High Performance Computing - Advanced

June 20, 2024

WAGENINGEN
UNIVERSITY & RESEARCH

Instructors

- **Jeremie Vandenplas**
- **Leonardo Honfi Camilo**
- **Dapeng Sun**
- **Jan van Haarst**
- **Fons Prinsen**

WAGENINGEN
UNIVERSITY & RESEARCH

After this course, you should

- Be able to setup ssh keys and use them to connect to the cluster
- Be able to launch and monitor slurm jobs
- Understand different types of parallelism
- Be aware of apptainer containers and their basic usage
- Use storage resources effectively for your jobs
- Be able to interact with the community

Ice Breaker

1. Go to **wooclap.com**
2. Enter code **WURHPC2**

WAGENINGEN
UNIVERSITY & RESEARCH

Outline

12:30 - Introduction and IceBreaker

12:45 - SSH Keys

13:00 - HPC Basics Review

13:30 - Job Arrays + Exercise

14:00 - Types of Parallelism + Exercise

14:45 - Break

15:00 - Apptainer + Exercise

15:30 - Types of Storage + Exercise

16:00 - Bring Your Own Case

17:00 - End

WAGENINGEN
UNIVERSITY & RESEARCH

SSH keys

How to generate and use keys for connecting to the HPC

Advantages

- Identifies machine as a trusted device
- Saves you from typing your password (prevents bans)
- Handy for file transfers

Procedure

1. Generate ssh key on your laptop
2. Copy public key to hpc
3. Append key to ~/.ssh/authorized_keys

Generating - ssh-keygen

Open a terminal/powershell in your machine

```
$ ssh-keygen -t ed25519
```

1. Confirm key name and location (press enter)
    1. Windows: C:\Users\[USER]\.ssh
    2. Linux/Mac: ~/.ssh
2. Type passphrase (optional but **very recommended**)
3. Type passphrase again

Generating - ssh-keygen

The procedure above generates 2 files:

```
~/.ssh/id_ed25519
~/.ssh/id_ed25519.pub    # public key
```

- **id_ed25519**: Your identification, do NOT share it.
- **id_ed25519.pub**: Public key, copy to the computers you want to access.

Example

```
user001@myLaptop:~$ ssh-keygen -t ed25519
Generating public/private ed25519 key pair.
Enter file in which to save the key (/home/user001/.ssh/id_ed25519):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/user001/.ssh/id_ed25519
Your public key has been saved in /home/user001/.ssh/id_ed25519.pub
The key fingerprint is:
SHA256:pwYbiOWWkmVD+yawF1nwWYqO8A29FvxJstBYKefMYbk user001@myLaptop
The key's randomart image is:
+--[ED25519 256]--+
|                 |
|                 |
|            .    |
|         . o     |
|  .     S . .    |
| o     . +   o   |
|. o . ..+ +.o.o. |
|=++E o.... ++=B.+|
|=@Boo++.    =*=@o|
+----[SHA256]-----+
```

Upload Key to HPC

1. Copy the public key to your home folder in the HPC:

```
$ scp ~/.ssh/id_ed25519.pub [USER]@login.anunna.wur.nl:/home/WUR/[USER]/
```

2. Append (**>>**)the public key to *~/.ssh/authorized_keys* in the HPC:

```
$ cat id_ed25519.pub >> ~/.ssh/authorized_keys
```

Using the keys

- **ssh-agent** start a service that caches you ssh keys
- **ssh-add** manages and loads the key into ssh-agent

workflow:

```
$ ssh-agent #start ssh-agent process
$ ssh-add
$ Enter passphrase for /home/user/.ssh/id_ed25519:
$ Identity added: /home/user/.ssh/id_ed25519 (user@myLaptop)
```

checking:

```
$ ssh-add -L
```

Multiple keys

If you have multiple keys you can give the key a custom name and a comment.
The following command creates **mykey** and **mykey.pub** at **$HOME/.ssh**

```
$ ssh-keygen -t ed25519 -f $HOME/.ssh/mykey -C "key for this and that"
```

With a custom key name you need to specify the key you want to use:

```
$ ssh-agent #start ssh-agent process
$ ssh-add $HOME/.ssh/mykey
$ Enter passphrase for /home/user/.ssh/mykey:
$ Identity added: /home/user/.ssh/mykey (key for this and that)
```

Additional Information

Windows:

- MobaXTerm is recommended and has a GUI
- WSL is also a good option, but installation may be tricky
- PowerShell is present in WUR Clients by default but seems to lack some functionality.

Additional Resources:

wiki.anunna.wur.nl

Surf's Wiki

WAGENINGEN
UNIVERSITY & RESEARCH

Enabling a Graphical User Interface

Linux/Mac:

```
$ ssh -X login.anunna.wur.nl
```

- Mac users must have **XQuartz** installed
- MobaXTerm already does this by default

SSH Config

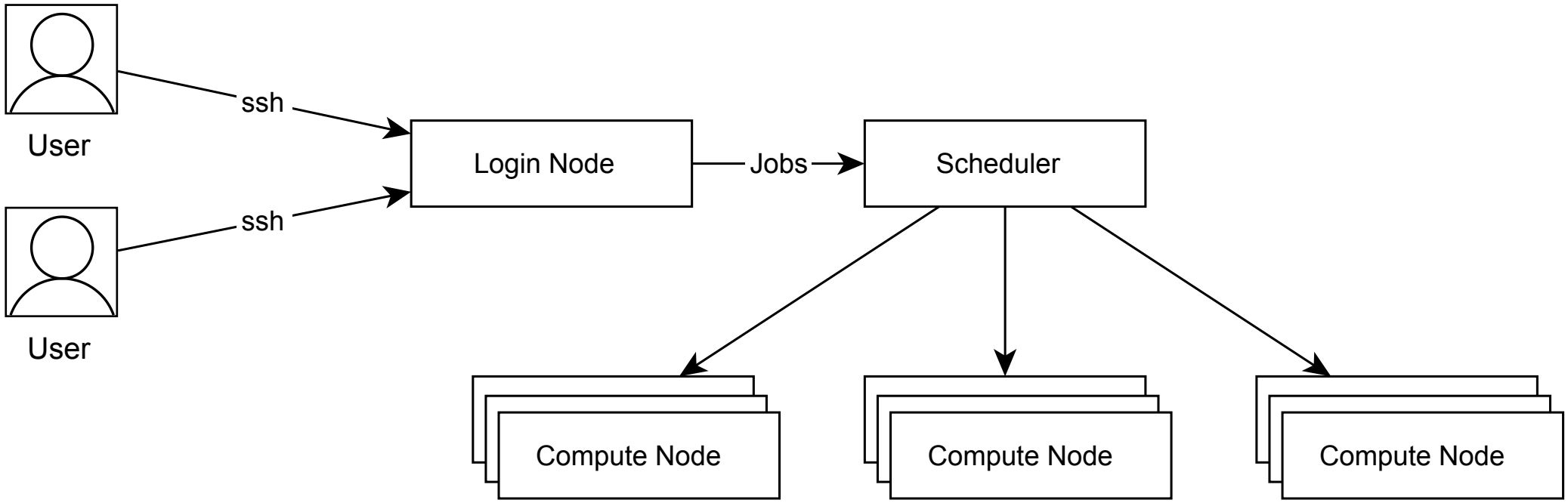
To work with SSH more easily, you can put settings in your .ssh/config file:

```
Match host *.anunna.wur.nl
    User                    user001
    Compression             no
    RequestTTY              force
    IdentityFile            ~/.ssh/mykey
    IdentitiesOnly          yes
    ForwardX11              yes
    ForwardX11Trusted       yes
```

HPC Basics Review

A Brief introduction into High performance computing concepts

HPC Diagram


User
ssh
User
ssh
Login Node
Jobs
Scheduler
Compute Node
Compute Node
Compute Node

Anunna

Operating System

Ubuntu 20.04

Scheduler:

SLURM

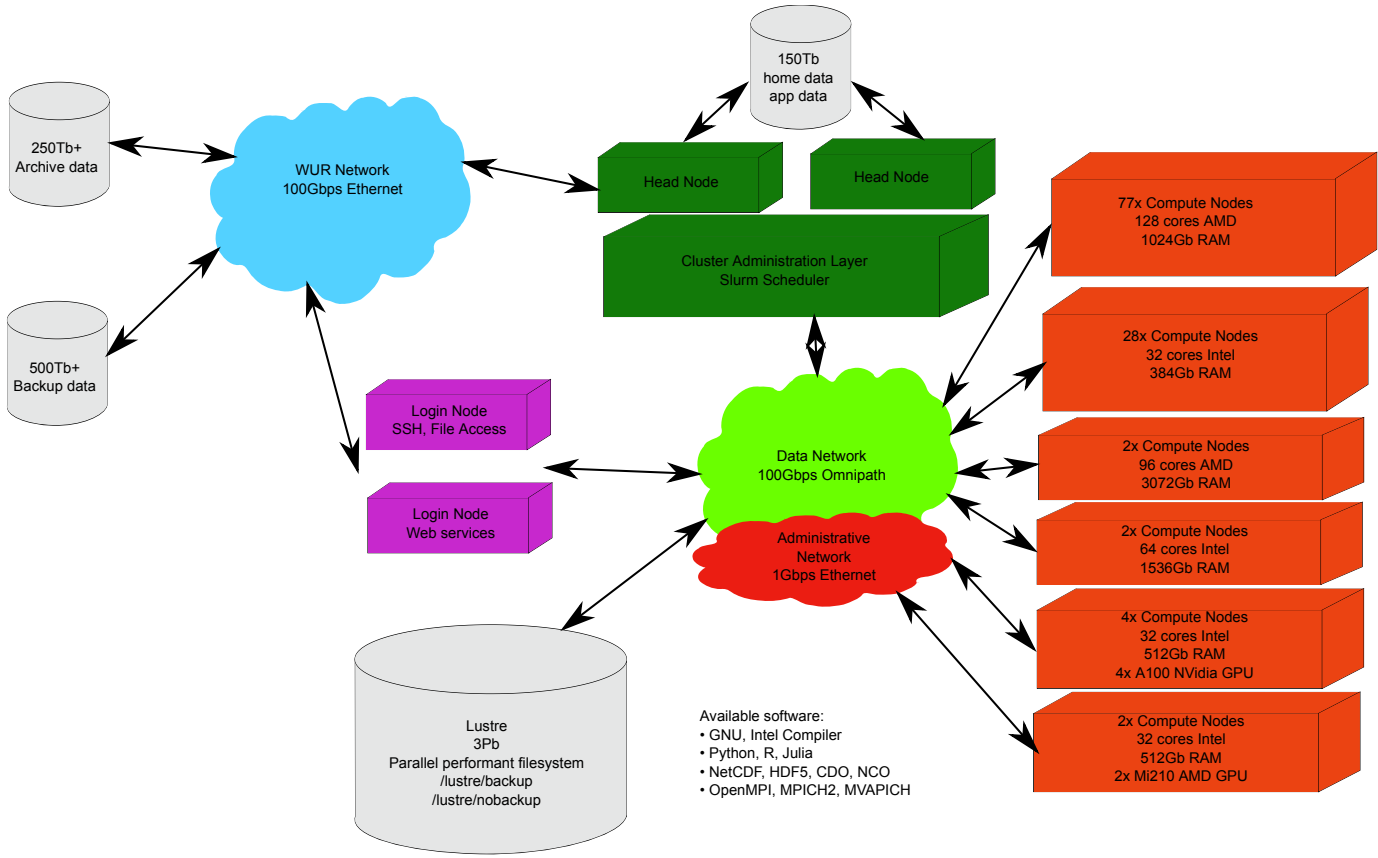
Compilters

GCC, Intel, …

Software

R, Octave, Matlab, julia, python, OpenFOAM, AlphaFold, …

GPUs:	Nodes:	Cores:
22	119	11520



WAGENINGEN
UNIVERSITY & RESEARCH

Storage

Home:

- /home/[institution]/[username]
- limited to 200 GB (210GB hard limit)
- slow filesystem (not suitable for jobs)

Archive:

- /archive/[institution]/[username]
- Cheap
- Only for storage (no jobs)

backup:

- /lustre/backup/[institution]/unit/[username]
- Extra costs for backup

nobackup:

- /lustre/nobackup/[institution]/unit/[username]

scratch:

- /lustre/scratch/[institution]/unit/[username]
- Can be cleaned up

Webservices

notebook.anunna.wur.nl

- Sandbox mode for small scripts running at the login0 node
- Larger jobs can run in the cluster
- Supports various versions of Julia, Python and R (**Ju**ll**Py**te**R**)
- Reservations can be made for courses with support from the HPC team
  - Reservations for courses can be made via a form in the itsupport page
  - New request →Hosting → High Performance Computing

galaxy.anunna.wur.nl

- Bioinformatics tools and workflows

WAGENINGEN
UNIVERSITY & RESEARCH

Best Practices and Policies

- We **never** remove user data
- No jobs should run on login node
- No jobs should output/write to the home directory
- Do not add modules to your **~/.bashrc** or **~/.bash_profile** files
- Anaconda installations at home directory are not supported

Env and Notable Variables

Display the variables in your session:

```
$ env
```

Notable:

HOME - stores the location of your home directory

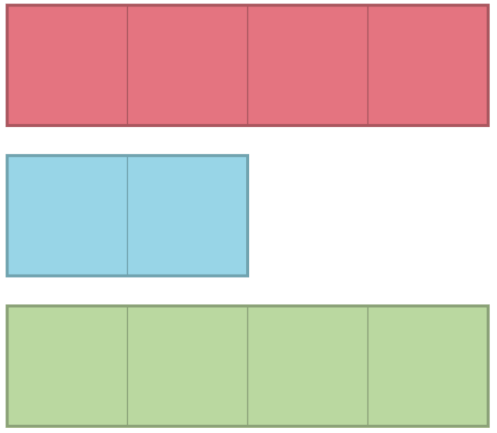
PATH - stores locations of your executable files (separated by :)

LD_LIBRARY_PATH - stores location of libraries

MODULEPATH - stores location of module files

Modules at Anunna

- Modules are to be arranged in "buckets" wrt to year
  - One version of software per bucket
  - Conveyor belt: software older than 3 years will be removed or moved to the **legacy** bucket.
  - A new bucket will be released each year.
- The building of the packages is being done with **EasyBuild**
  - Software tested across many HPCs worldwide
- If you need to a specific software
  - Submit a ticket
  - If you can provide an EasyBuild recipe, it will speed things up



EASYBUILD.io
building software with ease

WAGENINGEN
UNIVERSITY & RESEARCH

Listing Available Modules

Get overview of available modules

```
$ module overview # ml ov
```

List available software:

```
$ module avail # ml av
$ module spider # ml spider
```

Search with a clear cache (slower):

```
$ module --ignore_cache av
$ module --ignore_cache spider
```

Searching for Modules

Searching for modules

```
$ module avail R/ # ml av R/
$ module spider R/4 # ml spider R/
```

Search for a keyword inside a module's description

```
$ module key numpy # ml key numpy
```

Getting Additional Information

Get the description of a module

```
$ module whatis Python-bundle-PyPI
```

Get the entire module file

```
$ module show Python-bundle-PyPI
```

Loading Modules

load module (e.g. python)

```
$ module load 2023
$ module load Python/3.11.3-GCCcore-12.3.0
```

List loaded modules

```
$ module list
```

Removing Modules

Remove individual module (e.g. python)

```
$ module unload Python/3.11.3
```

Remove all modules

```
$ module purge
```

Swap a module with another

```
$ ml module swap Python-bundle-PyPI SciPy-bundle
```

Jargon

Job:
All the processes, or tasks, involved in achieving a computational goal

Tasks:
A process, or a unit of work, that comprises a job.

Threads:
Compute operations that make up a process.

CPU:
Core or hardware thread

Muti-threaded:
Tasks are broken down onto several CPUs

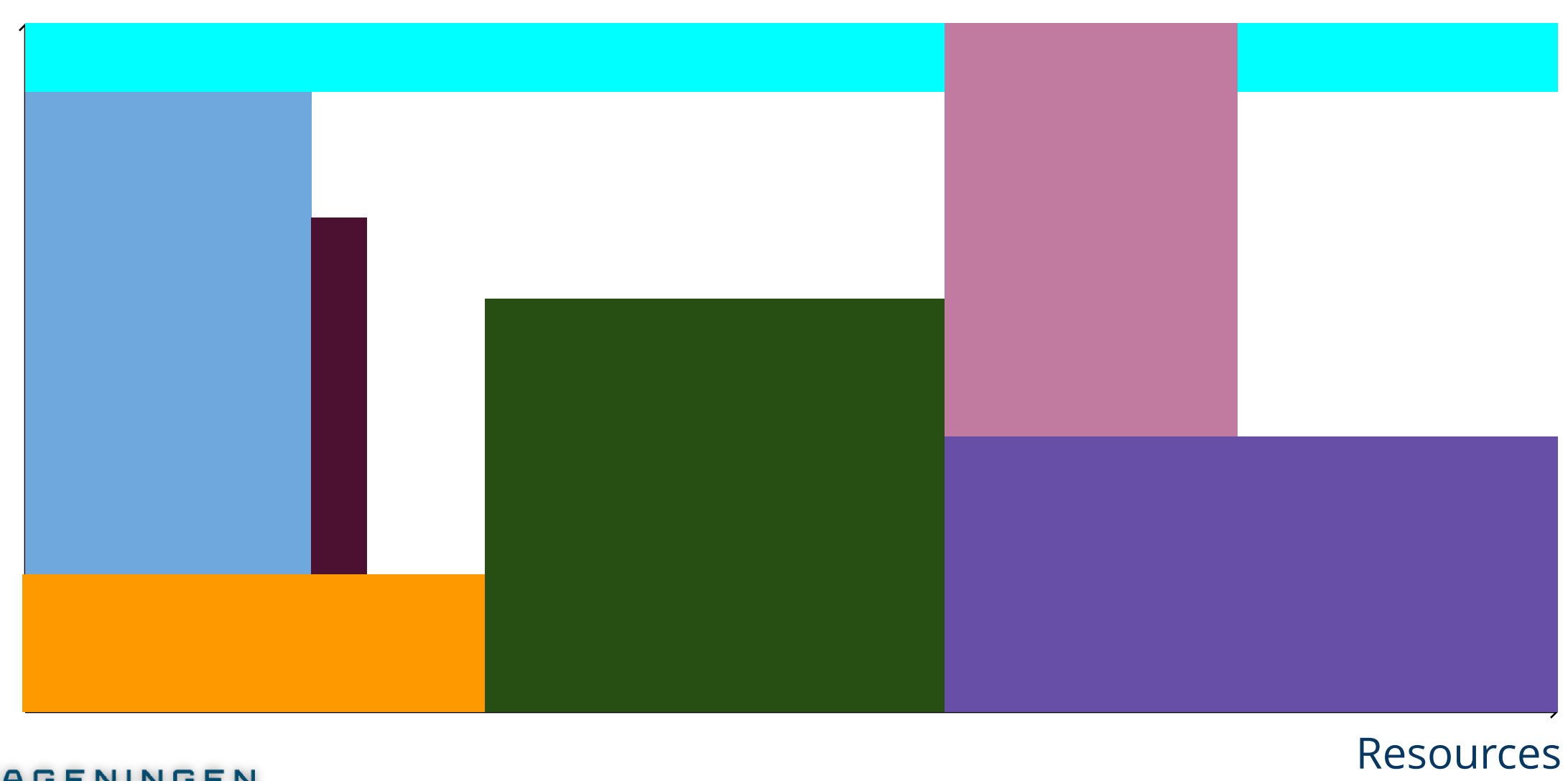
Multi-process:
Job composed of several tasks

SLURM

Simple Linux Utility for Resource Management

- Manages and **allocates resources** (compute nodes)
- Manages and **schedules jobs** on a set of allocated nodes
- **Sets up environments** for parallel and distributed computing

Tetris



WAGENINGEN
UNIVERSITY & RESEARCH

Interactive jobs

sinteractive template

```
$ sinteractive -c <num_cpus> --mem <amount_mem> --time <minutes> -p <partition>
```

Example

```
# sinteractive -c <cpus> --mem <MB>
$ sinteractive -c 1 --mem 2000
srun: job 19271078 queued and waiting for resources
srun: job 19271078 has been allocated resources
```

Equivalent **srun** command

```
$ srun -N 1 -n 1 --mem 2000 --pty bash -i
```

Note: sinteractive will fail if job is not allocated in 60 seconds

WAGENINGEN
UNIVERSITY & RESEARCH

SLURM Scripts

```
#!/bin/bash
##---------- Name of the job ----------
#SBATCH --job-name=example01
##---------- Mail address ----------
#SBATCH --mail-user=my.email@wur.nl
#SBATCH --mail-type=ALL
##---------- Output files ----------
#SBATCH --output=output_%j.txt
#SBATCH --error=error_output_%j.txt
##---------- Other information ----------
#SBATCH --comment='My project number'
##---------- required resources ----------
#SBATCH --time=0-1
#SBATCH --ntasks=1
#SBATCH --mem=4096
#SBATCH --cpus-per-task=1

##---------- Environment, Operations and job step ----------

module load Python/3.11.5

echo "Hello World!"

```

Submitting SLURM scripts

sbatch

```
$ sbatch script_slurm.sh
Submitted batch job 19271078
```

Options specified in the command overrule the script

```
$ sbatch script_slurm.sh -N 2 -n 10 \
    --mem 32G --time=0-3:0:0 --cpu-per-task=2
```

WAGENINGEN
UNIVERSITY & RESEARCH

Extra options

Main partition:

```
--partition=main
```

GPU partition:

```
--partition=gpu
```

Specific feature:

```
--constraint=gen3
```

Enable GPUs

```
--gres=gpu:1
```

CPUs per GPU:

```
--cpus-per-gpu=gpu:4
```

Monitoring Jobs -squeue

squeue

View **information** about jobs located in the SLURM **scheduling queue**

To report a list of users:

```
$squeue -u <user_id>
```

To report a list of specific jobs:

```
$squeue -j <job_id_list>
```

To report the expected start time of pending jobs:

```
$squeue --start
```

Monitoring Jobs - sacct

sacct

- Displays accounting data for all jobs/steps
- Some information are available only at the end of the job

To get an overview of all jobs run in 2024

```
$ sacct -X -u <name> --starttime 2024-01-01 --endtime now
```

To get an overview of a job

```
$ sacct -j <job_id> --format=JobID%-20,AveRSS, MaxRSS,Elapsed
```

Monitoring Jobs - scontrol

scontrol

- Update job resource requests
- Work only for running jobs
- **Provides a lot of information…**

To get information of a job

```
$ scontrol show job <job_id>
```

To get information on nodes

```
$ scontrol show nodes
```

WAGENINGEN
UNIVERSITY & RESEARCH

Cancelling Jobs - scancel

Cancel a single job:

```
$scancel <job_id>
```

Cancel multiple jobs:

```
$scancel <job_id0> <job_id1> <job_id2> ...
```

Job Arrays

How to submit array of jobs in slurm

Embarrassingly Parrallel

Embarrassing Parallelism is obtained by launching the **same program** multiple times simultaneously

- `Every program does the same thing`
- `No inter-process communication`
- `Useful cases`
- `Multiple input/data files`
- `Random sampling`

Job Arrays - resource

To submit a job array with index values between 1 and 3

--array=1-3

To submit a job array with index values of 1, 5, 10, 11, 12

--array=1,5,10-12

To submit a job array with index values between 2 and 60, with a limited number (4) of simultaneous running jobs

--array=2-60%4

Job Arrays - environment variable

Job arrays have additional environment variables

The job array index value

$SLURM_ARRAY_TASK_ID

Exercise

Submit an array of jobs to run the same program simultaneously using different inputs

Task:

Write a SLURM script for a **job array** that loads and generate figures for **multiple Dutch weather stations**, with IDs 240, 260 and 270

- **Script:** /lustre/shared/training_slurm/spring_2024/serial/training/weer_vanaf_2000.py
- **SLURM option**: --array=240,260,270
- **SLURM variable**: $SLURM_ARRAY_TASK_ID
- **Command:** python weer_vanaf_2000.py $SLURM_ARRAY_TASK_ID

Parallelism

An overview of different job types

Job Types

- Serial
- Multi-threaded
- Multi-process
- GPU

Serial

- Simplest type of job
- Runs on a single process (and single thread)
- Job runs through program sequentially

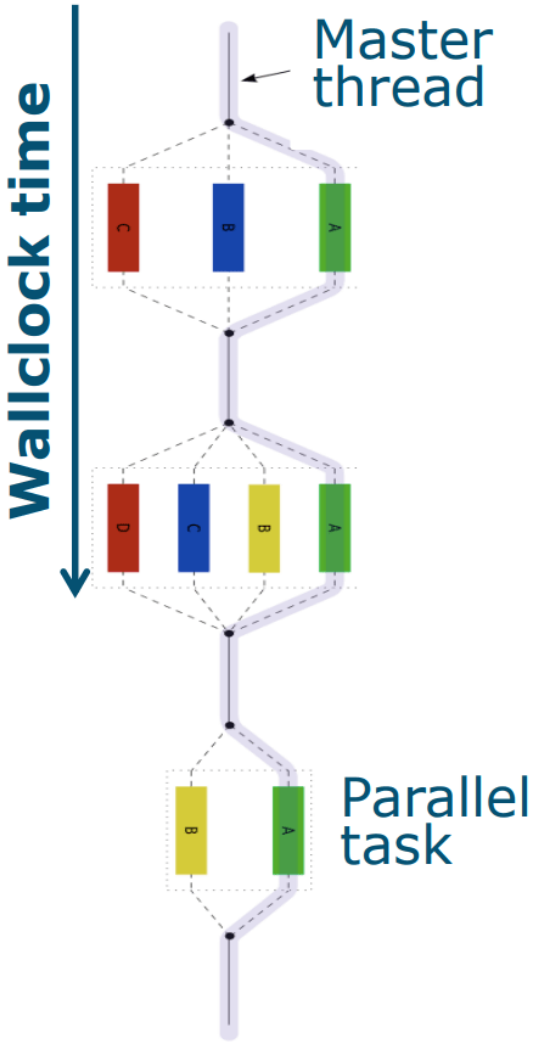
Example

```
##---------- required resources ----------
#SBATCH --time=0-0:10
#SBATCH --ntasks=1
#SBATCH --mem=4096
#SBATCH --cpus-per-task=1
##---------- Environment, Operations and job step ----------

module load myModule

myprog  # srun myprg
```

Multi-threaded



- Uses OpenMP or TBB (usually)
- Programs spawns itself on the node
- Runs on a single node
- Memory is shared between the threads

example

```
##---------- required resources ----------
#SBATCH --time=0-0:10
#SBATCH --ntasks=1
#SBATCH --mem-per-cpu=4000
#SBATCH --cpus-per-task=3
##---------- Environment, Operations and job step ----------

export OMP_NUM_THREADS=3

module load myModule

myprog # srun myprog
```

Note: Total memory used is 3 x 4 GB = 12 GB

WAGENINGEN
UNIVERSITY & RESEARCH

Multi-Process

- Enabled by Message Passing
  - MPI
- One program (rank 0) spawns several copies across the computational nodes
  - These processes are labelled and organized by ranks
- Processes communicate with each other across the network ("passing messages")

WAGENINGEN
UNIVERSITY & RESEARCH

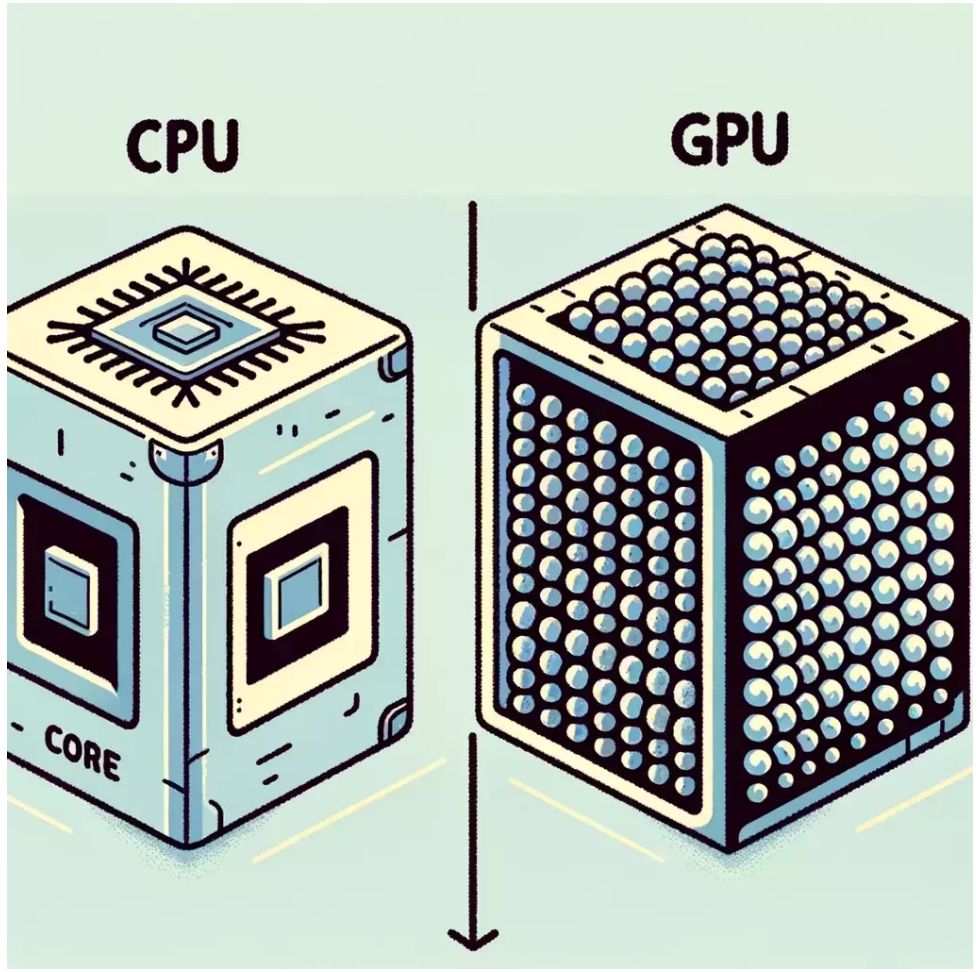
Example

```
##---------- required resources ----------
#SBATCH --time=0-0:10
#SBATCH --ntasks=4
#SBATCH --mem=4096
#SBATCH --cpus-per-task=1
##---------- Environment, Operations and job step ----------

module load myModule

mpirun myprog
```

GPU



- Embarrassingly parallel on steroids
- Hundreds of thousands, possibly millions of threads
- Relatively simple operations

Example

```
##---------- required resources ----------
#SBATCH --time=0-0:10
#SBATCH --ntasks=1
#SBATCH --mem=4096
##SBATCH --cpus-per-task=1
#SBATCH --cpus-per-gpu=4
#SBATCH --partition=gpu
#SBATCH --gres=gpu:1
##---------- Environment, Operations and job step ----------

module load myModule

myprog # srun myprog
```

Exercise - Compute π

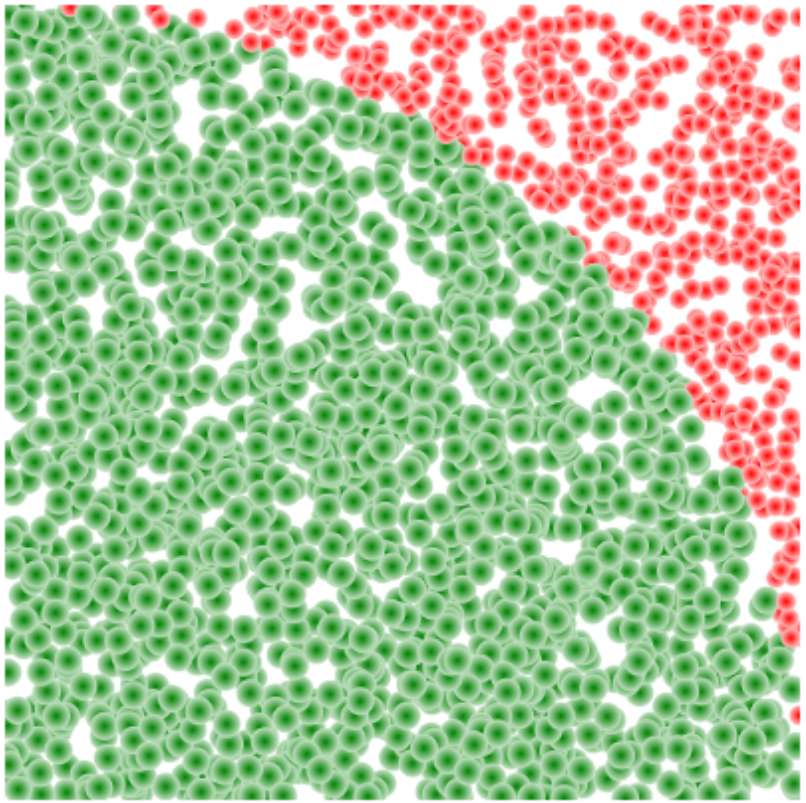
In this exercise, you will compute π with different job types and assess its performance

Exercise - Compute π - Method

The method involves following algorithm

1. Generate a pair of random numbers between 0 and 1 and assign to x and y
2. Each number generated increases the total number of samples by 1
3. Each number generated that lies within the unit circle is registered in the variable N_in

$$\pi \approx 4 \frac{N_{in}}{N_{total}}$$



Exercise - Compute π

C-code binaries have been compiled for each computation:

- pi-serial
- pi-threads
- pi-mpi
- pi-cuda

The binaries are located at:

```
/lustre/shared/hpcCourses/advanced/pi
```

A module file has been created at

```
/lustre/shared/hpcCourses/advanced/modules
```

Exercise - Compute π - Instructions

1. Add the pi module to your MODULEPATH
    1. Hint: module use
2. Search for the pi module with spider, and load it
3. Create a slurm script for each job type
    1. serial
    2. multithread
    3. multiprocess
    4. gpu
4. Experiment with the number of threads and tasks, see if there are any time improvements

Exercise - Compute π - Solutions

Please run the exercises on computing node

```
$ module use /lustre/shared/hpcCourses/advanced/modules/
$ # To access pi-serial.x pi-threads.x pi-mpi.x pi-cuda.x
$ module load pi
```

Hint for MPI run:

```
$ mpirun -n 8 pi-mpi
```

Hint for GPU node:

```
$ sinteractive --partition=gpu --gres=gpu:1
$ pi-cuda
```

WAGENINGEN
UNIVERSITY & RESEARCH

Break (till 15:00)

Apptainer Containers

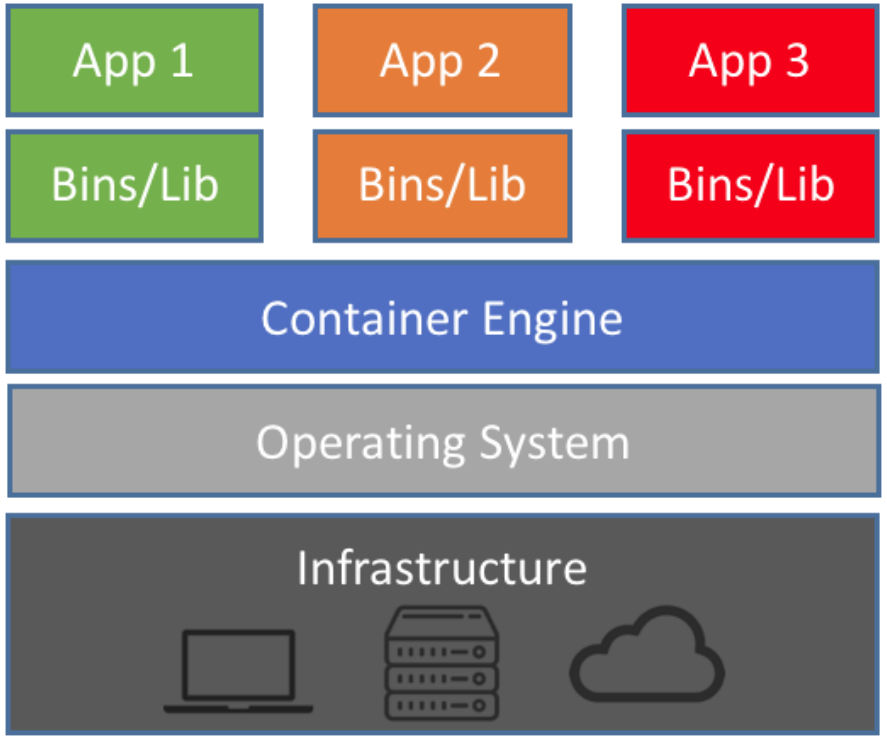
How to launch jobs with apptainer

WAGENINGEN
UNIVERSITY & RESEARCH

Containers

A **container** is an entity providing an isolated software environment (or filesystem) for an application and its dependencies.

Containers share the resources and kernel of the host machine.



Containers

Containers Examples

Docker:

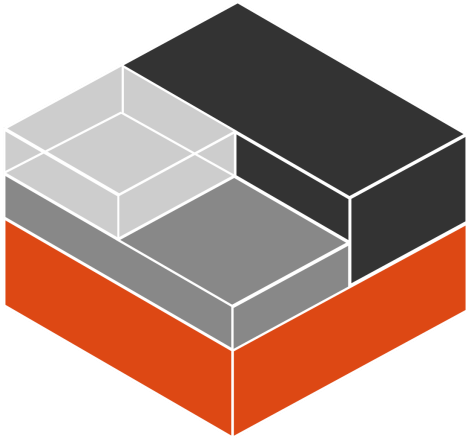
The first engine to gain popularity, still widely used in the IT industry. Not very suitable for HPC as it requires *root* privileges.

Apptainer:

Under free and open source software (FOSS) guarantee.

A simple, powerful *root*-less container engine for the HPC world.

Container Technologies



Workflow

- Create or download image
  - dockerhub
  - shub
  - definition files (.def)
- Execute image
  - **exec** - executes a command in the image environment
  - **shell -** opens interactive shell
  - **run** - executes a script built into an image, usually an application
- Publish/Distribute
  - Definition files and/or images will run on other machines

Example - Checking OS Version - 1/2

Use apptainer to download the another version of ubuntu

1. Open a **sinteractive** session with 4 GB of memory
2. Load **Apptainer** under the 2023 modules
3. Go to your scratch folder
4. Pull an **ubuntu 24.04** image from dockerhub and save it under **ubuntu-2404.sif**
5. Check the version of ubuntu running on the node
6. Run a shell session on the **ubuntu-2404.sif** image.
7. Check the version of ubuntu running on the container.

Example - Checking OS Version - 2/2

```
$ sinteractive --mem 4G
$ module load 2023
$ module load Apptainer/1.3.0
$ cat /etc/lsb-release
$ cd /lustre/scratch/WUR/user001
$ apptainer pull ubuntu-2404.sif docker://ubuntu:24.04
$ apptainer shell ubuntu-2404
Apptainer> cat /etc/lsb_release
Apptainer> exit
```

Exercise 1 - Hello World

Please **do not** run apptainer directly on login nodes

```
$ sinteractive --mem 1G
$ module load 2023
$ module load Apptainer/1.3.0
$ apptainer pull ./hello-world.sif shub://vsoch/hello-world
$ apptainer run hello-world.sif
```

Exercise 2 - lolcow

```
#!/bin/bash
#SBATCH --job-name=Apptainer_Docker_Pull
#SBATCH --ntasks=1
#SBATCH --nodes=1
#SBATCH --mem=2gb
#SBATCH --output=/home/WUR/user001/logs/apptainer.log
#SBATCH --error=/home/WUR/user001/logs/apptainer.error
#SBATCH --time=00:10:00
#SBATCH --partition=main
#SBATCH --comment="apptainer lolcow"

module load 2023
module load Apptainer/1.3.0

workDir=/lustre/scratch/WUR/user001/apptainer

apptainer pull $workDir/lolcow.sif docker://godlovedc/lolcow
apptainer run $workDir/lolcow.sif
```

Storage

Overview of different storage types available at the HPC

On the Node Storage

"I/O Performance"

RAM disk - /dev/shm

- Memory of the node
- Very fast, suitable for read/write intensive jobs
- Volatile, gets erased when job ends (or even before...)
  - **Hint:** Make a folder **/dev/shm/$USER** and set permissions to 1755 (or 1777)
  - Data needs to be copied to a permanent location

Local SSD - /tmp

- SSD on the node
- Volatile, gets erased when the node reboots
  - Data needs to be copied to a permanent location

Network Storage

Lustre - distributed filesystem

- scratch, backup, nobackup
- Built to be performant and to handle computations
- Low latency connection (OmniPath)

NFS

- /home & /shared
- Ethernet connection

Isilon

- /Archive
- Outside cluster

"I/O Performance"

Exercise

1. Open interactive session to a gen3 node (with 8G RAM)
2. Create test files, use /shared/apps/testdata/create_testdata.sh. Have a look at the file first !
3.
    - /dev/shm
    - /tmp
    - /lustre/nobackup/[your location]
    - [your home]

Storage Types - Solution

1. Open interactive session to a gen3 node (with 8G RAM)

```
sinteractive --mem 8G --constraint=gen3
```

2. Create test files, using the different storage options

```
for storage in /dev/shm /tmp /lustre/nobackup/[GROUP]/$USER $HOME
do
 echo $storage
 cd $storage
 time /shared/apps/testdata/create_testdata.sh
done
```

Bring Your Own Case

Discuss your use case and/or problem with the group

Closing Remarks

Support, wiki, community and whatever else

Communication and Support

Message of the day

- Announcements often posted

Annuna Users at Teams

- Feel free to post there if you have any questions or problems

Tickets

- For more involved support and questions, create a ticket.
  - https://itsupport.wur.nl/
  - New Requests
  - Create a ticket
  - Type HPC at the field "*What is your question about?*"

Wiki - wiki.anunna.wur.nl

- Slides from courses
- Instructions on how to run jobs
- Tips about python and R
- ...

Feedback Form

WAGENINGEN
UNIVERSITY & RESEARCH

So long, and thanks for all the fish!

WAGENINGEN
UNIVERSITY & RESEARCH