, exit container

```
root@ee4d845bb163:/workdir# tophat -h
tophat:
TopHat maps short sequences from spliced transcripts to whole genomes.

Usage:
  tophat [options] <bowtie_index> <reads1[,reads2,...]> [reads1[,reads2,...]] \
      [quals1[,quals2,...]] [quals1[,quals2,...]]

Options:
[...]
  --rg-date           <string>      (ISO 8601 date of the sequencing run)
  --rg-platform      <string>      (Sequencing platform descriptor)

  for detailed help see http://tophat.cbcb.umd.edu/manual.html
root@ee4d845bb163:/workdir# tophat -h >& /workdir/tophat.help
root@ee4d845bb163:/workdir# ls -al /workdir
total 8
drwxr-xr-x  2 4965 root   24 May 22 22:14 .
drwxr-xr-x 22 root root  257 May 22 21:52 ..
-rw-r--r--  1 root root 6620 May 22 22:15 tophat.help
root@ee4d845bb163:/workdir# exit
exit
[jarekp@cbsum1c2b014 ~]$
```

BioHPC Docker Example: install TopHat

4. How about the resulting files? Where are they and how to get them?

```
[jarekp@cbsum1c2b014 ~]$ ls -al /workdir/jarekp
total 8
drwxr-xr-x  2 jarekp root   24 May 22 18:14 .
drwxrwxrwx. 4 root   root   30 May 22 13:26 ..
-rw-r--r--  1 root   root  6620 May 22 18:15 tophat.help
```

```
[jarekp@cbsum1c2b014 ~]$ docker1 claim
```

```
[jarekp@cbsum1c2b014 ~]$ ls -al /workdir/jarekp
total 8
drwxr-xr-x  2 jarekp root   24 May 22 18:14 .
drwxrwxrwx. 4 root   root   30 May 22 13:26 ..
-rw-r--r--  1 jarekp root  6620 May 22 18:15 tophat.help
[jarekp@cbsum1c2b014 ~]$
```

Files generated in Docker may belong to root or other system users. You need to use **docker claim** to get permissions to deal with them.

BioHPC Docker Example: install TopHat

5. Save the container as image for future use

```
[jarekp@cbsum1c2b014 ~]$ docker1 ps -a
```

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS
ee4d845bb163	docker.io/biohpc/ubuntuudev	"/bin/bash"	28 minutes ago	Up 28 minutes

```
jarekp__biohpc_1
```

```
[jarekp@cbsum1c2b014 ~]$ docker1 export -o /home/jarekp/mydockerimage.tar ee4d845bb163
```

```
[jarekp@cbsum1c2b014 ~]$ ls -alh /home/jarekp/mydockerimage.tar
```

```
-rw----- 1 jarekp jarekp 1.2G May 22 18:22 /home/jarekp/mydockerimage.tar
```

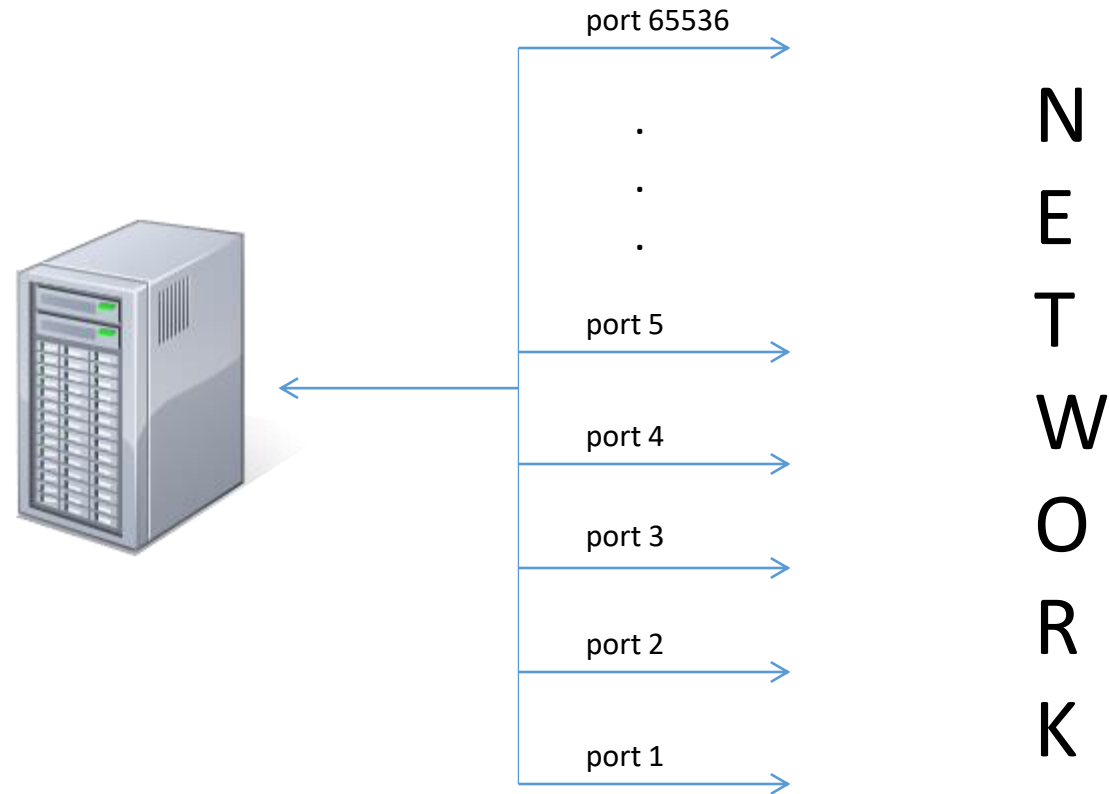
```
[jarekp@cbsum1c2b014 ~]$
```

BioHPC Docker - network ports

Each service on the network is referenced by two values

1. Server address (i.e. IP number usually linked to a name)
2. Service port (a number referencing network socket to connect to).

BioHPC Docker - network ports



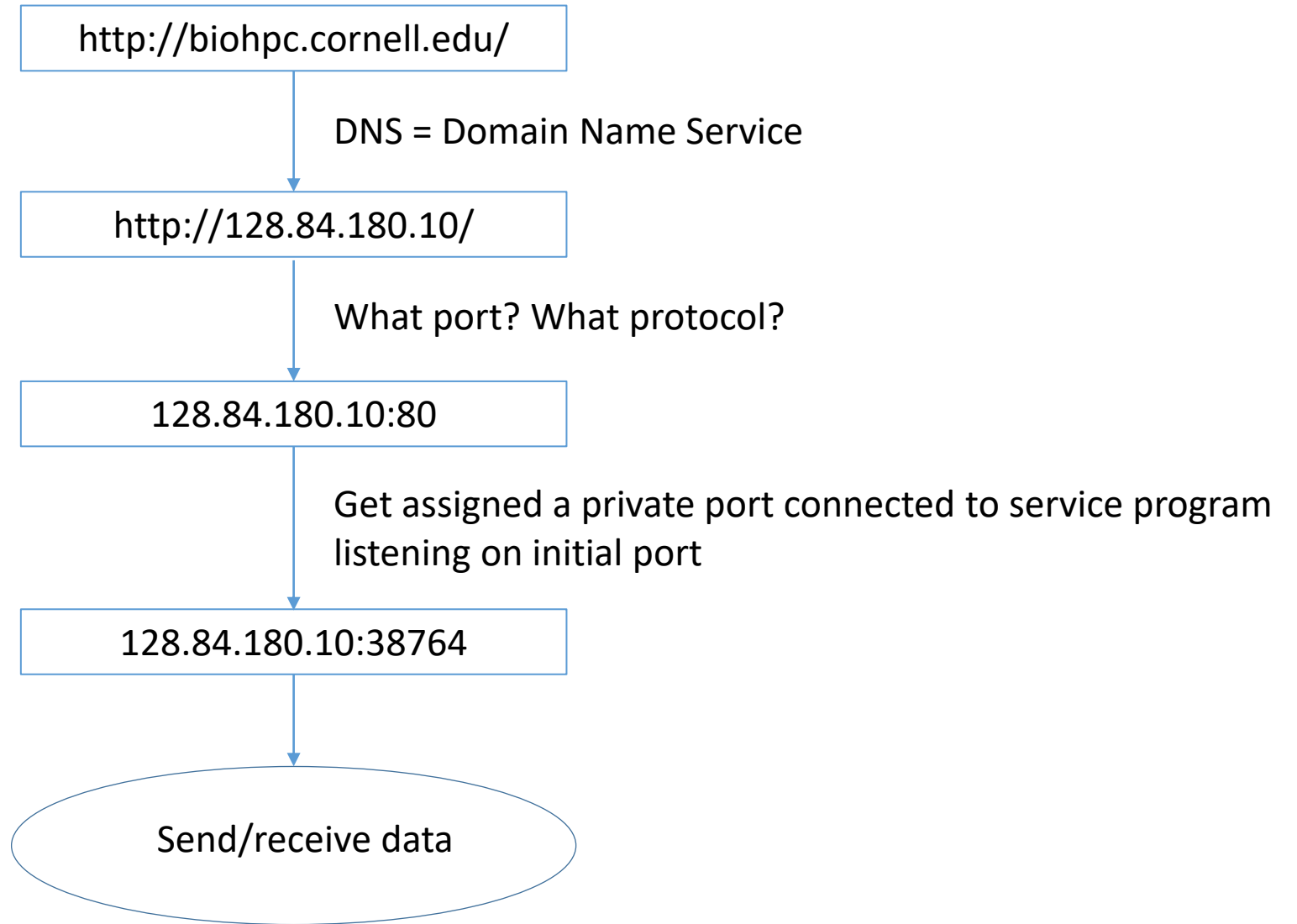
Computer full address: ip_number:port

i.e. 128.8.3.22:22

BioHPC Docker - network ports

Service (protocol)	Port
FTP	20 and 21
TELNET	23
SSH	22
SMTP (mail service)	25
DNS (domain name system)	53
HTTP (www)	80
HTTPS (www secure)	443

BioHPC Docker - network ports



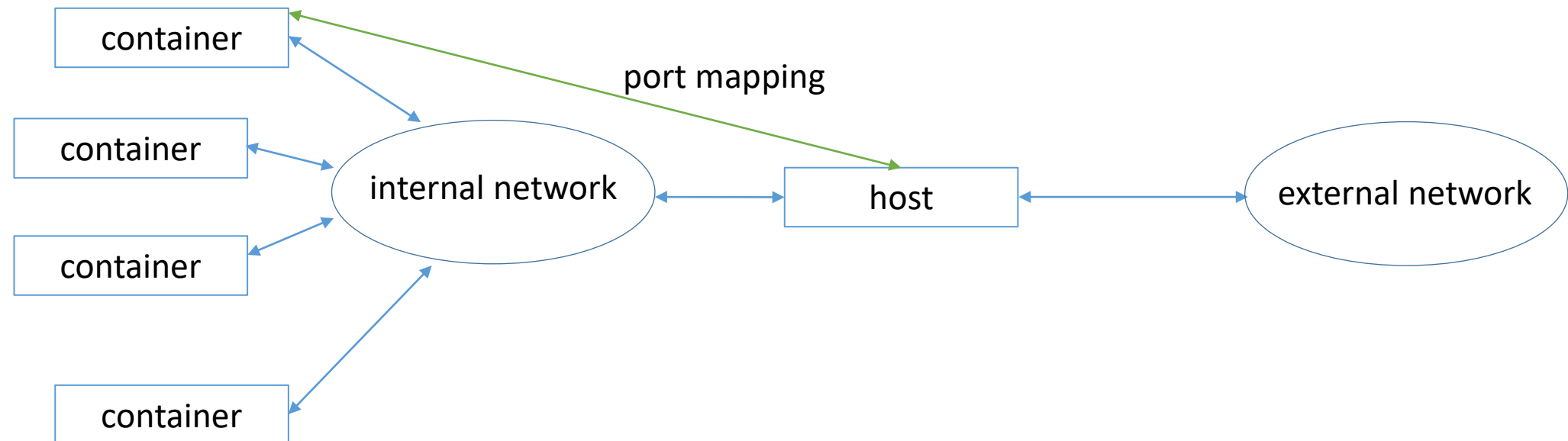
BioHPC Docker - network ports

Containers can offer network services, and you can connect to them if you know what is container IP and service port.

You can also map container port to host port. Then you can connect to your host and reference assigned port number to access container service.

BioHPC Docker - network ports

Docker maintains an internal network inside host usually 172.17.0.*. This network is NOT accessible from the outside of the server, but it is used for communication between containers and the host server.



BioHPC Docker – setting up a web server

1. Google about “web server” on CentOS Linux – it is Apache
2. Find out how to install Apache on CentOS 7 – plenty examples
3. Do it – pull CentOS 7 image and install Apache
4. How to run Apache in container – search that too, it may be different than in non-Docker CentOS
5. Prepare test HTML file, startup script and run it!
6. Map ports so you can enjoy your new web server from campus network

BioHPC Docker – setting up a web server

pull CentOS 7 and install Apache inside

```
[jarekp@cbsum1c1b009 ~]$ docker1 run -it -d centos:7 /bin/bash
Unable to find image 'centos:7' locally
Trying to pull repository dtr.cucloud.net/centos ...
Trying to pull repository docker.io/library/centos ...
sha256:4a701376d03f6b39b8c2a8f4a8e499441b0d567f9ab9d58e4991de4472fb813c: Pulling from
docker.io/library/centos
ab5ef0e58194: Pull complete
Digest: sha256:4a701376d03f6b39b8c2a8f4a8e499441b0d567f9ab9d58e4991de4472fb813c
Status: Downloaded newer image for docker.io/centos:7
8e7a04aac719de0f8e553e820dc1df7ac8537c6b7f418d7d219eded11141b665
[jarekp@cbsum1c1b009 ~]$ docker1 ps -a
CONTAINER ID        IMAGE               COMMAND             CREATED             STATUS
PORTS              NAMES
8e7a04aac719        centos:7           "/bin/bash"        2 minutes ago      Up 2 minutes
jarekp__biohpc_1
[jarekp@cbsum1c1b009 ~]$ docker1 exec -it 8e7a04aac719 /bin/bash
[root@8e7a04aac719 /]# yum -y install httpd
Loaded plugins: fastestmirror, ovl
Determining fastest mirrors
 * base: mirror.siena.edu
 * extras: mirror.mi.incx.net
 * updates: mirror.metrocast.net
base
```

BioHPC Docker – setting up a web server

How to start a service inside container?

Apache service name is 'httpd' and normally it is started on CentOS with command 'systemctl start httpd'

Won't work in a container, it requires 'systemd' service, not present in Docker.

Instead search online “how to start httpd in CentOS Docker”

BioHPC Docker – setting up a web server

```
/usr/bin/env bash -c 'exec /usr/sbin/apachectl -DFOREGROUND' &
```

In the container create a file `/start.sh` that will be starting our web server

```
-----
```

```
#!/bin/bash
```

```
rm -rf /run/httpd/* /tmp/httpd*
```

```
/usr/bin/env bash -c 'exec /usr/sbin/apachectl -DFOREGROUND' &
```

```
/bin/bash
```

```
-----
```

BioHPC Docker – setting up a web server

Also create a test HTML file `/var/www/html/test.htm` so we can see something in browser

```
<h1>Hello from Docker</h1
```

```
[root@8e7a04aac719 /]# vi /var/www/html/test.htm
```

```
[root@8e7a04aac719 /]# vi /start.sh
```

```
[root@8e7a04aac719 /]# chmod a+x /start.sh
```

```
[root@8e7a04aac719 /]# exit
```

BioHPC Docker – setting up a web server

Our web server is almost ready. We need to save it as an image so we can start it later and execute `/start.sh` to run our web server program.

```
[jarekp@cbsum1c1b009 ~]$ docker1 ps -a
CONTAINER ID        IMAGE               COMMAND             CREATED             STATUS              PORTS              NAMES
8e7a04aac719       centos:7           "/bin/bash"        30 minutes ago     Up 30 minutes      jarekp__biohpc_1
[jarekp@cbsum1c1b009 ~]$ docker1 commit 8e7a04aac719 webserv
sha256:0b7cd07abff03b034c3ba8146e3ae9f620e1ca1a41a5700e8635965ecd9569d7
[jarekp@cbsum1c1b009 ~]$ docker1 images
REPOSITORY          TAG                IMAGE ID           CREATED            SIZE
biohpc_jarekp/webserver  latest           0b7cd07abff0      11 seconds ago   348 MB
[jarekp@cbsum1c1b009 ~]$ docker1 save -o webserv
server.tar biohpc_jarekp/webserver
```

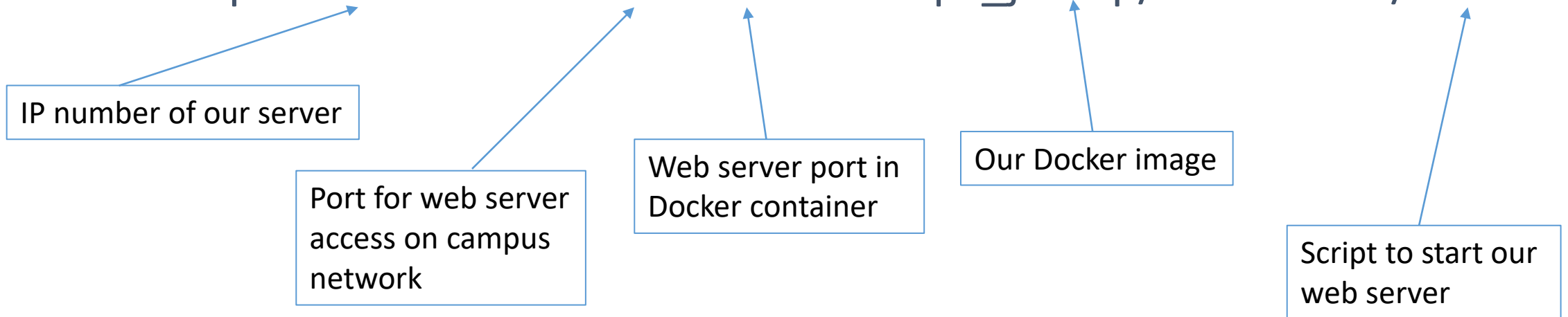
New image is stored on this server only!
MUST be saved to reuse on other servers.

BioHPC Docker - network ports

You can map container ports to external host ports, but they need to be opened in firewall to be accessible.

We keep ports 8009 – 8019 open for campus access.

```
docker1 run -d -p 128.84.181.175:8009:80 -t biohpc_jarekp/webserver /start.sh
```



BioHPC Docker – setting up a web server

```
[jarekp@cbsum1c1b009 ~]$ ping cbsum1c1b009
PING cbsum1c1b009 (128.84.181.175) 56(84) bytes of data.
64 bytes from cbsum1c1b009 (128.84.181.175): icmp_seq=1 ttl=64 time=0.050 ms
^C
--- cbsum1c1b009 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 0.050/0.050/0.050/0.000 ms
```

Easy way to find IP number of our server

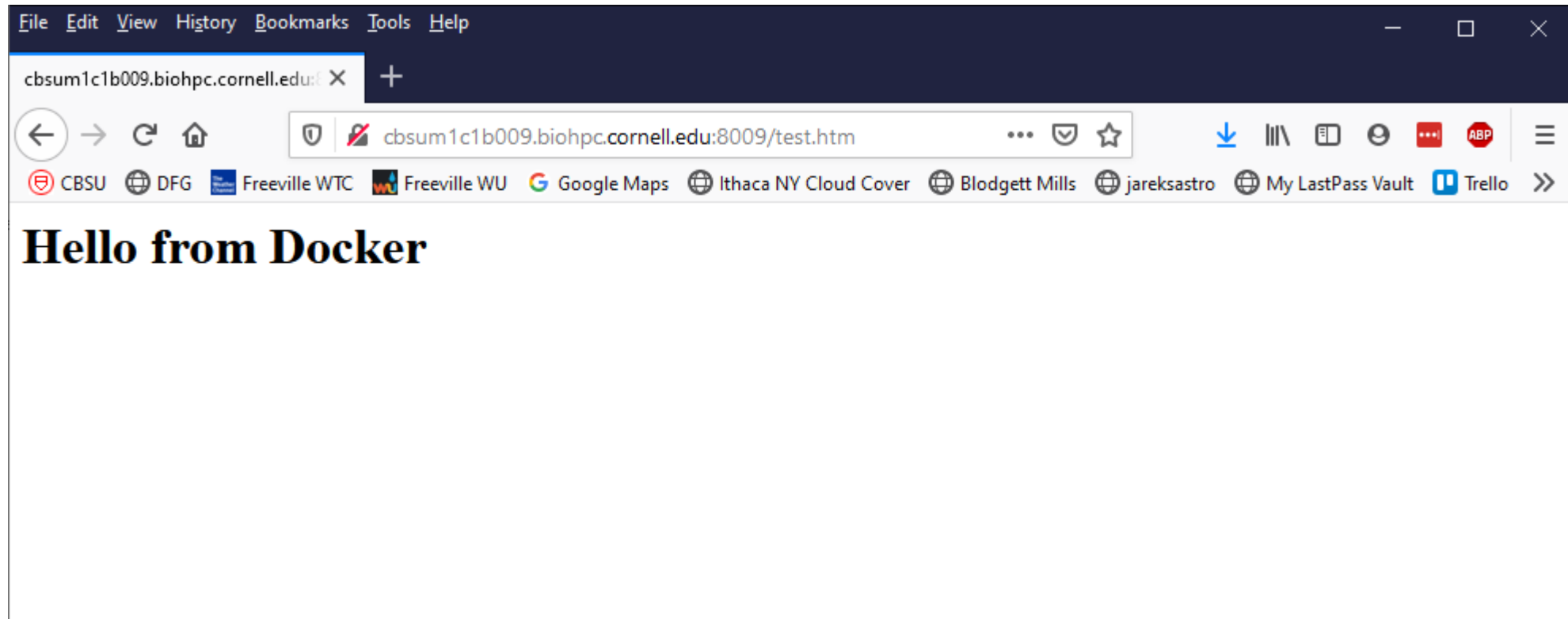
```
[jarekp@cbsum1c1b009 ~]$ docker1 run -d -p 128.84.181.175:8009:80 -t biohpc_jarekp/webserver /start.sh
909f4daec197c17b81e5ba59476634065cbfb16cd85efb317593a96b67993e48
```

```
[jarekp@cbsum1c1b009 ~]$ docker1 ps -a
```

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NAMES
909f4daec197	biohpc_jarekp/webserver	"/start.sh"	13 seconds ago	Up 11 seconds	128.84.181.175:8009->80/tcp	jarekp__biohpc_2

NOTE: several users can run containers on your server. Run the command above to check if port 8009 is free. If not use other port between 8009-8019. If you try to use port that has been allocated you will get an error message “port is already allocated”.

BioHPC Docker – setting up a web server



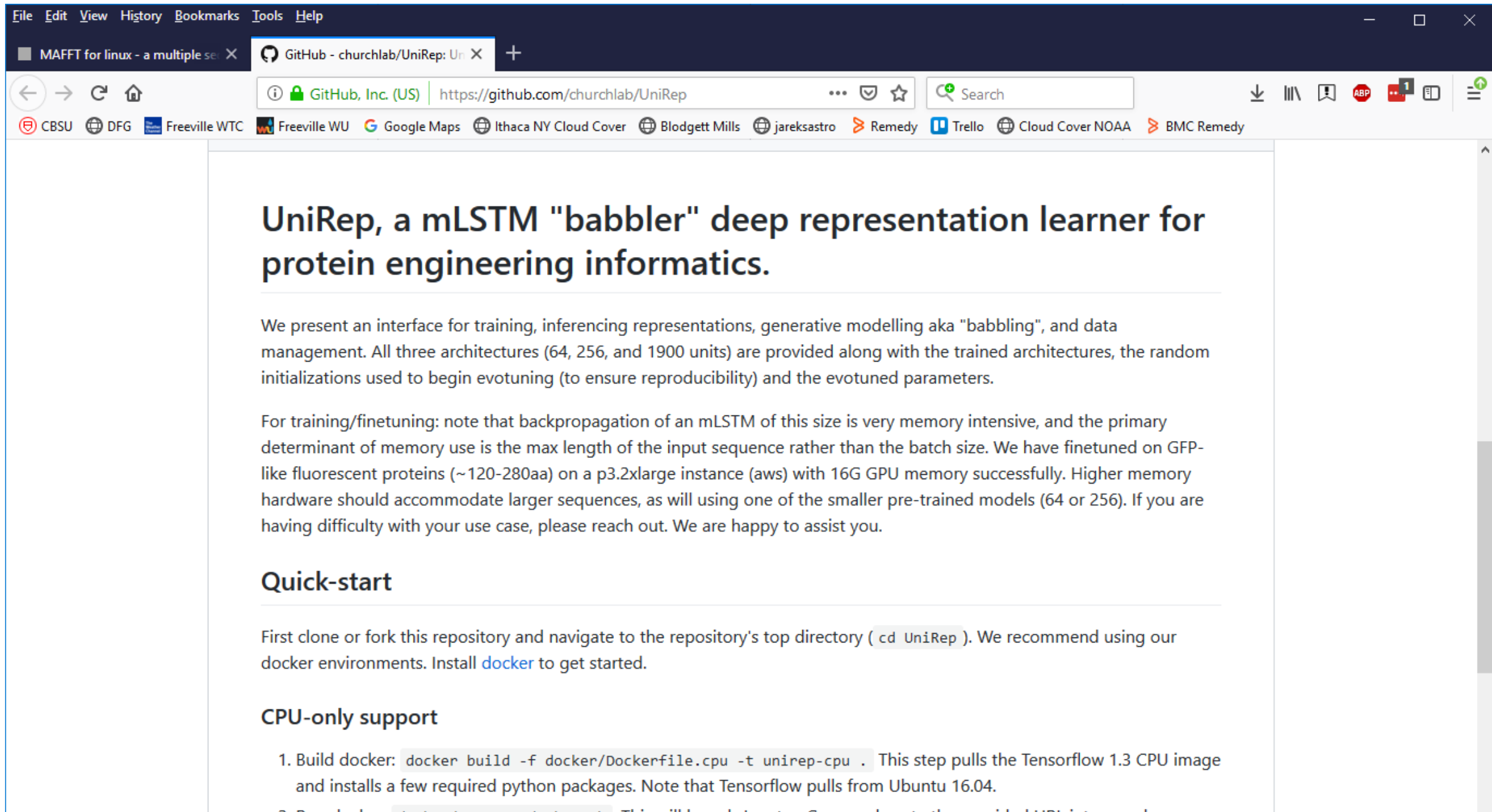
Our URL is

<http://cbsum1c1b009.biohpc.cornell.edu:8009/test.htm>

BioHPC Docker Example – install UniRep from dockerfile

1. Go to UniRep website and read instructions.
2. Download appropriate dockerfile
3. Build image from dockerfile
4. Save image for future use

BioHPC Docker Example – install UniRep from dockerfile



File Edit View History Bookmarks Tools Help

MAFFT for linux - a multiple se x GitHub - churchlab/UniRep: Un x +

GitHub, Inc. (US) | https://github.com/churchlab/UniRep

CBSU DFG Freeville WTC Freeville WU Google Maps Ithaca NY Cloud Cover Blodgett Mills jareksastro Remedy Trello Cloud Cover NOAA BMC Remedy

UniRep, a mLSTM "babbling" deep representation learner for protein engineering informatics.

We present an interface for training, inferencing representations, generative modelling aka "babbling", and data management. All three architectures (64, 256, and 1900 units) are provided along with the trained architectures, the random initializations used to begin evotuning (to ensure reproducibility) and the evotuned parameters.

For training/finetuning: note that backpropagation of an mLSTM of this size is very memory intensive, and the primary determinant of memory use is the max length of the input sequence rather than the batch size. We have finetuned on GFP-like fluorescent proteins (~120-280aa) on a p3.2xlarge instance (aws) with 16G GPU memory successfully. Higher memory hardware should accommodate larger sequences, as will using one of the smaller pre-trained models (64 or 256). If you are having difficulty with your use case, please reach out. We are happy to assist you.

Quick-start

First clone or fork this repository and navigate to the repository's top directory (`cd UniRep`). We recommend using our docker environments. Install [docker](#) to get started.

CPU-only support

1. Build docker: `docker build -f docker/Dockerfile.cpu -t unirep-cpu` . This step pulls the Tensorflow 1.3 CPU image and installs a few required python packages. Note that Tensorflow pulls from Ubuntu 16.04.

BioHPC Docker Example – install UniRep from dockerfile

File Edit View History Bookmarks Tools Help

MAFFT for linux - a multiple se x GitHub - churchlab/UniRep: Un x +

GitHub, Inc. (US) | <https://github.com/churchlab/UniRep> Search

CBSU DFG Freeville WTC Freeville WU Google Maps Ithaca NY Cloud Cover Blodgett Mills jareksastro Remedy Trello Cloud Cover NOAA BMC Remedy

like fluorescent proteins (~120-200aa) on a ps.zxlarge instance (aws) with 10GB GPU memory successfully. Higher memory hardware should accommodate larger sequences, as will using one of the smaller pre-trained models (64 or 256). If you are having difficulty with your use case, please reach out. We are happy to assist you.

Quick-start

First clone or fork this repository and navigate to the repository's top directory (`cd UniRep`). We recommend using our docker environments. Install [docker](#) to get started.

CPU-only support

1. Build docker: `docker build -f docker/Dockerfile.cpu -t unirep-cpu .` This step pulls the Tensorflow 1.3 CPU image and installs a few required python packages. Note that Tensorflow pulls from Ubuntu 16.04.
2. Run docker: `docker/run_cpu_docker.sh` . This will launch Jupyter. Copy and paste the provided URL into your browser. Note that if you are running this code on a remote machine you will need to set up port forwarding between your local machine and your remote machine. See this [example](#) (note that in our case jupyter is serving port 8888, not 8889 as the example describes).
3. Open up the `unirep_tutorial.ipynb` notebook and get started. The 64-unit model should be OK to run on any machine. The full-sized model will require a machine with more than 16GB of RAM.

GPU support

0. System requirements: NVIDIA CUDA 8.0 (V8.0.61), NVIDIA cuDNN 6.0.21, NVIDIA GPU Driver 410.79 (though == 361.93 or >= 375.51 should work. Untested), nvidia-docker. We use the AWS [Deep Learning Base AMI for Ubuntu](#) (tested on version 17.0 ami-0ff00f007c727c376), which has these requirements pre-configured.
1. Build docker: `docker build -f docker/Dockerfile.gpu -t unirep-gpu .` This step pulls the Tensorflow 1.3 GPU

BioHPC Docker Example – install UniRep from dockerfile

```
cd /workdir/jarekp  
git clone https://github.com/churchlab/UniRep.git  
cd UniRep
```

Command suggested in instructions:

```
docker build -f docker/Dockerfile.cpu -t unirep-cpu .
```

However, docker1 requires FULL PATH to dockerfile and dockerfile directory! Our command:

```
docker1 build -f /workdir/jarekp/UniRep/docker/Dockerfile.cpu -t unirep-cpu /workdir/jarekp/UniRep
```

Command suggested to run is docker/run_cpu_docker.sh . What is inside?

```
docker run -it -p 8888:8888 -p 6006:6006 -v `pwd`:/notebooks unirep-cpu:latest
```

=> Won't work! Need to change *docker* to *docker1*. Also need to change image name

BioHPC Docker Example – install UniRep from dockerfile

docker1 images

REPOSITORY	TAG	IMAGE ID	CREATED	SIZE
biohpc_jarekp/unirep-cpu	latest	08bfd512403	5 minutes ago	1.37 GB

Our command will be:

It will only work if I am in UniRep directory



```
docker1 run -it -p 8888:8888 -p 6006:6006 -v `pwd` :/notebooks biohpc_jarekp/unirep-cpu
```

After changing docker to docker1 and image name it is a good idea to save both the image and all the files. I use vi to edit, you can use your favorite Linux editor

```
vi docker/run_cpu_docker.sh
```

```
cd ..
```

```
docker1 save -o /home/jarekp/UniRep_image.tar biohpc_jarekp/unirep-cpu
```

```
tar -cf /home/jarekp/UniRep_files.tar UniRep
```


Docker @ BioHPC

Often there is a Docker image on a public hub with a program you may want to run.

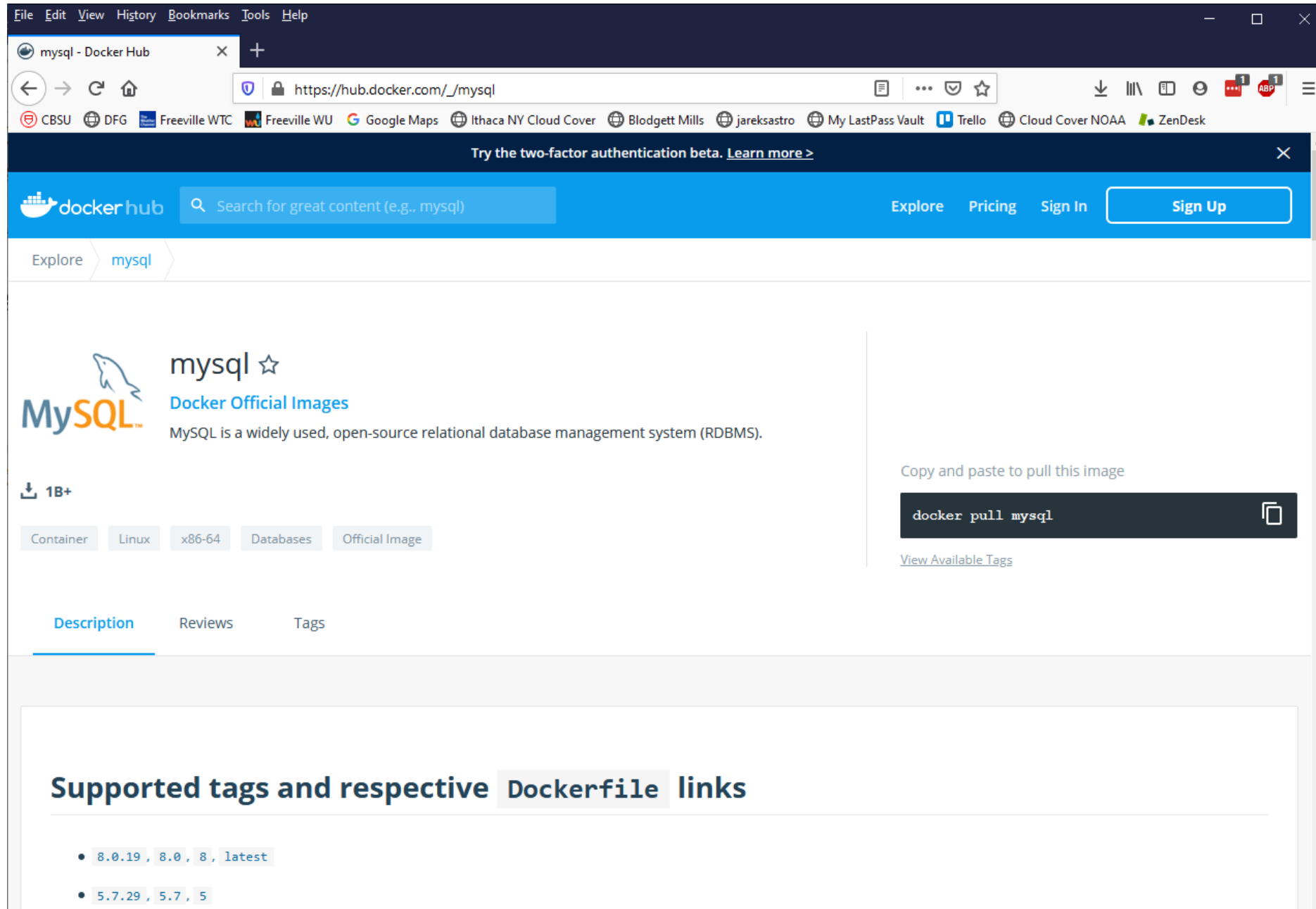
Sometimes such an image is too limited, and you will need to install it yourself anyway, but often it is useful

Usually worth a try.

BioHPC Docker Example – MySQL Database Server

1. Search online for “MySQL and Docker”.
2. Turns out there is already a Docker image for MySQL
3. Follow instructions to pull and run the image
4. Customize it as needed
5. Test access from other applications.
6. Save image.

BioHPC Docker Example – MySQL Database Server



The screenshot shows a web browser window displaying the Docker Hub page for the 'mysql' image. The browser's address bar shows 'https://hub.docker.com/_/mysql'. The page features the Docker Hub logo, a search bar, and navigation links for 'Explore', 'Pricing', 'Sign In', and 'Sign Up'. The main content area displays the 'mysql' image with the MySQL logo, a star icon, and the text 'Docker Official Images'. Below this, it states 'MySQL is a widely used, open-source relational database management system (RDBMS)'. A download icon and '1B+' are visible. There are also tags for 'Container', 'Linux', 'x86-64', 'Databases', and 'Official Image'. On the right side, there is a section for copying the image pull command, which is 'docker pull mysql'. Below this, there is a link to 'View Available Tags'. At the bottom of the page, there is a section titled 'Supported tags and respective Dockerfile links' with a list of tags: '8.0.19', '8.0', '8', 'latest', '5.7.29', '5.7', and '5'.

File Edit View History Bookmarks Tools Help


mysql - Docker Hub

https://hub.docker.com/_/mysql

Try the two-factor authentication beta. [Learn more >](#)

docker hub Search for great content (e.g., mysql) Explore Pricing Sign In Sign Up

Explore > mysql

 **mysql** ☆
Docker Official Images
MySQL is a widely used, open-source relational database management system (RDBMS).

↓ 1B+

Container Linux x86-64 Databases Official Image

Description Reviews Tags

Copy and paste to pull this image

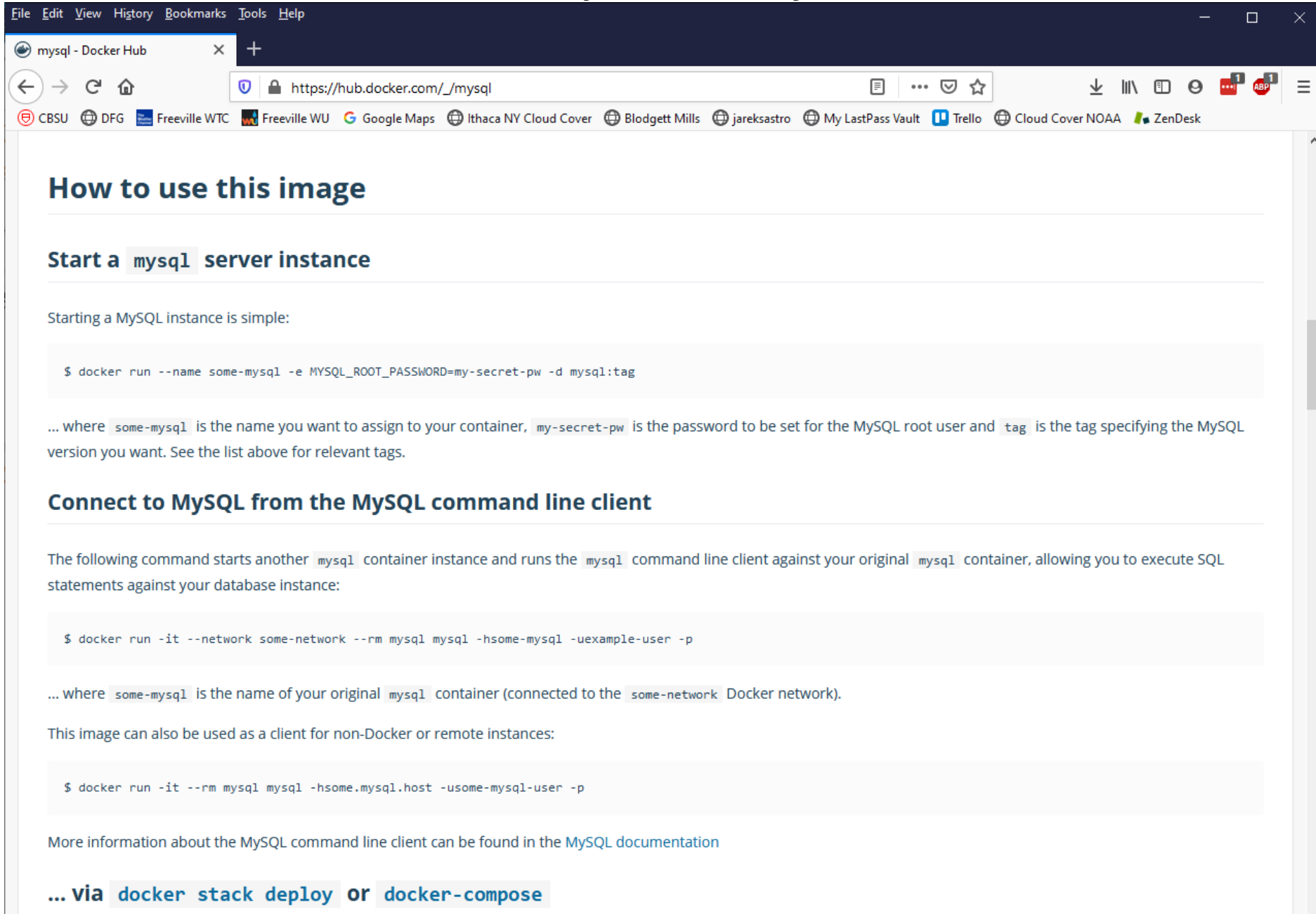
```
docker pull mysql
```

[View Available Tags](#)

Supported tags and respective Dockerfile links

- 8.0.19 , 8.0 , 8 , latest
- 5.7.29 , 5.7 , 5

BioHPC Docker Example – MySQL Database Server



File Edit View History Bookmarks Tools Help

mysql - Docker Hub

https://hub.docker.com/_/mysql

How to use this image

Start a `mysql` server instance

Starting a MySQL instance is simple:

```
$ docker run --name some-mysql -e MYSQL_ROOT_PASSWORD=my-secret-pw -d mysql:tag
```

... where `some-mysql` is the name you want to assign to your container, `my-secret-pw` is the password to be set for the MySQL root user and `tag` is the tag specifying the MySQL version you want. See the list above for relevant tags.

Connect to MySQL from the MySQL command line client

The following command starts another `mysql` container instance and runs the `mysql` command line client against your original `mysql` container, allowing you to execute SQL statements against your database instance:

```
$ docker run -it --network some-network --rm mysql mysql -hsome-mysql -uexample-user -p
```

... where `some-mysql` is the name of your original `mysql` container (connected to the `some-network` Docker network).

This image can also be used as a client for non-Docker or remote instances:


```
$ docker run -it --rm mysql mysql -hsome.mysql.host -usome-mysql-user -p
```

More information about the MySQL command line client can be found in the [MySQL documentation](#)

... via `docker stack deploy` or `docker-compose`

BioHPC Docker Example – MySQL Database Server

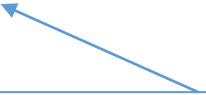
```
[jarekp@cbsum1c1b009 ~]$ docker1 run --name mysqlserver -e MYSQL_ROOT_PASSWORD=edeyewrg3 -d mysql
Unable to find image 'mysql:latest' locally
Trying to pull repository dtr.cucloud.net/mysql ...
Trying to pull repository docker.io/library/mysql ...
sha256:9643e9fbd6330d10686f8922292dcb20995e7b792c17d4e94ddf95255f1d5449: Pulling from
docker.io/library/mysql
54fec2fa59d0: Pull complete
bcc6c6145912: Pull complete
951c3d959c9d: Pull complete
05de4d0e206e: Pull complete
319f0394ef42: Pull complete
d9185034607b: Pull complete
013a9c64dadc: Pull complete
96d4c3d31f9f: Pull complete
785bc90808da: Pull complete
1339cf094729: Pull complete
beb8f531cc37: Pull complete
2b40c9f6a918: Pull complete
Digest: sha256:9643e9fbd6330d10686f8922292dcb20995e7b792c17d4e94ddf95255f1d5449
Status: Downloaded newer image for docker.io/mysql:latest
79b38279788c923b4aab7a3ae2830f28fa392f56e634714885482991a7c23177
[jarekp@cbsum1c1b009 ~]$
```



Password to access
the database.

BioHPC Docker Example – MySQL Database Server

```
[jarekp@cbsum1c1b009 ~]$ docker1 ps -a
CONTAINER ID   IMAGE     COMMAND                  CREATED        STATUS        PORTS                    NAMES
79b38279788c  mysql    "docker-entryp.."       35 seconds ago Up 34 seconds  3306/tcp, 33060/tcp     jarekp__biohpc_mysqlserver
[jarekp@cbsum1c1b009 ~]$
```



Network ports exposed from container,
but what network?

Before we can connect to our MySQL database remotely, we need to figure out what is our container IP address.

Docker maintains an internal network inside host usually 172.17.0.*. This network is NOT accessible from the outside of the server, but it is used for communication between containers and the host server.

Use command “`docker1 inspect conatinerid`” to find out more.

We have two ways to communicate with the database: get inside container or use network.

BioHPC Docker Example – MySQL Database Server

```
[jarekp@cbsum1c1b009 ~]$ docker1 exec -it jarekp__biohpc_mysqlserver /bin/bash
```

```
root@79b38279788c:/# mysql -u root -p
```

```
Enter password:
```

```
Welcome to the MySQL monitor.  Commands end with ; or \g.
```

```
Your MySQL connection id is 8
```

```
Server version: 8.0.19 MySQL Community Server - GPL
```

```
Copyright (c) 2000, 2020, Oracle and/or its affiliates. All rights reserved.
```

```
Oracle is a registered trademark of Oracle Corporation and/or its  
affiliates. Other names may be trademarks of their respective  
owners.
```

```
Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.
```

```
mysql> show databases;
```

```
+-----+  
| Database          |  
+-----+  
| information_schema |  
| mysql              |  
| performance_schema |  
| sys                |  
+-----+
```

```
4 rows in set (0.00 sec)
```

BioHPC Docker Example – MySQL Database Server

```
[jarekp@cbsum1c1b009 ~]$ docker1 inspect jarekp__biohpc_mysqlserver  
[...]
```

```
"NetworkSettings": {  
  "Bridge": "",  
  "SandboxID": "b8e44e387c4e0cb7a63c4abe86386f72f0f1b1524854bdf570ac9921b541f63b",  
  "HairpinMode": false,  
  "LinkLocalIPv6Address": "",  
  "LinkLocalIPv6PrefixLen": 0,  
  "Ports": {  
    "3306/tcp": null,  
    "33060/tcp": null  
  },  
  "SandboxKey": "/var/run/docker/netns/b8e44e387c4e",  
  "SecondaryIPAddresses": null,  
  "SecondaryIPv6Addresses": null,  
  "EndpointID": "7e64e46e4bb45e790114fff95e4c9ede3c2efc0d72f29e09363d76cdca21ee23",  
  "Gateway": "172.17.0.1",  
  "GlobalIPv6Address": "",  
  "GlobalIPv6PrefixLen": 0,  
  "IPAddress": "172.17.0.2",  
  "IPPrefixLen": 16,
```



Here it is!

BioHPC Docker Example – MySQL Database Server

```
[jarekp@cbsum1c1b009 ~]$ mysql -u root -p -h 172.17.0.2
Enter password:
ERROR 2059 (HY000): Authentication plugin 'caching_sha2_password' cannot be loaded:
/usr/lib64/mysql/plugin/caching_sha2_password.so: cannot open shared object file: No such file or directory
[jarekp@cbsum1c1b009 ~]$
```

Something is wrong!

What happened is that the local “mysql” program uses different type of passwords than the container (legacy vs sha2)

If you search Internet you will find it is a common problem, and it has a common solution: modify password type :-)

BioHPC Docker Example – MySQL Database Server

```
[jarekp@cbsum1c1b009 ~]$ docker1 exec -it jarekp__biohpc_mysqlserver /bin/bash
root@79b38279788c:/# mysql -u root -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 10
Server version: 8.0.19 MySQL Community Server - GPL

Copyright (c) 2000, 2020, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> ALTER USER 'root'@'%' IDENTIFIED WITH mysql_native_password BY 'edeyewrg3';
Query OK, 0 rows affected (0.04 sec)

mysql> quit
Bye
root@79b38279788c:/# exit
exit
[jarekp@cbsum1c1b009 ~]$
```

BioHPC Docker Example – MySQL Database Server

```
[jarekp@cbsum1c1b009 ~]$ mysql -u root -p -h 172.17.0.2
```

```
Enter password:
```

```
Welcome to the MySQL monitor.  Commands end with ; or \g.
```

```
Your MySQL connection id is 11
```

```
Server version: 8.0.19 MySQL Community Server - GPL
```

```
Copyright (c) 2000, 2019, Oracle and/or its affiliates. All rights reserved.
```

```
Oracle is a registered trademark of Oracle Corporation and/or its  
affiliates. Other names may be trademarks of their respective  
owners.
```

```
Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.
```

```
mysql> show databases;
```

```
+-----+  
| Database          |  
+-----+  
| information_schema |  
| mysql              |  
| performance_schema |  
| sys                |  
+-----+
```

```
4 rows in set (0.00 sec)
```

BioHPC Docker Example – MySQL Database Server

You can run MySQL container with native port 3306 mapped to an external port from range 8009-8019. This would allow to connect to the database from anywhere on campus network. Similar as with our web server example.

You can also install MySQL server yourself.
Instructions are provided in “optional exercise” file.

Exercises

[install TopHat](#)

[setting up a web server](#)

[install UniRep from dockerfile](#)

[install MySQL Database](#)