



Norwegian
Meteorological
Institute



SINGULARITYCE

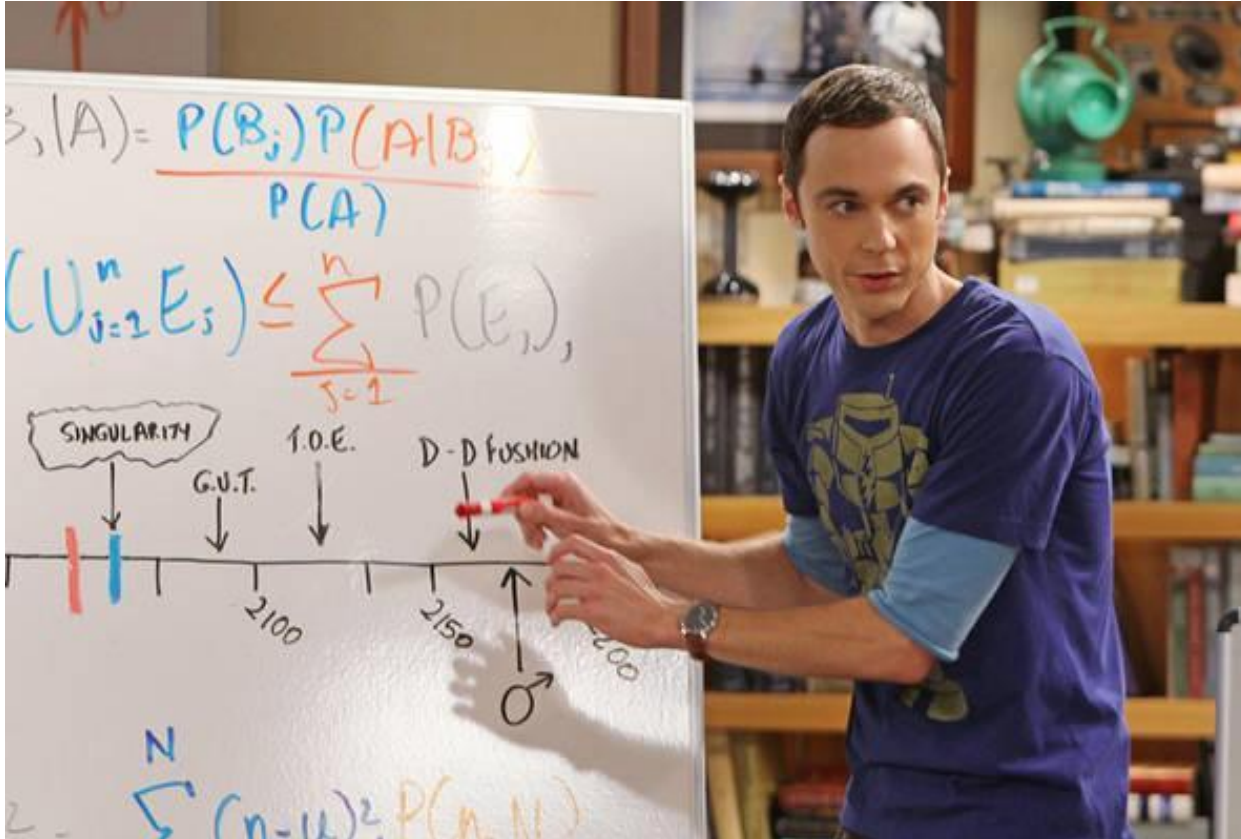
Singularity on PPI

Dr. Georgios Magklaras - Senior Engineer - IT Infrastructure - HPC

20.09.2022

Classification: Open, Internal

Agenda



© CBS Corporation

Agenda

- Containers
 - Why and how
 - History
 - A review of the container ecosystem

- Singularity on PPI
 - Reasons to select
 - How to use
 - [Singularity tutorial/practical](#)
 - Image building pointers

Containers - Scenarios on why?

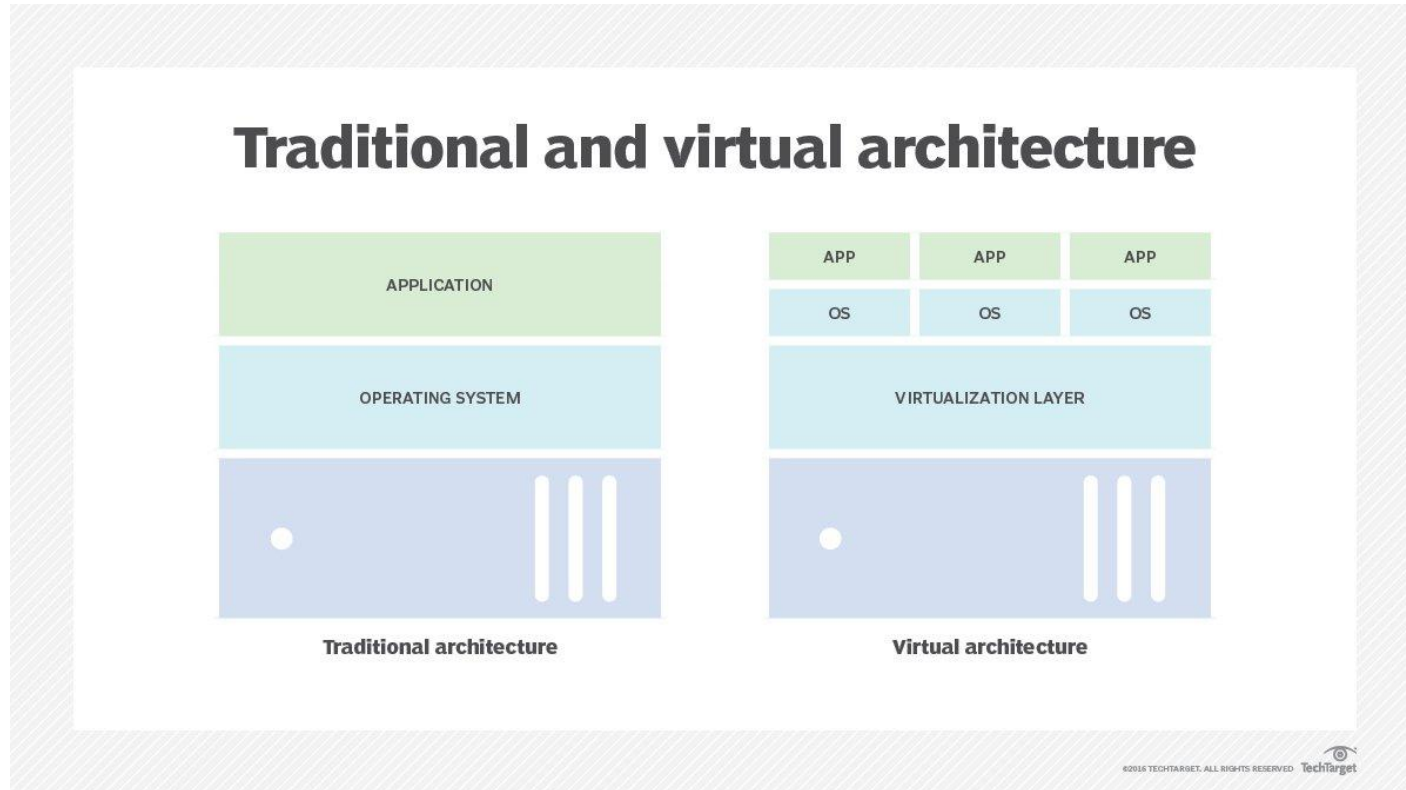
“I am often confused with which modules or conda environments I have to use. Can I not have a simpler mechanism?” (ISC 2022)

“I am developing on Ubuntu, but my supercomputer vendor runs on some sort of RedHat variant.” (MET)

“When I distribute my complex application with a gazillion dependencies, how do I ensure it will run properly in systems I do not control?” (ISC 2022)

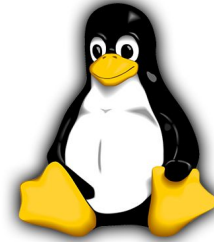
“I want to execute my workload and be done, not consume compute resources while I am not running.” (USENIX LISA 2020)

Containers - how?(1)

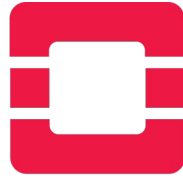


Containers - how?(2)

vmware®



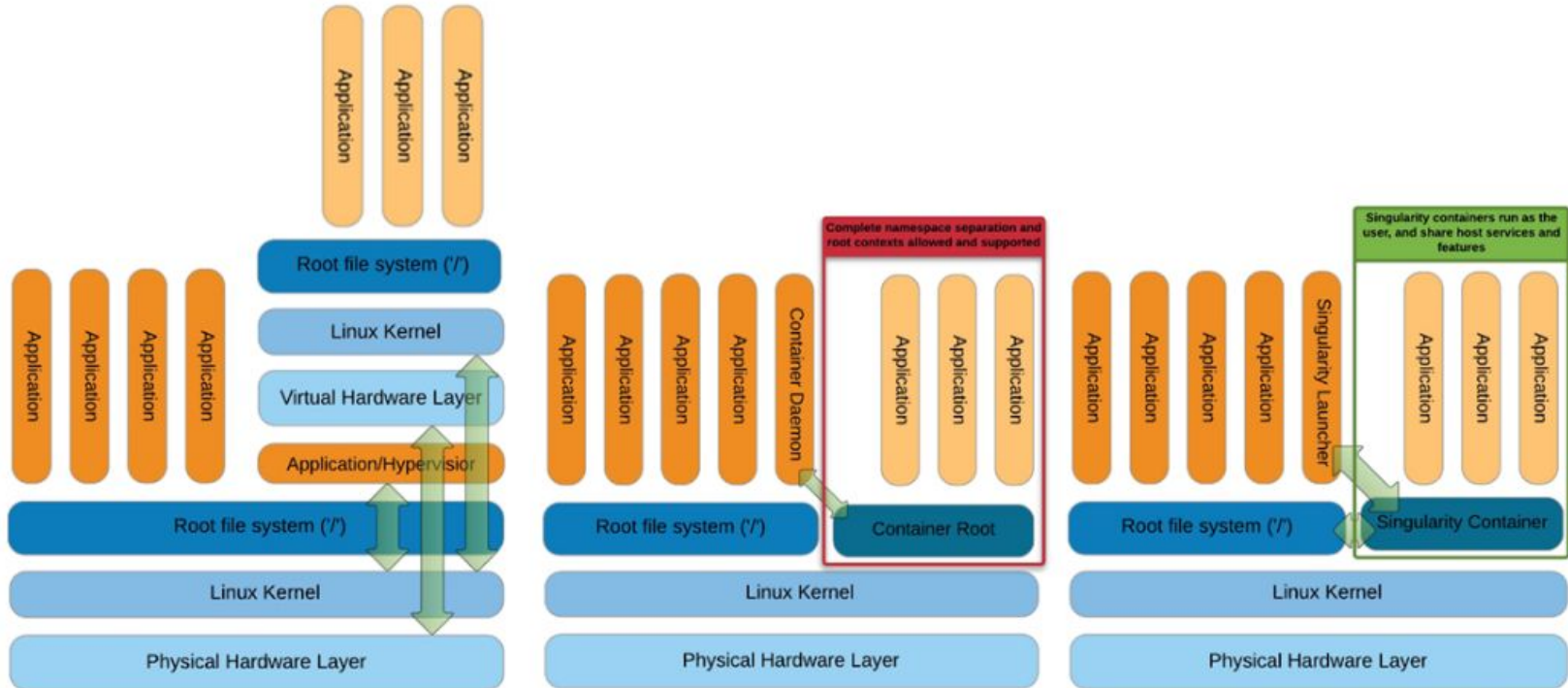
Linux
KVM



openstack®

- Does the traditional virtualization architecture scale beyond the limits of a single server?
- Does the traditional virtualization architecture scale to thousands of independent OSes?
- How do the hyperscalers (Google, Amazon, Microsoft) achieved huge scale?

Containers - how?(3)



Containers - how? (4)

Hypescalars (Google, Amazon, Microsoft) have modified older ideas to introduce the concept of a software container as an evolved virtualization technology by:

- Removing the overhead of the guest operating system
- Employing three key technologies:
 - [Kernel namespaces](#)
 - [cgroups](#)
 - [Union filesystems](#)

The goal is to produce a **virtualization technology** that simultaneously allows:

- Scalability (tenths of thousands of containers instances)
- Performance (near bear metal)
- Reproducibility (package the software in your own conditions/distro and run it in any Linux distro)

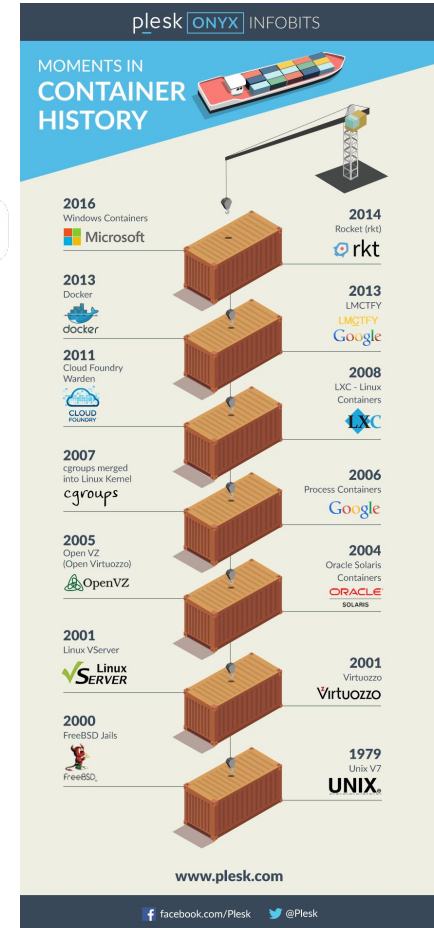
Containers - History

Container virtualization is not a new idea:

- Year 2000: FreeBSD jails
- Year 2004: Solaris Containers
- Year 2008: LXC - Linux Containers
- Year 2013: Docker
- Year 2015: Singularity
- Year 2016: Windows Containers

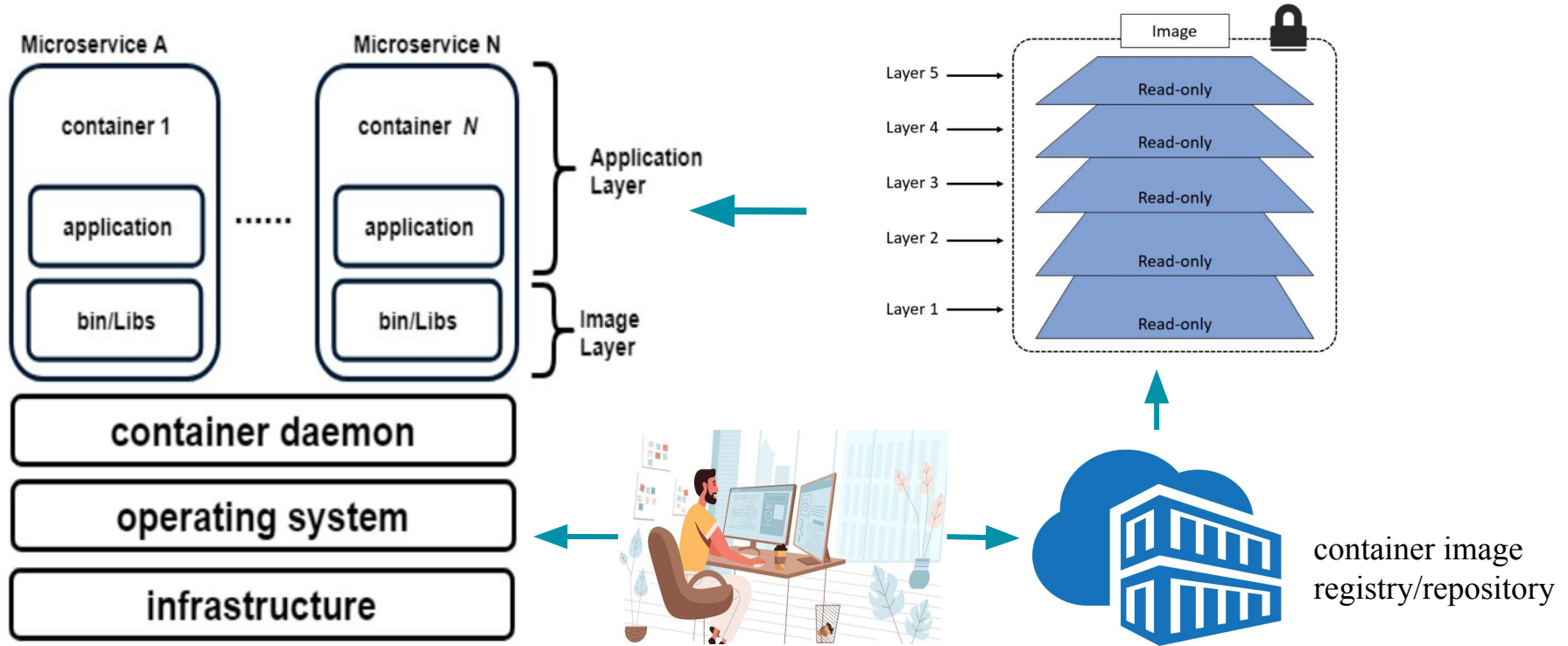


2015



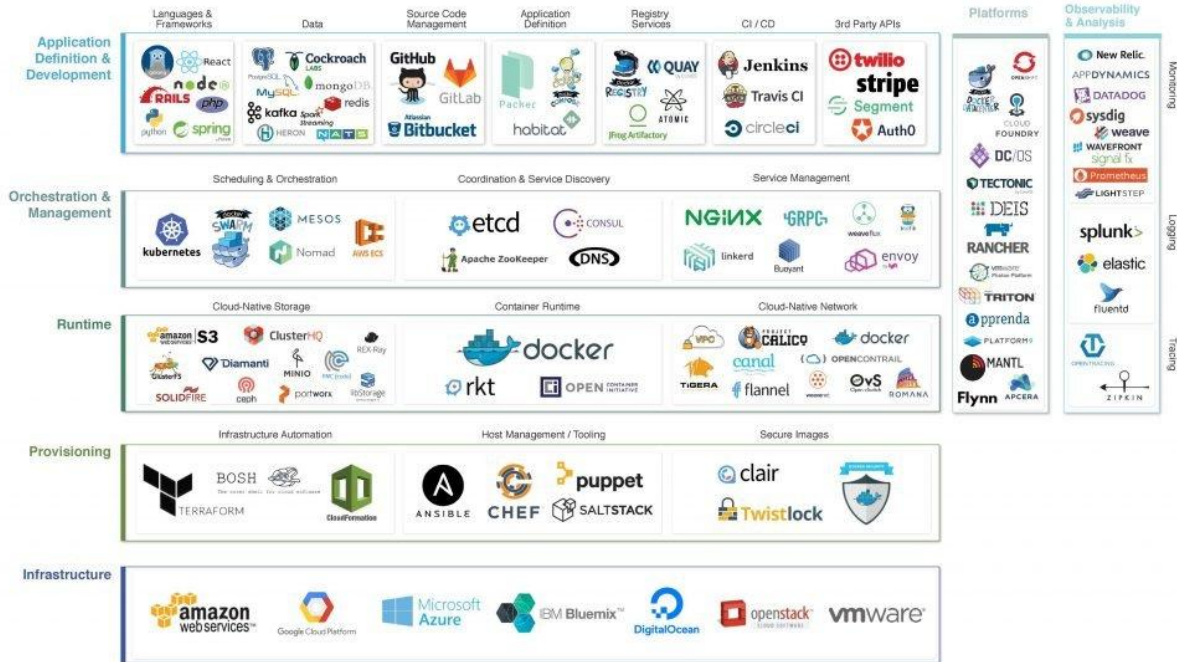
Norwegian Meteorological Institute

Containers - software ecosystem



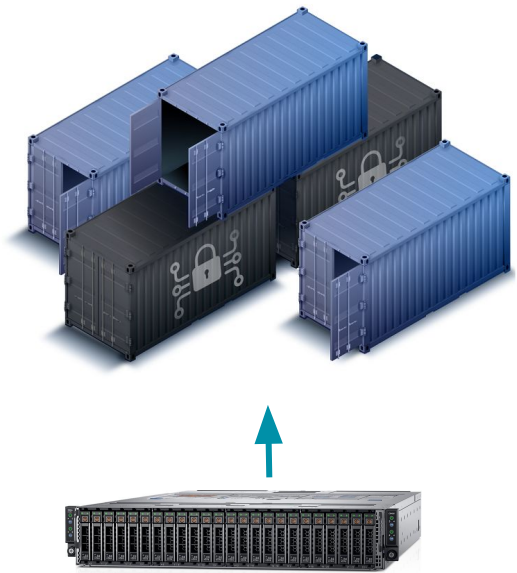
Containers - software ecosystem

Cloud Native Landscape v0.9.2



<http://github.com/cncf/landscape> @dankohn1 @lennypruss

Containers - concepts

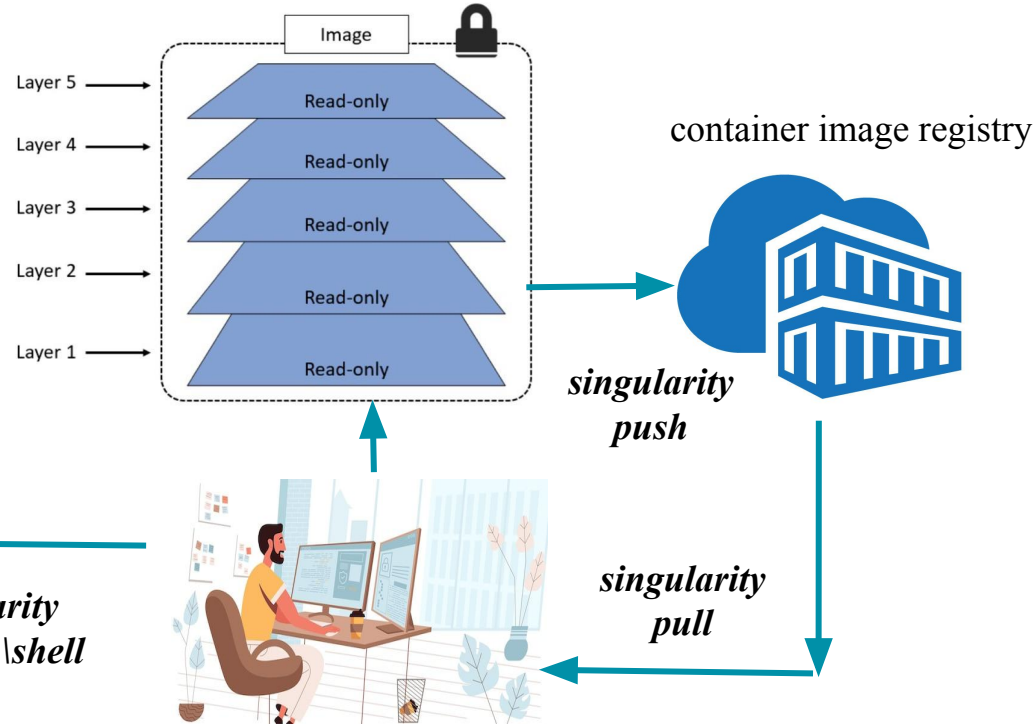


- Containers are virtualized software constructs that use the host operating system **kernel** resources but they package their own distro specific libraries (ex. run on RHEL8 an application packaged in Ubuntu)
- They are short lived, they run normally one or few applications. Once the execution of the application is complete, they die, no resources are consumed.
- A container image is the blueprint of the container. A running container is the instantiation of that image.
- Scalability and non persistence are their main characteristics.

Containers - images push,pull,exec

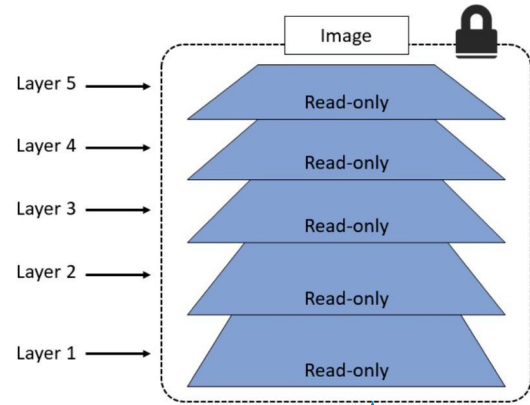


singularity
exec|run|shell



Containers - image recipes

container image recipe



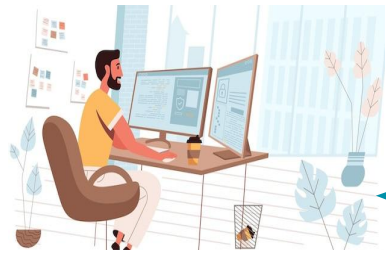
Bootstrap: docker
From: ubuntu:20.04

```
%post  
  apt-get -y update && apt-get  
  install -y python
```

```
%runscript  
  python -c 'print("Hello  
  World! Hello from our custom  
  Singularity image!")'
```



singularity
exec|run|shell



sudo
singularity
build

Containers - image repositories



<https://hub.docker.com/>



[Sylabs cloud](#)

Image repositories:

- Promote code re-usability
- Search them to find relevant software
- You can choose all up-to-date linux distros
- Each repository normally has images of a specific format:
 - Docker
 - Singularity
- It is possible to convert between image formats (ex. Docker -> Singularity format)

Beware with repositories of the following:

- Code will run but might not produce the results that you want
- Choose well established/official images (security)
- When uncertain, build your own image from up-to-date linux distributions

Singularity on PPI - reasons to select

Who uses Singularity for HPC workloads today?



- Fugaku - RIKEN - No.2 in the June 2022 Top500 list



- Summit - Oak Ridge National Lab - No.4 in the June 2022 Top500 list



- LUMI - CSC - No.3 in the June 2022 Top500 list

Norwegian Research
Infrastructure Services

Singularity on PPI - internal users

Who uses Singularity for HPC workloads at MET? (Introduced in PPI August 2021)



- yr team



- Various satellite processing data as part of projects/products/suites like:
 - camsatrec
 - arome-arctic
 - wave

Singularity on PPI - reasons to select(2)

In short, Singularity is made for HPC because:

- It permits container execution/manipulation without superuser privileges.
- It can accommodate/cooperate with a number of essential HPC technologies: MPI, OpenMPI, queue, GPU and parallel filesystems.
- It can provide substantial interoperability with other container runtime environments (i.e. Docker)
- **Reproducibility**: Lots of applications are now published containerized: The authors choose to encapsulate the applications and their lib dependencies so that the users can run them exactly as the developers intended.

Singularity on PPI - All aboard



Chance for a coffee/tea refill before the practical

Singularity on PPI - load the module

SSH login to a PPI login node. Use the PPI module system from any login/compute node (bionic,centos7,rhel8), interactively:

```
(base) georgiosm@ppi-blogin-a1:~$ module load singularity/3.7.3
(base) georgiosm@ppi-blogin-a1:~$ module list
Currently Loaded Modulefiles:
 1) go/1.16.4   2) singularity/3.7.3
```

```
(base) georgiosm@ppi-cllogin-a1:~$ module load singularity/3.7.3
(base) georgiosm@ppi-cllogin-a1:~$ module list

Currently Loaded Modules:
 1) go/1.16.4   2) singularity/3.7.3
```

```
[root@c6525-2y6sjb3-ah-compute ~]# module use /modules/MET/rhel8/user-modules/
[root@c6525-2y6sjb3-ah-compute ~]# module load singularity/3.9.4
Loading singularity/3.9.4
  Loading requirement: go/1.17.6
[root@c6525-2y6sjb3-ah-compute ~]# █
```

Singularity on PPI - run and inspect

- Run a ready made container with the *singularity run* command:

```
(base) georgiosm@c6525-2y6sjb3-ah-compute:~$ singularity run /home/singularityimages/demo2022/mycontainer.sif
Hello World! Hello from our custom Singularity image!
(base) georgiosm@c6525-2y6sjb3-ah-compute:~$
```

- What kind of container image are we dealing with (*singularity inspect*):

```
(base) georgiosm@c6525-2y6sjb3-ah-compute:~$ singularity inspect /home/singularityimages/demo2022/mycontainer.sif
org.label-schema.build-arch: amd64
org.label-schema.build-date: Tuesday_6_September_2022_13:3:12_CEST
org.label-schema.schema-version: 1.0
org.label-schema.usage.singularity.deffile.bootstrap: docker
org.label-schema.usage.singularity.deffile.from: ubuntu:20.04
org.label-schema.usage.singularity.version: 3.8.7-2.fc35
(base) georgiosm@c6525-2y6sjb3-ah-compute:~$ cat /etc/os-release
NAME="Red Hat Enterprise Linux"
VERSION="8.5 (Ootpa)"
ID="rhel"
ID_LIKE="fedora"
VERSION_ID="8.5"
PLATFORM_ID="platform:el8"
PRETTY_NAME="Red Hat Enterprise Linux 8.5 (Ootpa)"
```

Looks like we are dealing with an Ubuntu container whereas our Singularity host is a RHEL 8 system!

Singularity on PPI - exec and shell

- Are we indeed talking to an Ubuntu packed container? (singularity exec)

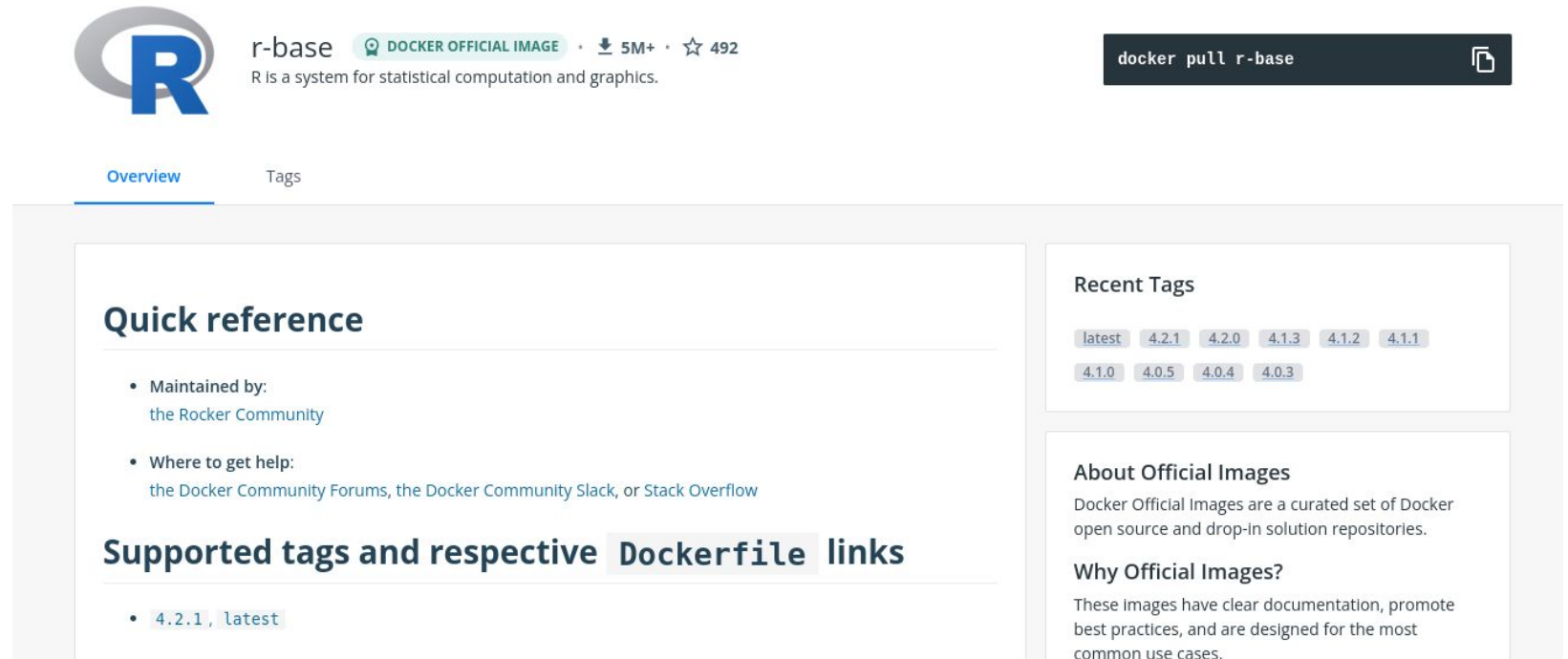
```
(base) georgiosm@c6525-2y6s3b3-ah-compute:~$ singularity exec /home/singularityimages/demo2022/mycontainer.sif cat /etc/os-release
NAME="Ubuntu"
VERSION="20.04.5 LTS (Focal Fossa)"
ID=ubuntu
ID_LIKE=debian
PRETTY_NAME="Ubuntu 20.04.5 LTS"
VERSION_ID="20.04"
HOME_URL="https://www.ubuntu.com/"
SUPPORT_URL="https://help.ubuntu.com/"
BUG_REPORT_URL="https://bugs.launchpad.net/ubuntu/"
PRIVACY_POLICY_URL="https://www.ubuntu.com/legal/terms-and-policies/privacy-policy"
VERSION_CODENAME=focal
UBUNTU_CODENAME=focal
```

- Can we keep alive the running container a bit longer? (singularity shell)

```
(base) georgiosm@c6525-2y6s3b3-ah-compute:~$ singularity shell /home/singularityimages/demo2022/mycontainer.sif
Singularity> python
Python 2.7.18 (default, Jul 1 2022, 12:27:04)
[GCC 9.4.0] on linux2
Type "help", "copyright", "credits" or "license" for more information.
>>>
Singularity> python -c 'print("Hello, I am still inside the container!")'
Hello, I am still inside the container!
Singularity> exit
exit
(base) georgiosm@c6525-2y6s3b3-ah-compute:~$
```

Singularity on PPI - pull a docker image

Say that you were browsing around Docker Hub and you found the following image:



The screenshot shows the Docker Hub interface for the 'r-base' image. At the top left is the R logo, followed by the text 'r-base' and 'DOCKER OFFICIAL IMAGE'. Below this, it says 'R is a system for statistical computation and graphics.' To the right, there are statistics: '5M+' downloads and '492' stars. A dark button with the text 'docker pull r-base' and a copy icon is visible. Below the header, there are two tabs: 'Overview' (selected) and 'Tags'. The main content area is divided into two columns. The left column has a 'Quick reference' section with two bullet points: 'Maintained by: the Rocker Community' and 'Where to get help: the Docker Community Forums, the Docker Community Slack, or Stack Overflow'. Below this is a section titled 'Supported tags and respective Dockerfile links' with a single bullet point: '4.2.1, latest'. The right column has a 'Recent Tags' section with a grid of tags: 'latest', '4.2.1', '4.2.0', '4.1.3', '4.1.2', '4.1.1', '4.1.0', '4.0.5', '4.0.4', and '4.0.3'. Below the tags is an 'About Official Images' section with the text: 'Docker Official Images are a curated set of Docker open source and drop-in solution repositories.' and a 'Why Official Images?' section with the text: 'These images have clear documentation, promote best practices, and are designed for the most common use cases.'

Singularity on PPI - pull a docker image (2)

To obtain (and convert in one step) that image, use singularity pull: *singularity pull docker://r-base*

```
(base) georgiosm@6525-2y6sjb3-ah-compute:~$ singularity pull docker://r-base
INFO:   Converting OCI blobs to SIF format
INFO:   Starting build...
Getting image source signatures
Copying blob df98203a8b21 done
Copying blob a226d216a623 done
Copying blob d06267fca963 done
Copying blob 7aff5dbd7edf done
Copying blob 05f509fe31e6 done
Copying blob 0cbc1ba94271 done
Copying config e1706c1575 done
Writing manifest to image destination
Storing signatures
2022/09/06 15:56:34 info unpack layer: sha256:df98203a8b21cb36e308dd353908ed338642491ef167d757bebb3297146a77a4
2022/09/06 15:56:34 warn xattr{etc/gshadow} ignoring ENOTSUP on setattr "user.rootlesscontainers"
2022/09/06 15:56:34 warn xattr{/home/SINGULARITYTMP/build-temp-2738880293/rootfs/etc/gshadow} destination filesystem does not support xattrs, further warnings will be suppressed
2022/09/06 15:56:44 info unpack layer: sha256:a226d216a62319dc0bd33f9f2ab068e10914c1ea87f26a9af2a505fb3407c92a
2022/09/06 15:56:44 warn xattr{etc/gshadow} ignoring ENOTSUP on setattr "user.rootlesscontainers"
2022/09/06 15:56:44 warn xattr{/home/SINGULARITYTMP/build-temp-2738880293/rootfs/etc/gshadow} destination filesystem does not support xattrs, further warnings will be suppressed
2022/09/06 15:56:44 info unpack layer: sha256:d06267fca963eae8a21b9c01db4027c0fce62c50c1b2af7e1b64ee3831395be9
2022/09/06 15:56:46 warn xattr{/var/cache/apt/archives/partial} ignoring ENOTSUP on setattr "user.rootlesscontainers"
2022/09/06 15:56:46 warn xattr{/home/SINGULARITYTMP/build-temp-2738880293/rootfs/var/cache/apt/archives/partial} destination filesystem does not support xattrs, further warnings will be suppressed
2022/09/06 15:56:46 info unpack layer: sha256:7aff5dbd7edf267d009f7431ab32808362abb6d08b12d8e2b754e28f178bdb02
2022/09/06 15:56:47 info unpack layer: sha256:05f509fe31e626b0bb83a3c72375959cb5bead186d6743a524ae5ba3f7ab35d0
2022/09/06 15:56:47 info unpack layer: sha256:0cbc1ba94271ad07911f284bcda964616faab47cbcab0c3aab7a25ca65d35a86
2022/09/06 15:56:59 warn xattr{/usr/local/share/fonts} ignoring ENOTSUP on setattr "user.rootlesscontainers"
2022/09/06 15:56:59 warn xattr{/home/SINGULARITYTMP/build-temp-2738880293/rootfs/usr/local/share/fonts} destination filesystem does not support xattrs, further warnings will be suppressed
INFO:   Creating SIF file...
```

This should create an **r-base_latest.sif** file (approx. 309 Megabytes) which is the converted image.

Singularity on PPI - pull a docker image (3)

You can now run the pulled and converted Docker Hub image with **singularity shell** and call **R** interactively:

```
(base) georgiosm@c6525-2y6s3b3-ah-compute:~/demoSEP2022$ singularity shell r-base_latest.sif
Singularity> R

R version 4.2.1 (2022-06-23) -- "Funny-Looking Kid"
Copyright (C) 2022 The R Foundation for Statistical Computing
Platform: x86_64-pc-linux-gnu (64-bit)

R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

  Natural language support but running in an English locale

R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

>
Save workspace image? [y/n/c]: n
Singularity> exit
exit
(base) georgiosm@c6525-2y6s3b3-ah-compute:~/demoSEP2022$
```

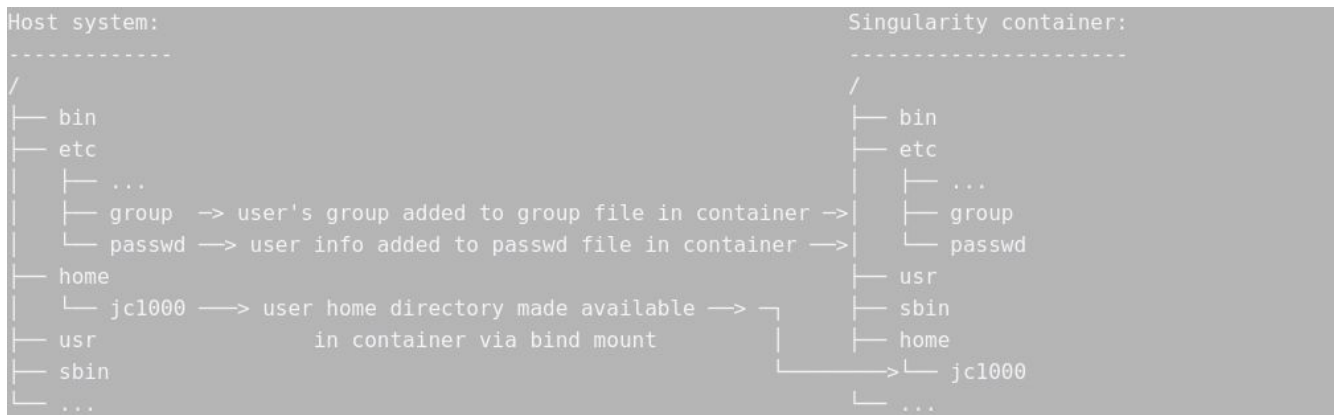
Notes:

- This is the container packaged R
- Control-D will exit from R

Singularity on PPI - Folder/volume binding

- Containers are not persistent. They execute, finish their job and die, so any computation results, if not saved, will be gone forever.
- Container images are/should be small. One should never dist/pack large datasets as part of the image

For these reasons, there needs to be a way for containers to map/bind to the local filesystems outside their realm. This is called **container folder/volume binding**. Some directories are automatically bound by default (example \$HOME), when the container runs. However, for other non standard directories folders, you need to instruct Singularity what to bind (from the host filesystems) and where (inside the container).



Singularity on PPI - Folder/volume binding (2)

Let's create a realistic container computation scenario where the input data do not reside inside the container and the output will be saved in the host filesystem (your *\$HOME*):

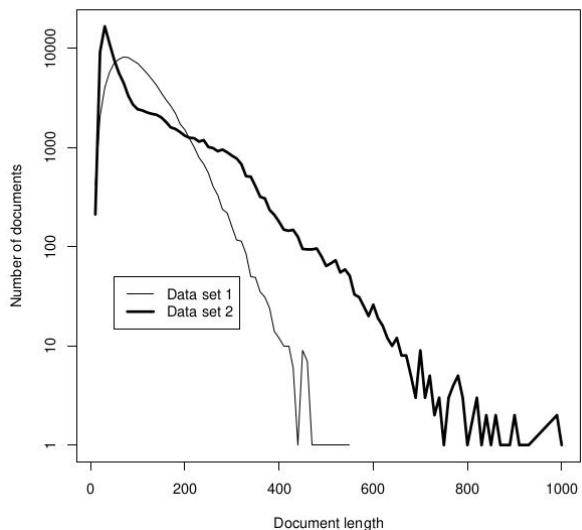
```
(base) georgiosm@c6525-2y6sjb3-ah-compute:~$ cp -r /home/singularityimages/demo2022/rexample ./
(base) georgiosm@c6525-2y6sjb3-ah-compute:~$ ls -la ./rexample/
total 24
drwxrwxr-x  2 georgiosm georgiosm  78 Sep  6 16:04 .
drwx----- 60 georgiosm georgiosm 8192 Sep  6 16:04 ..
-rw-rw-r--  1 georgiosm georgiosm  386 Sep  6 16:04 log-d1.data
-rw-rw-r--  1 georgiosm georgiosm  686 Sep  6 16:04 log-d2.data
-rw-rw-r--  1 georgiosm georgiosm  707 Sep  6 16:04 myscript.R
```

The *myscript.R* file aims to instruct the R instance inside the container of our Docker pulled image (slides 21-22) to plot document length versus the number of documents for two data set files *log-d1.data* and *log-d2.data* .

Singularity on PPI - Folder/volume binding (3)

We will now employ container folder/volume binding to tell our container where to find the data to read/write:

```
(base) georgiosm@c6525-2y6sjb3-ah-compute: $ singularity exec -B ./reexample/:/data r-base_latest.sif Rscript /data/myscript.R  
[1] "t-test pvalue = 0.0355339379646605"
```



host
filesystem

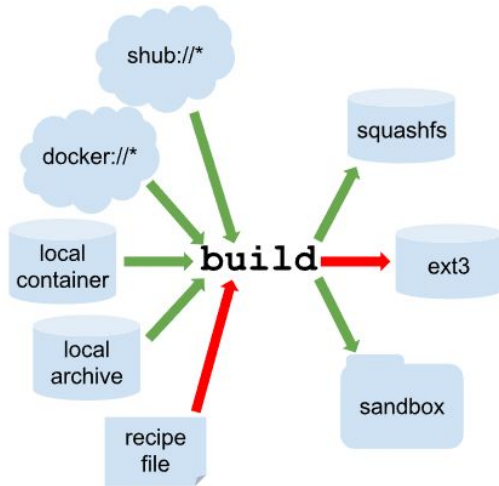
container
filesystem

Resulting **Rplots.pdf** file

Singularity on PPI - Image building

Container repositories are great tools for code reuse and effort saving. They might not be a good choice if:

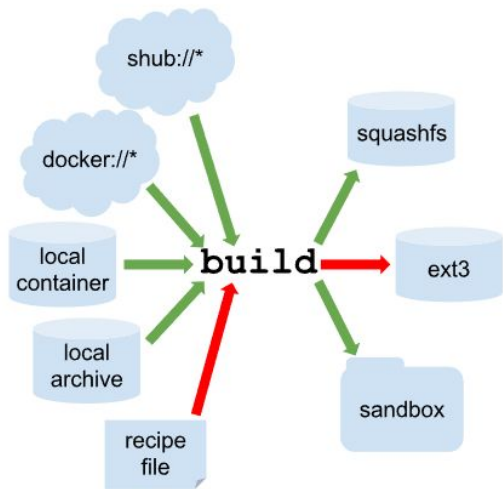
- You need to build something from scratch
- You need to performance optimize something
- You have strict cyber security or licensing/SLA requirements



In all of these cases, you will need to go down the path of making an image from scratch, by means of a recipe file. Singularity has the **singularity build** command for that purpose.

- For security reasons that have to do with subuid/subgid manipulation, the PPI Singularity environment currently **does not allow** the use of this command.
- Do it in your laptop/VM server.

Singularity on PPI - Image building (2)



There are many ways to build images including:

- Command line tools on your workstation
- Web services of public hubs

Preference should always be given to command line tools on your workstation. You can build the image there and then scp it to the PPI environment, under:

`/home/singularityimages`

- **Always build from official distro container images**
- **Good if you shasum/checksum your image**
- **Good if you use internal institutional container repositories**
 - **registry.met.no****to register your images.**



Norwegian
Meteorological
Institute

Questions?

georgiosm@met.no

<https://twitter.com/gmagklaras?lang=en>