Introduction to the world of containers

Emilio Lario

Introduction to containers

- Packages code and all its dependencies.
- Runs anywhere with a compatible kernel.

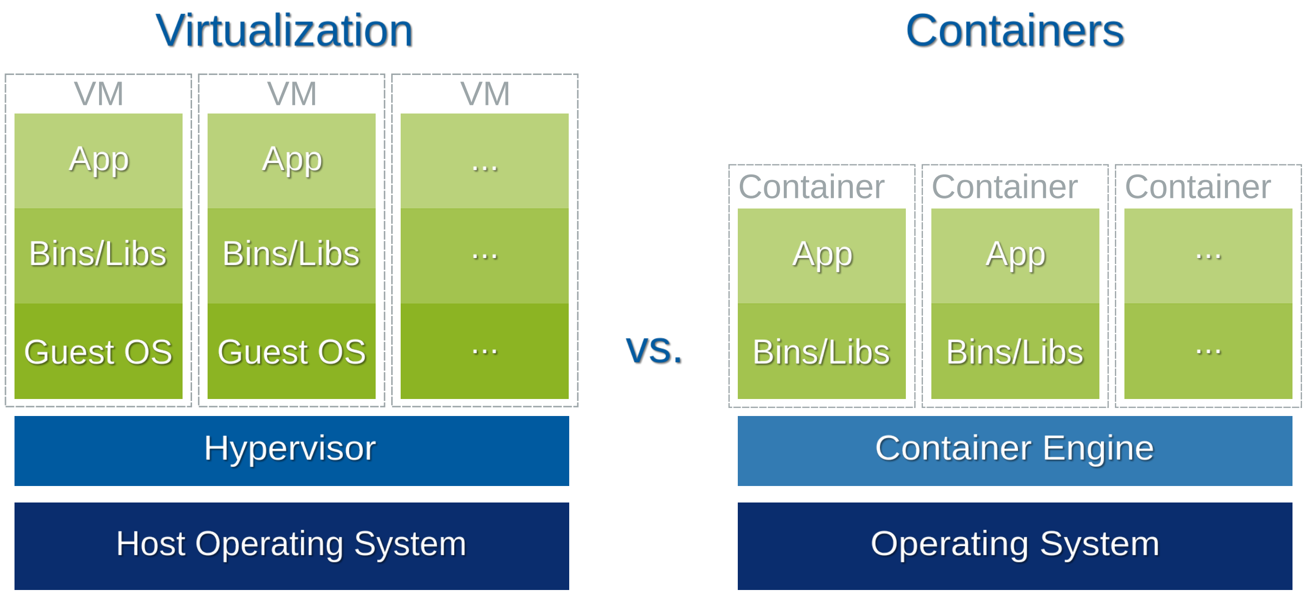
Exercice 1

1. Go to the VPN and log in to the Linux course machine: vpn
2. Find a container named "lolcow" from "godlovedc" on Docker Hub: hub.docker.com
3. Go back to the Linux terminal and type:

```
singularity run
docker://container/name
```

▶ Solution

Virtualization and containerization



Virtualization

VM | VM | VM

App | App | …

Bins/Libs | Bins/Libs | …

Guest OS | Guest OS | …

Hypervisor

Host Operating System

vs.

Containers

Container | Container | Container

App | App | …

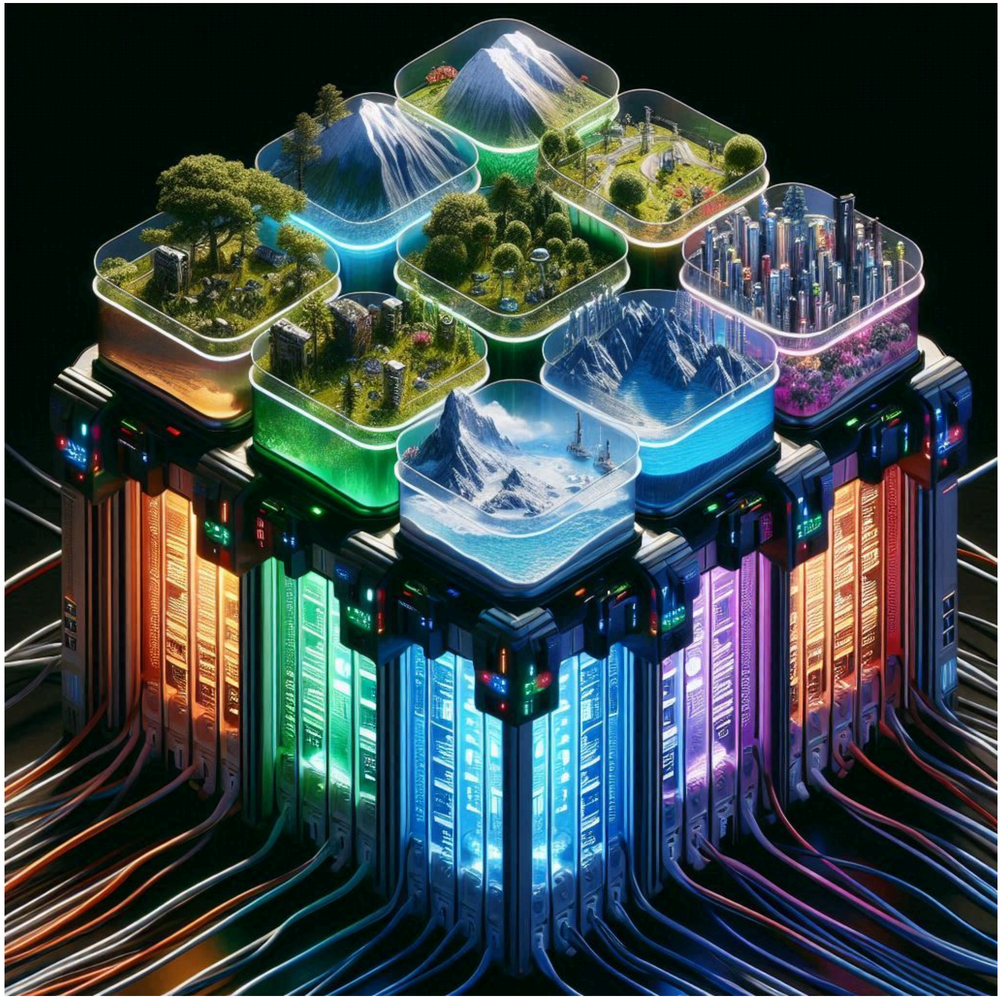
Bins/Libs | Bins/Libs | …

Container Engine

Operating System

	Virtual Machines
Guest OS	Each VM runs on virtual hardware and the kernel is loaded into its own memory region.
Communication	Through Ethernet Devices.
Performance	Small overhead as the Machine instructions need to be translated from Guest to Host OS.
Startup time	Takes up to a few minutes to boot up.
Isolation	Sharing libraries, files, etc. between guests and between guests and host not natively possible.
Storage	VMs usually require more storage as the whole OS kernel and associated programs have to be install

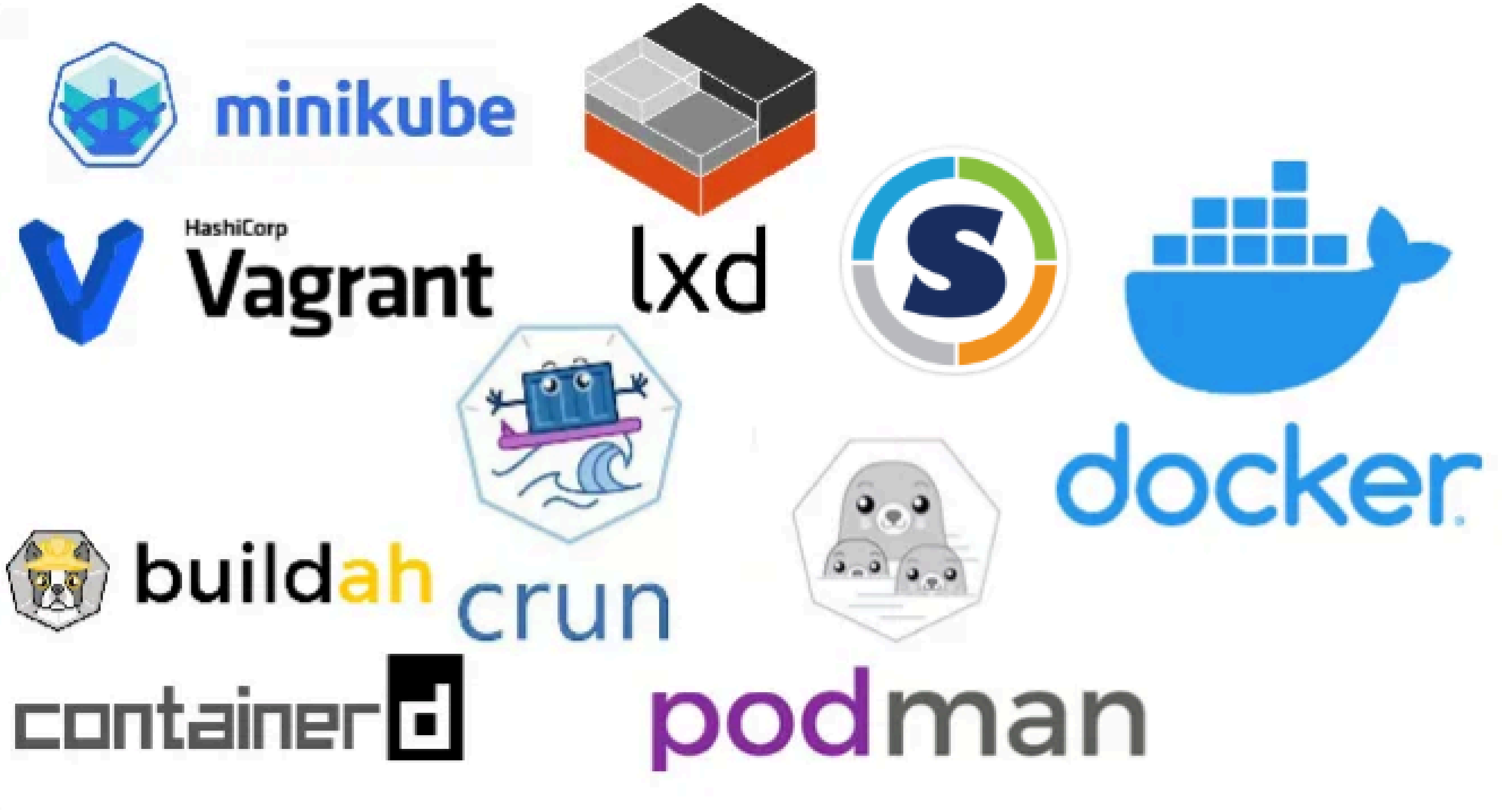
How containers work?



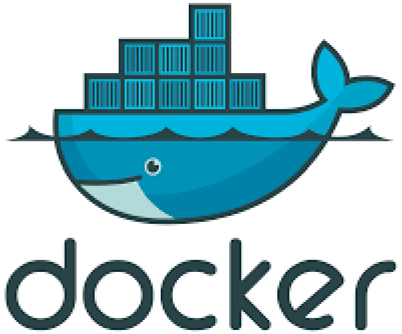
- Namespaces for access control
- cgroups for resources management

For a detailed explanation, see Iván Moreno’s article on Medium

Not only Docker



Docker vs Singularity



Sudo in docker

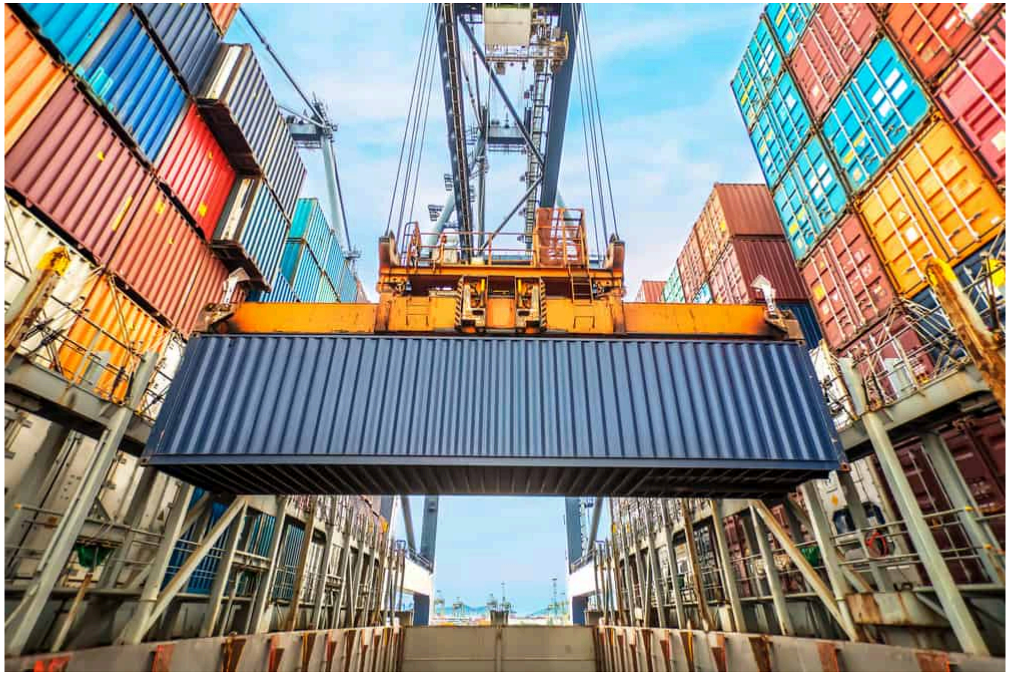
Singularity runs with the same user that starts the container while the Docker daemon runs as root.

Host files access

Default Behavior: Docker isolates the container from the host machine by default. Singularity mounts the /home directory of the user running the container.

> ⚠ **Side effects of mounting the home directory**
>
> All user customization files will be in the container when it runs. This can break the isolation/reproducibility concept, such as having user-defined R libraries.
>
> It is recommended to run Singularity with –no-home or even with –containall (-C) and bind only the required directories.

Pulling container images



You can create your own container image from ‘scratch,’ but normally you will start with an image that someone else has already created. To do so, you can either run the image straight from the external repository or download it locally and run it later.

Note

1. Go back to the Linux terminal and type: `singularity pull local_image_name.sif docker://container/name` to download the image.
2. Run the local image with: `singularity run local_image_name.sif`

▶ Solution

Interactive shell

When we run the container, it loads the image and executes the predefined command, if one is defined. Instead

Exercice 3

1. To open an interactive shell in a Singularity container, type: `singularity shell local_image_name.sif`
   (To exit the container, type `exit`)
2. Check the user outside and inside the container with: `whoami`
3. Check the contents of the home directory outside and inside the container:
4. Now run the interactive shell with the `-C` option and verify that the home directory is empty.

▶ Solution

Run in a container

We can tell the container to run and execute a command with:

```
singularity exec local_image_name.sif command
```

Exercice 4

1. Check the operating system outside the container with:

```
head /etc/os-release
```

2. Check the operating system inside the container with:

```
singularity exec local_image_name.sif command
```

▶ Solution

Files access

Persistent changes

All changes made inside a container are lost when the container is stopped. If we need the changes to be persistent, we must use a folder from the host machine.

If we need to access files outside the container, we can “BIND” folders from the host machine to the container:

```
singularity shell  -B /host/path:/path/in/container local_image_name.sif
```

Exercice 5

1. Create a folder named “dummy” on the host machine.
2. Open a terminal inside the container with the `-C` option (`--containall`).
3. Check that there is no “dummy” folder.
4. Exit the container and enter again, but this time use the bind (-B) option to mount the directory you have created to “/dummy” inside the container.
5. Create a text file inside the “/dummy” directory inside the container with `touch test.txt`
6. Exit the container and check that the file you created inside the container is in the “dummy” directory of the host machine:

▶ Solution

Build a container

- scratch
- Alpine
- Ubuntu
- DockerHUb https://hub.docker.com/ –> Rocker, Bioconductor, conda, …

Singularity & Apptainer

S
APPTAINER

“–fakeroot” or remote build

Container recipe

```
Bootstrap: localimage
From: lolcow.sif

%post
  apt-get -y update
  apt-get -y install sl=3.03-17build1
  apt-get -y install curl

%runscript
  sl -F
```

Exercice 6

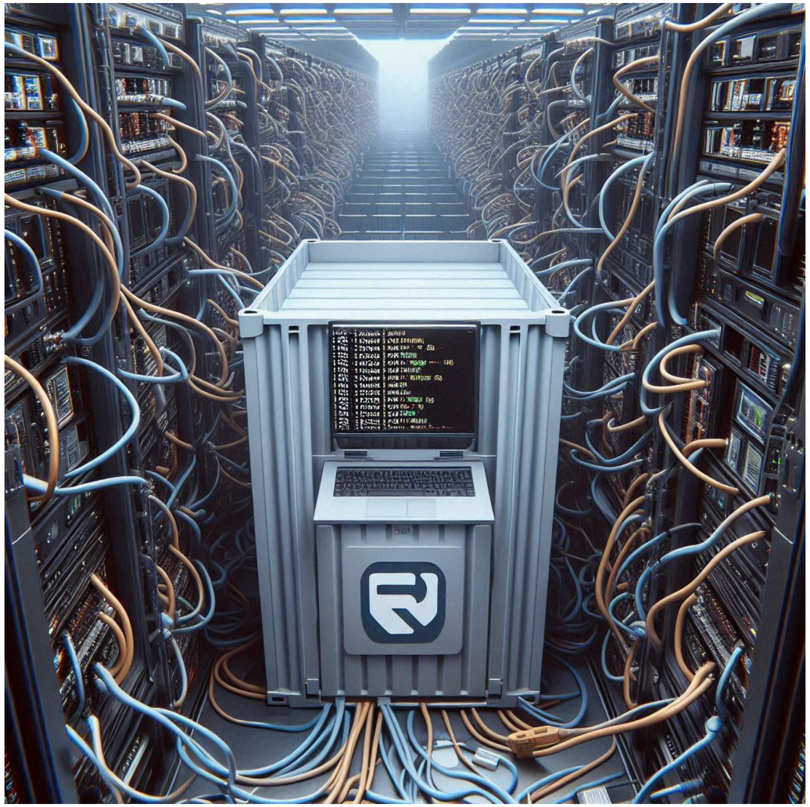
1. Create a file named steam.def with the above code.
2. Build the modified image with the command: `apptainer build image_name.sif recipe.def`
3. Run the newly created container.

▶ Solution

Importance of version reference

Different versions of a dependency may not be 100% compatible.

Containers in the IJC’s HPC



- Account for the resources needed by all the dependencies of your script.

- Do not run a container on the master node!

- We are not sudo in the cluster.
- Every dependency must be installed by the system administrator.
- Run containers with the Slurm scheduler.

Exercice 7

1. Create a bash script file with the following content:

```
#!/bin/bash
curl -s "wttr.in/$1?m1"
```

2. Create a Slurm script to execute the script from step 1 inside the container you made in the previous exercise. Copy the following sbatch header and **add the command to run the script inside the container**. The container will run the command that you type after the image name; therefore, you need to use bash script.sh to actually run the script.

```
#!/bin/bash
#SBATCH --job-name=run_lolcow
```

3. Use `sbatch` to run the script and check the output. You can use `squeue` to see the progress. You will find the log file in the current folder when the job finishes.

▶ Solution