

Containers in HPC – Pyxis

UManitoba Spring 2026
High-Performance Computing And Cloud
Workshop

Stefano Ansaloni

University of Manitoba

May 21, 2026



**University
of Manitoba**



**Digital Research
Alliance** of Canada

Stefano Ansaloni

Cloud Computing Specialist at University of Manitoba
(part of the Advanced Research Computing team)

Software Developer and DevOps Specialist since 2017

Linux User/Admin since 2005

What is a container

From Wikipedia ([Containerization](#)):

Containerization is operating-system–level virtualization or application-level virtualization so that softwares can run in isolated user spaces called “containers” in any cloud or non-cloud environment, regardless of type or vendor.

Properties of containers

- ▶ Each container is a fully functional and portable computing environment surrounding the application and keeping it independent of other environments running in parallel
- ▶ Each container simulates a different software application and runs isolated processes (including configurations, libraries, and dependencies)
- ▶ Multiple containers share a common operating system kernel (operative system)

Terminology

Image	Archive (or number of archives – i.e. “layers”) of a filesystem tree along with metadata
Containerfile	Recipe for building an image, including OS and software within the image (e.g. <i>Dockerfile</i>)
Container	A running instance of an image (can be a computing process, or a service daemon)
Container Runtime	Lower level component responsible for reading the image and communicating with the host kernel to start containerized processes (e.g. <i>runc</i> , <i>crun</i>)
OCI (Open Container Initiative)	Open governance structure that creates open industry standards for container formats and runtimes
Registry	An online storage area for images (e.g. <i>DockerHub</i> , <i>Quay.io</i>)

Focusing On Pyxis

What is Pyxis

From GitHub ([NVIDIA/pyxis](https://github.com/NVIDIA/pyxis)):

*Pyxis is a **SPANK** plugin for the Slurm Workload Manager. It allows unprivileged cluster users to run containerized tasks.*

*A pyxis is an **ancient small box or container**.*

Features

Pyxis is currently only available on Grex, providing the following benefits:

- ▶ Seamlessly execute Slurm jobs in unprivileged containers
- ▶ Simple command-line interface
- ▶ Support for OCI image registries
- ▶ Support for layers caching and layers sharing across nodes
- ▶ Supports multi-node MPI jobs through PMI2 or PMIx
- ▶ Allows users to install packages inside the container

Command line options

Pyxis introduces some new command line options to the Slurm default job submission tools (i.e. *salloc*, *sbatch*, *srun*):

<code>--container-image=IMAGE[:TAG]</code>	Image to use for the container filesystem
<code>--container-mounts=SRC:DST[:OPTS] [,SRC:DST]</code>	Bind mount(s) directories inside the container
<code>--container-workdir=PATH</code>	Working directory inside the container
<code>--container-remap-root</code>	Ask to be remapped to root inside the container
<code>--container-entrypoint</code>	Execute the entrypoint from the container image
<code>--container-env=NAME[,NAME]</code>	Names of environment variables to override with the host environment

The full list of options can be printed using the “`--help`” flag.

Example

Using salloc

To submit a test job using “salloc” and Pyxis, it is enough to specify the “--container-image” option:

```
salloc --partition=skylake --container-image=debian:stable-slim
```

When using Pyxis through “salloc”, Slurm will start an interactive job and return a shell from inside the container.

Example

Using sbatch

```
#!/bin/bash

#SBATCH --partition=lgpu
#SBATCH --nodes=1
#SBATCH --gpus=2
#SBATCH --ntasks=4
#SBATCH --cpus-per-task=8
#SBATCH --mem-per-cpu=8G
#SBATCH --container-image=nvcr.io/hpc/gromacs:2023.2
#SBATCH --container-mounts=/PATH/TO/INPUT:/host_pwd
#SBATCH --container-workdir=/host_pwd
#SBATCH --container-entrypoint

gmx mdrun -ntmpi 4 -ntomp 8 -nb gpu -pme gpu -npme 1 -update gpu -bonded gpu \
  -nsteps 100000 -resetstep 90000 -noconfout -dlb no -nstlist 300 -pin on \
  -v -gpu_id 01
```

Demo

Running a **GROMACS** benchmark with Pyxis

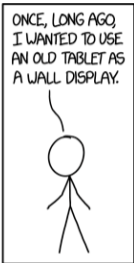
Running a GROMACS benchmark with Pyxis

Specifications

- Container Engine ⇒ Pyxis/Enroot (on Grex)
- Image ⇒ nvcr.io/hpc/gromacs:2023.2
(from the [nVidia NGC Catalog](#))
- Software ⇒ GROMACS
- Benchmark ⇒ STMV

Is Pyxis a silver bullet?

- ▶ Always check if the same software is already provided via modules-based HPC software stack
- ▶ It requires *well-built* images
 - ▶ Making or finding a suitable image is a bit of work
 - ▶ Bleeding-edge versions of programs could be poorly maintained/tested (including their images)
- ▶ Useful to encapsulate software and sometimes data to reduce number of files (e.g. python or conda based programs)



Questions?

Thank you