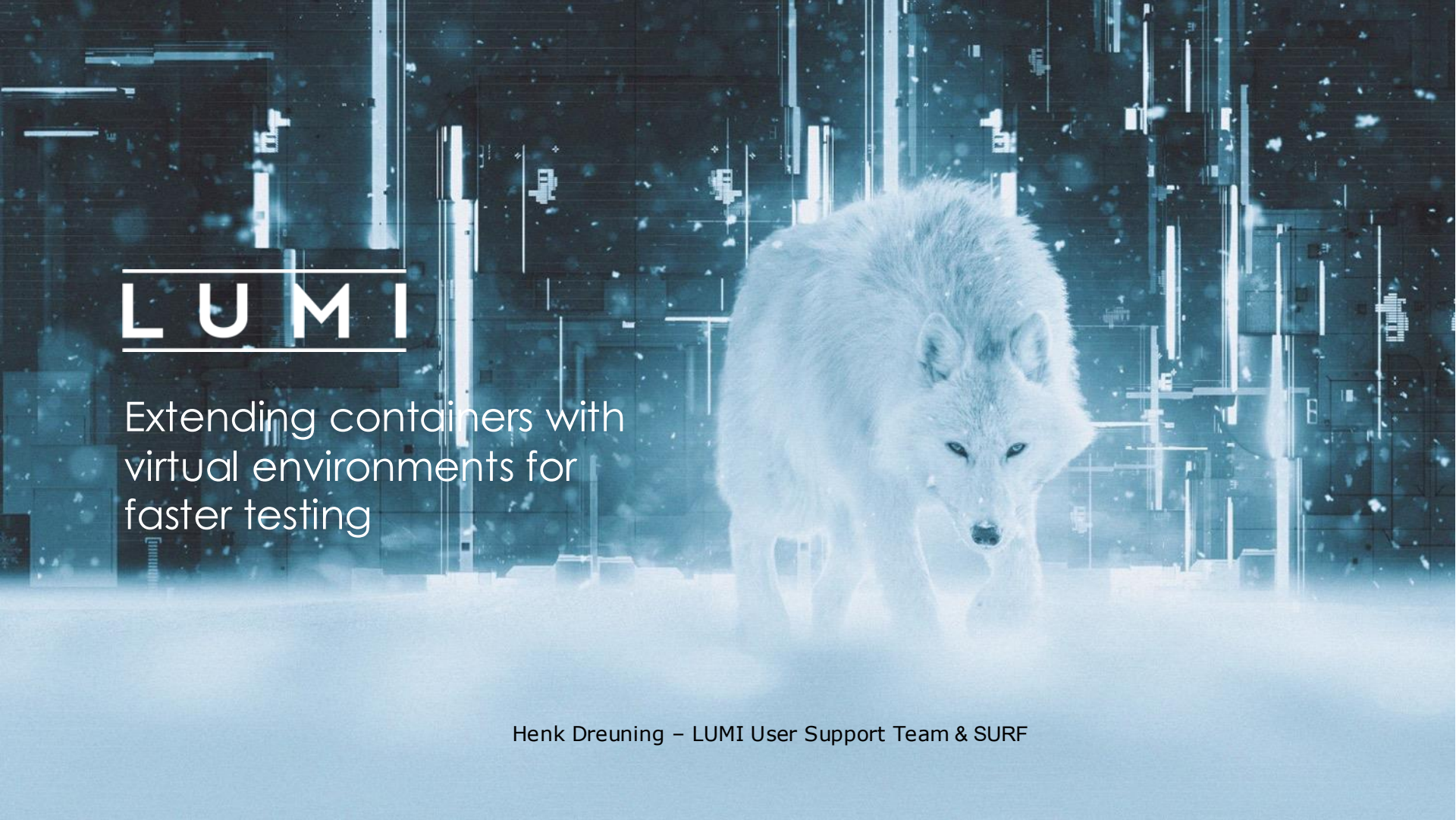


Extending containers with virtual environments for faster testing

Henk Dreuning – LUMI User Support Team & SURF

Motivation

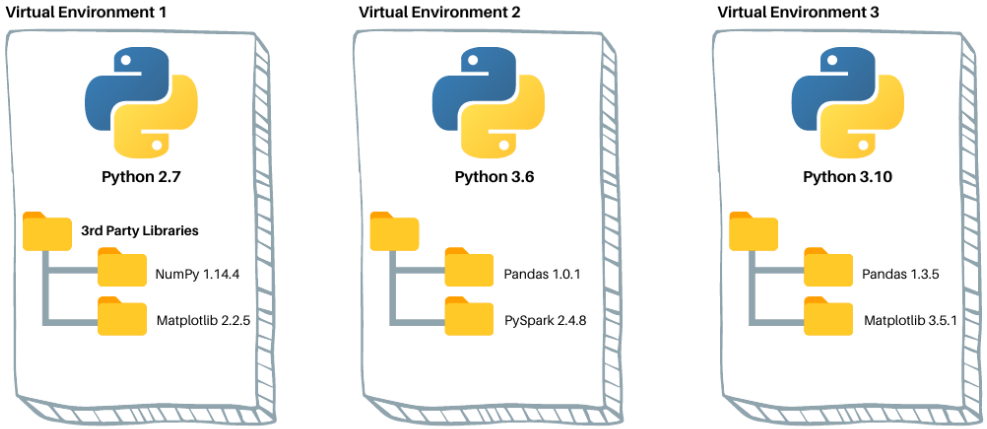
LUMI

Cotainr is great, but building a container takes time --> not ideal for quick testing / iterating on your project

Virtual environments offer a quick (and easy) way of installing additional packages to existing containers

What are virtual environments

A virtual environment is a folder tree containing a specific Python version, third-party libraries, and other scripts.



Source: www.dataquest.io/blog/a-complete-guide-to-python-virtual-environments/ dataquest.io

Virtual environments are conceptually similar to `conda` environments – just for `pip` only.

Requirements

We assume we already have a container built from a conda environment file. If not, we can build one via:

```
module load LUMI/25.03 cotainr

BASE_CONTAINER=/appl/local/laifs/containers/lumi-multitorch-u24r70f21m50t210-20260513_121430/lumi-multitorch-mpich-u24r70f21m50t210-20260513_121430.sif

cotainr build minimal_pytorch.sif --base-image=$BASE_CONTAINER --conda-env=minimal_pytorch.yml --accept-license
```

```
name: minimal_pytorch
channels:
  - conda-forge
dependencies:
  - filelock=3.29.0
  - fsspec=2026.4.0
  - jinja2=3.1.6
  - markupsafe=3.0.3
  - mpmath=1.3.0
  - networkx=3.6.1
  - numpy=2.4.6
  - pillow=12.2.0
  - pip=24.0
  - python=3.12.3
  - sympy=1.14.0
  - typing-extensions=4.15.0
  - pip:
    - --extra-index-url https://download.pytorch.org/whl/rocm7.0/
    - triton-rocm==3.6.0
    - torch==2.10.0+rocm7.0
    - torchaudio==2.10.0+rocm7.0
    - torchvision==0.25.0+rocm7.0
```

Run a shell inside the container

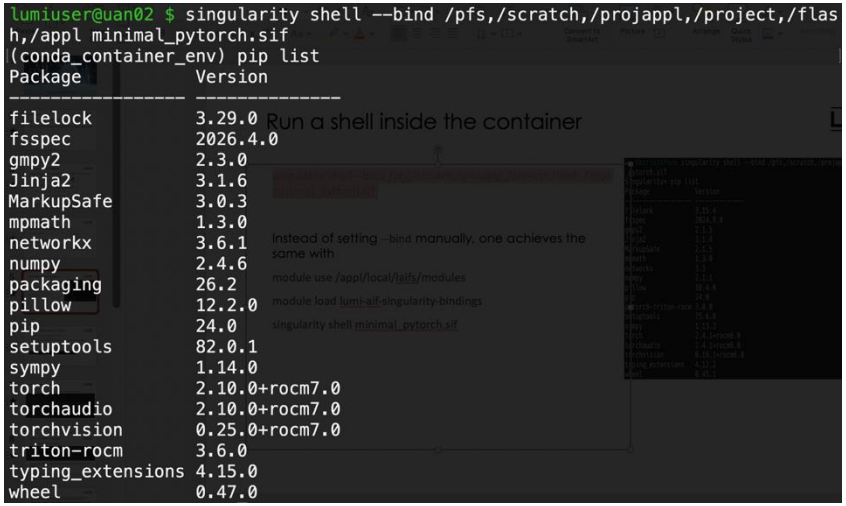
singularity shell --bind /pfs,/scratch,/projappl,/project,/flash,/appl minimal_pytorch.sif

Instead of setting --bind manually, one achieves the same with

module use /appl/local/laifs/modules

module load lumi-aif-singularity-bindings

singularity shell minimal_pytorch.sif



Create a virtual environment via venv

Inside the container, create a virtual environment via venv

```
python -m venv myenv --system-site-packages
```

The --system-site-packages flag gives the virtual environment access to the packages inside the container.

Activate the environment via

```
source myenv/bin/activate
```

```
(conda_container_env) python -m venv myenv --system-site-packages
(conda_container_env) source myenv/bin/activate
(myenv) (conda_container_env)
```

Install custom packages via pip

pip install torchmetrics

The new package will then be available alongside the packages in the container

```
(myenv) (conda_container_env) pip install torchmetrics
Collecting torchmetrics
  Downloading torchmetrics-1.9.0-py3-none-any.whl.metadata (23 kB)
Requirement already satisfied: numpy>1.20.0 in /opt/cotainr/conda/envs/conda_container_env/lib/python3.12/site-packages (from torchmetrics) (2.4.6)
Requirement already satisfied: packaging>17.1 in /opt/cotainr/conda/envs/conda_container_env/lib/python3.12/site-packages (from torchmetrics) (26.2)
Requirement already satisfied: torch>=2.0.0 in /opt/cotainr/conda/envs/conda_container_env/lib/python3.12/site-packages (from torchmetrics) (2.10.0+rocm7.0)
Collecting lightning-utilities>=0.15.3 (from torchmetrics)
  Downloading lightning_utilities-0.15.3-py3-none-any.whl.metadata (5.5 kB)
Requirement already satisfied: typing_extensions in /opt/cotainr/conda/envs/conda_container_env/lib/python3.12/site-packages (from lightning-utilities>=0.15.3->torchmetrics) (4.15.0)
Requirement already satisfied: filelock in /opt/cotainr/conda/envs/conda_container_env/lib/python3.12/site-packages (from torch>=2.0.0->torchmetrics) (3.29.0)
Requirement already satisfied: setuptools in /opt/cotainr/conda/envs/conda_container_env/lib/python3.12/site-packages (from torch>=2.0.0->torchmetrics) (82.0.1)
Requirement already satisfied: sympy>=1.13.3 in /opt/cotainr/conda/envs/conda_container_env/lib/python3.12/site-packages (from torch>=2.0.0->torchmetrics) (1.14.0)
Requirement already satisfied: networkx>=2.5.1 in /opt/cotainr/conda/envs/conda_container_env/lib/python3.12/site-packages (from torch>=2.0.0->torchmetrics) (3.6.1)
Requirement already satisfied: jinja2 in /opt/cotainr/conda/envs/conda_container_env/lib/python3.12/site-packages (from torch>=2.0.0->torchmetrics) (3.1.6)
Requirement already satisfied: fsspec>=0.8.5 in /opt/cotainr/conda/envs/conda_container_env/lib/python3.12/site-packages (from torch>=2.0.0->torchmetrics) (2026.4.0)
Requirement already satisfied: triton-rocm==3.6.0 in /opt/cotainr/conda/envs/conda_container_env/lib/python3.12/site-packages (from torch>=2.0.0->torchmetrics) (3.6.0)
Requirement already satisfied: mpmath<1.5,>=1.1.0 in /opt/cotainr/conda/envs/conda_container_env/lib/python3.12/site-packages (from sympy>=1.13.3->torch>=2.0.0->torchmetrics) (1.3.0)
Requirement already satisfied: MarkupSafe>=2.0 in /opt/cotainr/conda/envs/conda_container_env/lib/python3.12/site-packages (from jinja2->torch>=2.0.0->torchmetrics) (3.0.3)
Downloading torchmetrics-1.9.0-py3-none-any.whl (983 kB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 983.4/983.4 kB 2.5 MB/s eta 0:00:00
Downloading lightning_utilities-0.15.3-py3-none-any.whl (31 kB)
Installing collected packages: lightning-utilities, torchmetrics
Successfully installed lightning-utilities-0.15.3 torchmetrics-1.9.0
```

Location of installed packages

We can check the location of the installed files via

```
(myenv) (conda_container_env) python
Python 3.12.3 | packaged by conda-forge | (main, Apr 15 2024, 18:38:13) [GCC 12.3.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> import os
>>> import torchvision
>>> import torchmetrics
>>> os.path.abspath(torchvision.__file__)
'/opt/cotainr/conda/envs/conda_container_env/lib/python3.12/site-packages/torchvision/__init__.py'
>>> os.path.abspath(torchmetrics.__file__)
'/pfs/lustrep1/projappl/project_462000008/hdreuning/AI_workshop_2026/myenv/lib/python3.12/site-packages/torchmetrics/__init__.py'
```

The new package is installed in our virtual environment whereas the other packages are installed in the container.

Warning

LUMI

You should not stop here, as this way of installing python packages creates typically thousands of small files. This puts a lot of strain on the Lustre file system and might exceed your file quota.

Once you have a complete set of python packages and their versions, choose one of the following options:

- Create a new container with `cotainr` and delete virtual environment
- Turn `myenv` into a SquashFS file and bind mount it to the container

Option 1: Create a new container with cotainr

After having found all needed packages, add them to the conda environment file and create a new container:

```
module load LUMI/25.03 cotainr

BASE_CONTAINER=/appl/local/laifs/containers/lumi-multitorch-u24r70f21m50t210-20260513_121430/lumi-multitorch-mpich-u24r70f21m50t210-20260513_121430.sif

cotainr build updated_pytorch.sif --base-image=$BASE_CONTAINER --conda-env=updated_pytorch.yml --accept-license
```

The virtual environment should then be deleted:

```
rm -rf myenv
```

```
name: minimal_pytorch
channels:
  - conda-forge
dependencies:
  - filelock=3.29.0
  - fsspec=2026.4.0
  - jinja2=3.1.6
  - markupsafe=3.0.3
  - mpmath=1.3.0
  - networkx=3.6.1
  - numpy=2.4.6
  - pillow=12.2.0
  - pip=24.0
  - python=3.12.3
  - sympy=1.14.0
  - typing-extensions=4.15.0
  - pip:
    - --extra-index-url https://download.pytorch.org/whl/rocm7.0/
    - triton-rocm==3.6.0
    - torch==2.10.0+rocm7.0
    - torchaudio==2.10.0+rocm7.0
    - torchvision==0.25.0+rocm7.0
    - torchmetrics==1.9.0
```

Option 2: Turn myenv into a SquashFS file

Turn the myenv directory into a SquashFS file:

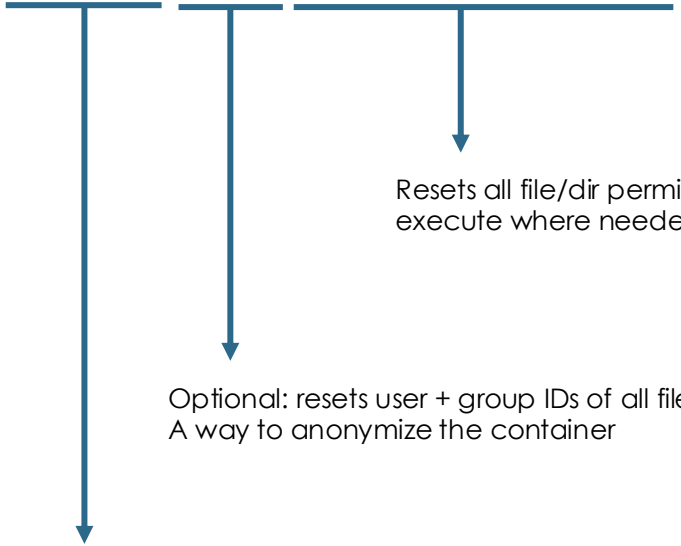
```
mksquashfs myenv myenv.sqsh
rm -rf myenv
```

Option 2: Turn myenv into a SquashFS file

Turn the myenv directory into a SquashFS file:

```
mksquashfs myenv myenv.sqsh -processors 1 -all-root -action "chmod(a+rX) @true"

rm -rf myenv
```



Resets all file/dir permissions to read (and execute where needed) for everybody

Optional: resets user + group IDs of all files to root
A way to anonymize the container

Use 1 core to keep load on login node low
Can use 1-16

Option 2: Turn myenv into a SquashFS file

LUMI

Bind mount it to the container:

```
export SINGULARITYENV_PREPEND_PATH=/user-software/bin
singularity exec -B myenv.sqsh:/user-software:image-src=/ minimal_pytorch.sif python my_script.py
```

This is much better for the file system as it regards the myenv.sqsh as a single file.

For advanced users:

This approach is compatible with packages that cannot be installed via cotainr (e.g. packages that require manual compilation)

Option 2: Adding more packages: un-squash myenv

It is also possible to un-squash the file:

```
unsquashfs -d ./myenv -processors 1 myenv.sqsh
```

Use the same steps as before to add more packages to the venv and squash it again.

Pros and Cons

LUMI

Pros:

- Quick (and easy) approach for installing additional packages to existing containers

Cons:

- Additional packages are installed directly on Lustre file system which can lead to bad performance and exceed your file limit (if `SquashFS` approach is not used)
- Necessary to source the `venv` / set `SINGULARITYENV_PREPEND_PATH` to get access to the packages in the virtual environment from inside the container
- Required to keep manually track of which `venv` matches which container for which use case

Summary of steps

LUMI

Open shell inside container

```
singularity shell --bind
/pfs,/scratch,/projappl,/project,/flash,/appl
container_image.sif
```

Create and activate a virtual environment:

```
python -m venv myenv --system-site-packages

source myenv/bin/activate
```

Install custom packages

```
pip install new_package
```

Make SquashFS file:

```
mksquashfs myenv myenv.sqsh -processors 1 -all-root
-action "chmod(a+rX) @true"
```

Remove venv:

```
rm -rf myenv
```

Bind and use SquashFS file:

```
export SINGULARITYENV_PREPEND_PATH=/user-
software/bin

singularity exec -B myenv.sqsh:/user-software:image-
src=/ container_image.sif python my_script.py
```

Un-squash to add packages after squashing:

```
unsquashfs -d ./myenv -processors 1 myenv.sqsh
```