



**imgw**

Institut für Meteorologie  
und Geophysik



**universität  
wien**

Faculty of Earth Sciences,  
Geography and Astronomy

# Singularity Examples

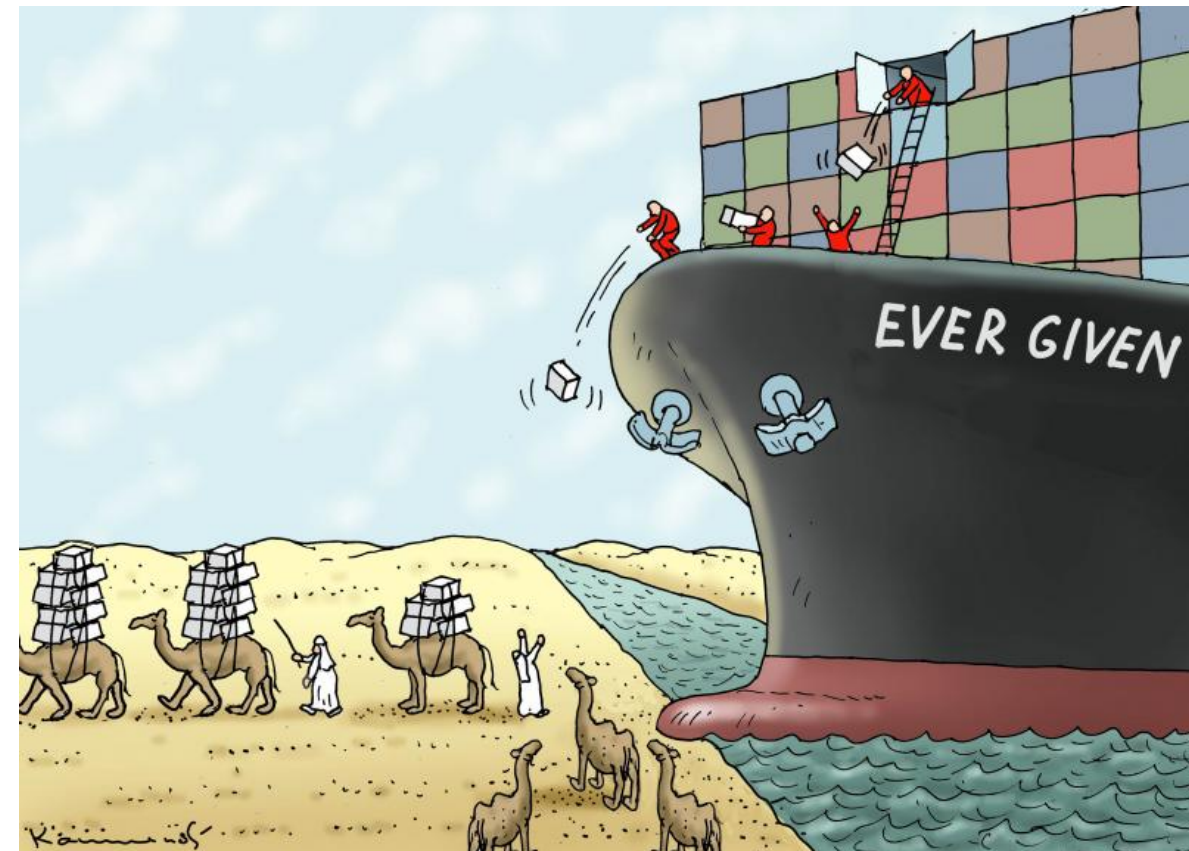


**User Workshop on Singularity on HPC**  
**24.11.2021, MB**

# Agenda

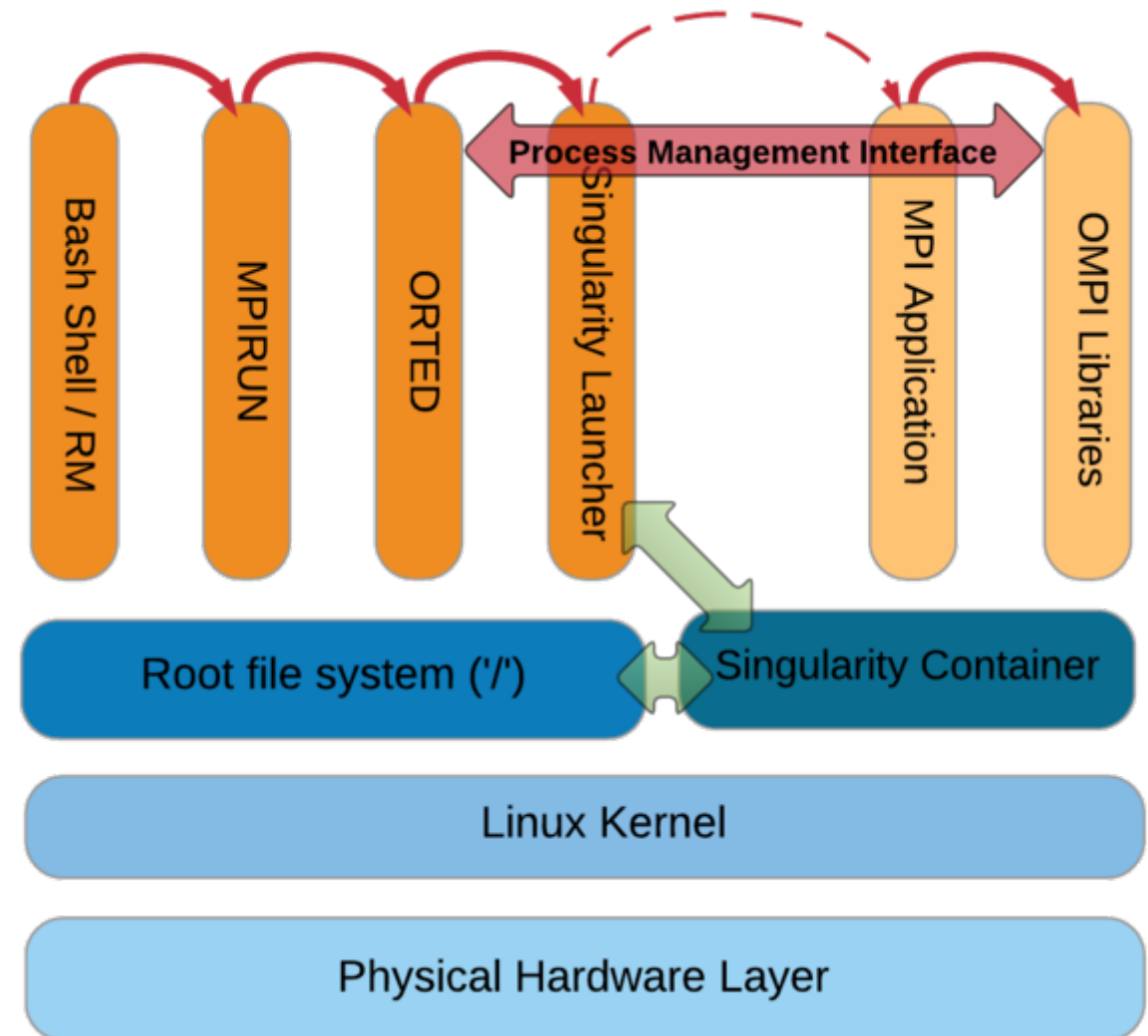


- Introduction
  - ~~What is a container~~
  - ~~Difference to docker~~
  - ~~Security on HPC + MPI + Infiniband~~
  - ~~Building a container~~
- ~~Hands on~~
  - ~~Build a container~~
  - ~~Modify a definition files / customized image~~
- Examples & Problems
  - MPI
  - Performance
  - Cloud.sylabs.io
  - Build service, overlay, signing, best practices



# Singularity & MPI

- Use same **Message Passing Interface** (MPI) distribution and **version** within container as would be used outside the container.
- MPICH, OpenMPI, Intel MPI,...
- If using **Infiniband** (IB), install same OFED drivers and libraries inside the container as used on underlying **HPC hardware**.



# Singularity & MPI

- Example in [Singularity.ubuntu-18.04-OMPI-gcc](#)

```
mpirun -np X singularity exec ubuntu-OMPI.sif /opt/mpitest
```

- Use the Host MPI (module load openmpi)
- Versions need to be feature compatible
- Slurm Job like this

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

```
#SBATCH --job-name singularity-mpi
```

```
#SBATCH -N $NNODES # total number of nodes
```

```
#SBATCH --time=00:05:00 # Max execution time
```

```
module load singularity
```

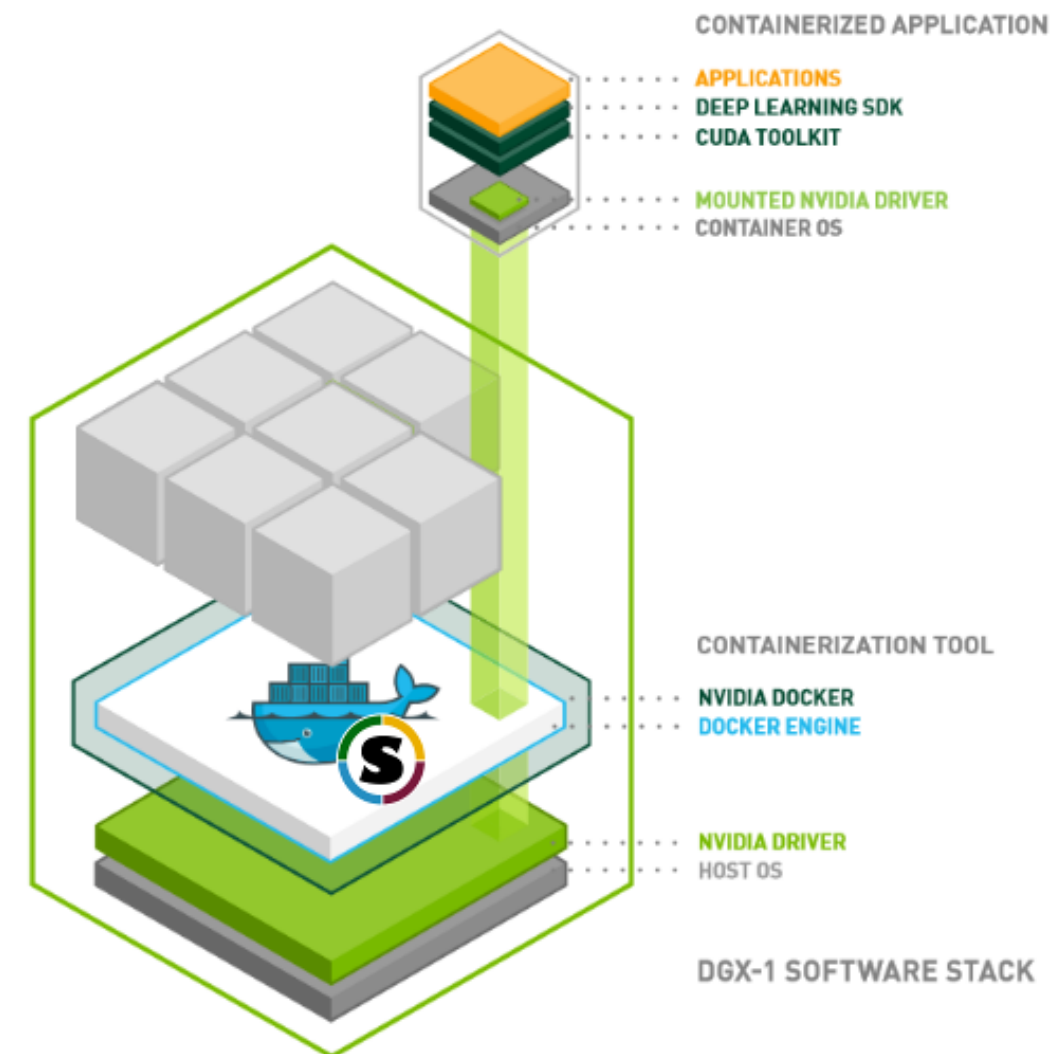
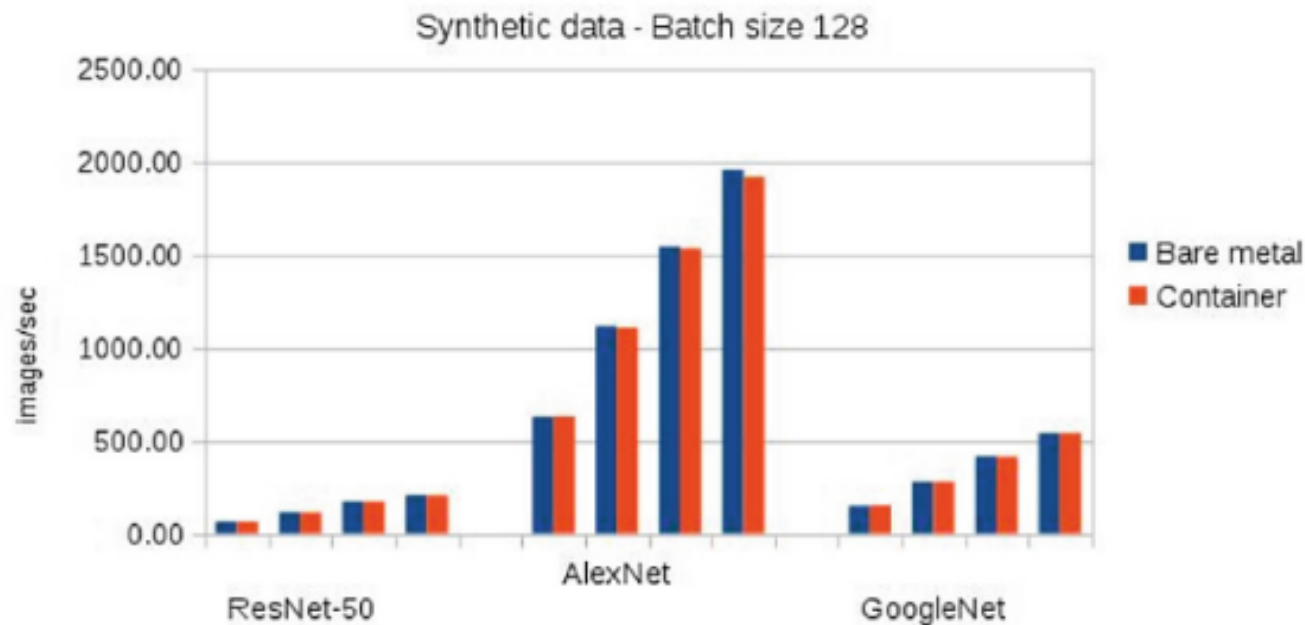
```
module load openmpi
```

```
mpirun -n $NP singularity exec ompitest.sif /opt/mpitest
```

# Singularity & GPUs

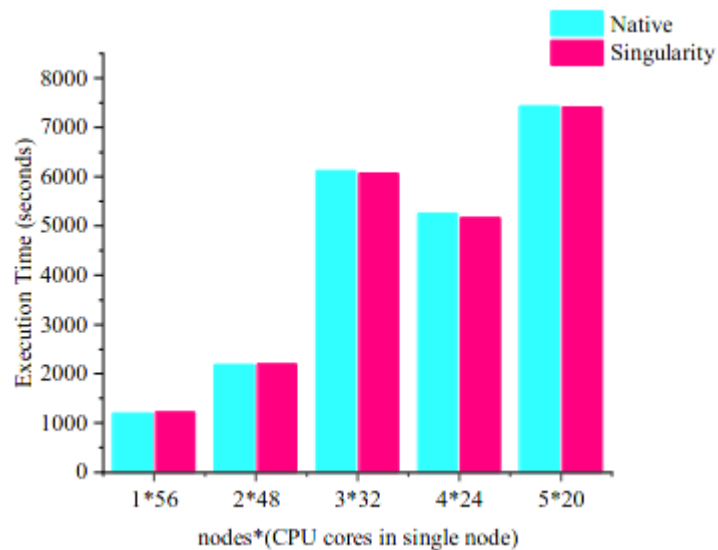
- Direct access, builtin
- `--nv` or `--mroc`

GALILEO@CINECA - Training with NVIDIA Testa K80 - 1, 2, 3 and 4 GPUs

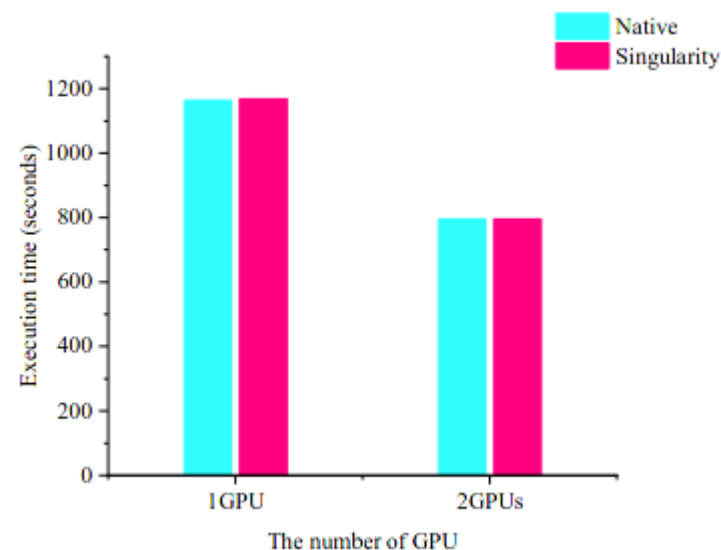


**Fig. 3.** Number of images per second computed in ResNet-50, AlexNet, and GoogleNet model for 1, 2, 3 and 4 K80 NVIDIA GPUs on a single GALILEO node. The batch size used is 128, the dataset is ImageNet - synthetic.

# Performance



CPU – Nodes - MP



GPUs

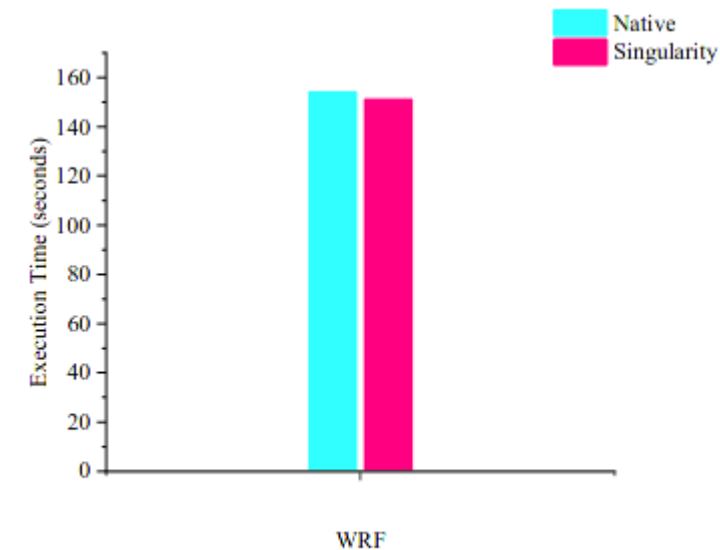


Fig. 10. Comparison of WRF performance overhead

WRF

- Admin can modify control over the system → HPC
- User inside == user outside
- Bind points, between Host and Guest
- No daemon processes (e.g. docker)



# Sylab & Signing

- Library of containers
- Remote Builder
- Keystore to sign containers
- Making everything reproducible @science

```
02:32 PM $ singularity verify lolcow_latest.sif
Container is signed by 1 key(s):

Verifying signature F: 4426F6CBCB38449E062940A42E6A34D6F54EBE59:
[REMOTE] Dave Godlove (this is the best key) <d@syllabs.io>
[OK] Data integrity verified

INFO: Container verified: lolcow_latest.sif
```



The screenshot displays the Sylabs.io website. At the top, there's a navigation bar with links for Home, Singularity Library, Remote Builder, and Keystore. Below this, the main heading reads "Singularity Library" and "Remote Builder". A sub-header states: "Singularity 3.0 introduces the ability to build a container in the cloud. Singularity user does not need to prepare an environment or assign permission. Remote Builder can build a container using a provided build definition file."

Two main content areas are visible:

- Build a Recipe:** This section prompts users to attach a build recipe by dragging & dropping or pasting from the clipboard. It includes a code editor with a sample recipe:

```
1 Bootstrap: docker
2 From: alpine:latest
3
4 %runscript
5
6 echo "Hello World!"
```
- Using the Remote Builder from the Singularity CLI:** This section provides instructions on how to use the Remote Build Service, including a command to get a cloud token and a terminal snippet showing the build command:

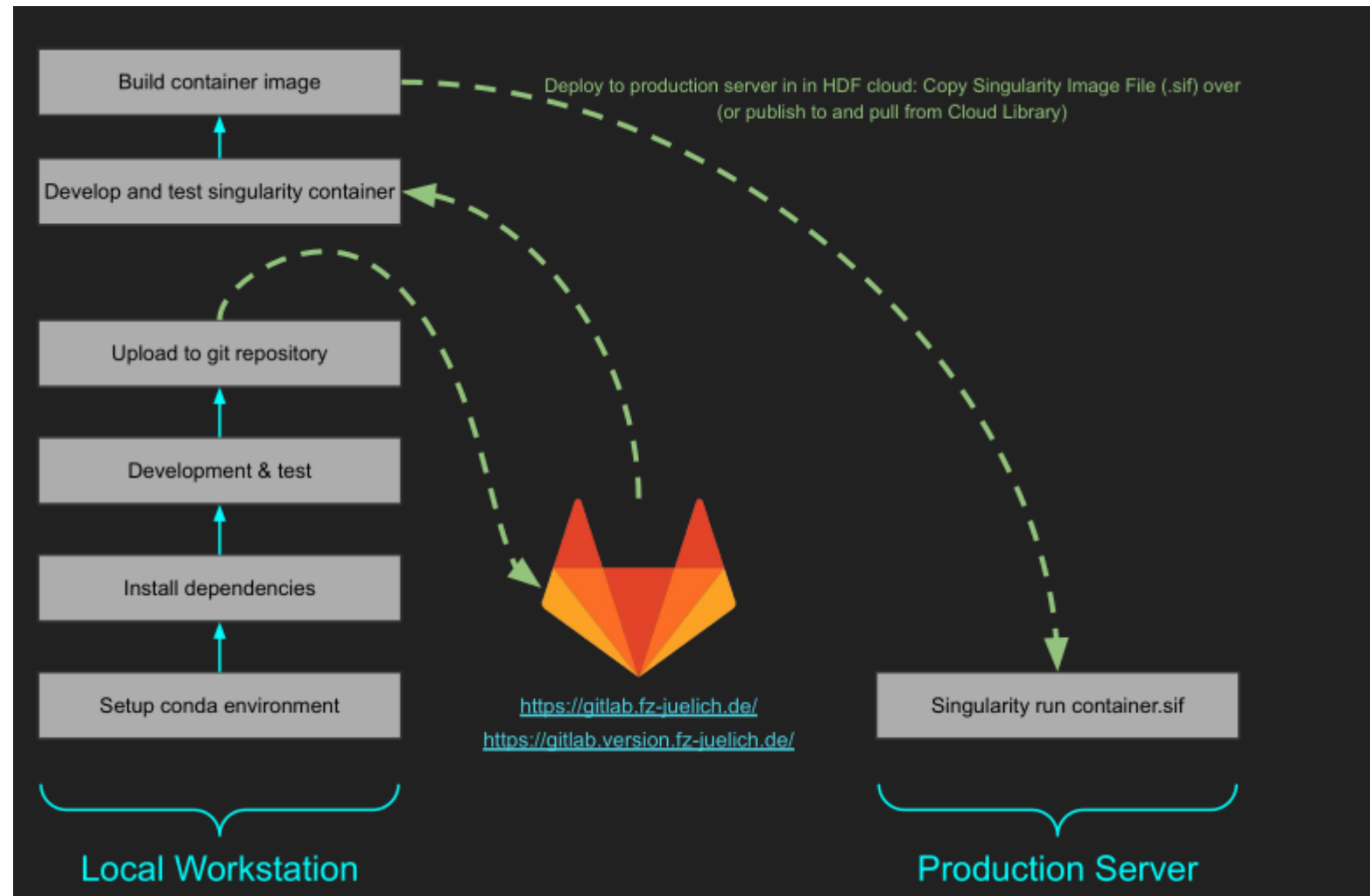
```
singularity build --remote output.sif examples/docker
/Singularity
```

At the bottom, there's a "Build" button and a search bar.



# Workflow

- Develop
- Test
- Build
- Deploy
- Run
- Share



# Best Practices

- Use existing containers that you trust
- Add just enough to make your app work ( - - sandbox for debug, - -writable)
- Do not try to fit everything into one
- Add documentation, versioning, git
- X11-Applications, can work too.
- Try to use an OS that is close to what you are running on
- Bind paths to your container @ runtime
- Consider a Build and Run container for faster turnaround



**imgw**

Institut für Meteorologie  
und Geophysik

**VIENNA  
SCIENTIFIC  
CLUSTER**

 **Sylabs.io**



**universität  
wien**

Faculty of Earth Sciences,  
Geography and Astronomy



# Questions?

## Singularity Workshop

Gitlab: <https://gitlab.phaidra.org/imgw/singularity>

