

# Getting Started with MeluXina

<https://luxprovide.github.io/SCynergy2026-GettingStartedWithMeluXina/>

Xavier Bessonon  
HPC Software Engineer



**EPICURE**  
Unlocking European-level HPC Support



## High Performance Computing

- What is HPC?

## HPC Platforms

- Hardware view
- Software view

## Using an HPC platform

- User Interfaces
- Interactive vs batch job
- Job scheduler
- Software environment

## MeluXina in practice

- User and project accounts
- Data storage
- Authentication
- Shell & web-portal access

## Hands-on!

- Configuration
- Urban wind simulation
- Deep learning with PyTorch



**EPICURE**  
Unlocking European-level HPC Support



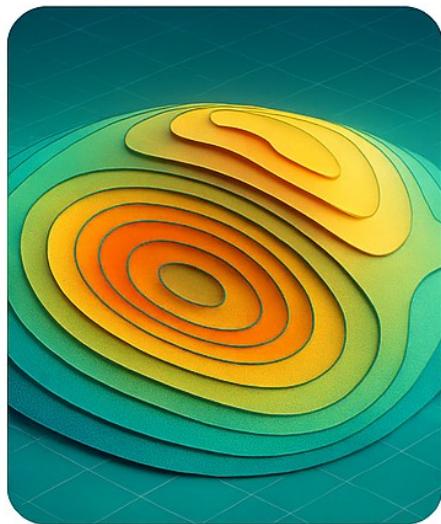
# What is High-Performance Computing?

SCynergy 2026

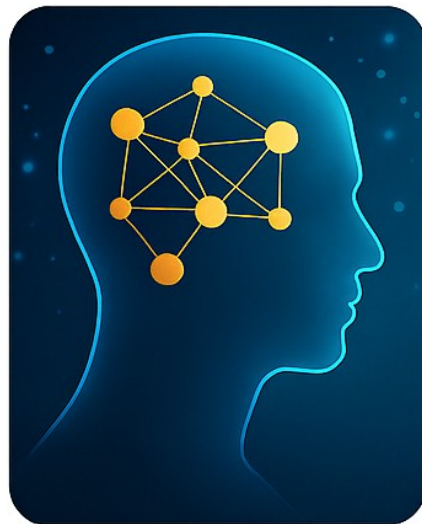


**EPICURE**  
Unlocking European-level HPC Support

# HPC is everywhere



**Simulations & modeling**



**AI / ML training & inference**



**Large-scale data analytics**



**Engineering, science & industrial R & D**

# What is High Performance Computing?

## High Performance Computing (HPC)

- Use of **parallel and distributed computers with fast interconnects**
- To execute an application quickly and efficiently

## Why parallel computers?

- Performance of single CPU core is getting limited (power, physics)
- Multiple cores are used to increase the computing capacity

## HPC is challenging

- Active research domain
- Provides tools for industry and research

# How to get faster with HPC?

## Build faster processors

- Moore's law continues but ***The free lunch is over!***
- CPU serial-processing speed is reaching its physical limit
- **Multi-cores processor architectures**
- **Accelerators and specialized processors** (GPU, TPU, FPGA, etc.)

## Combine multiple computers

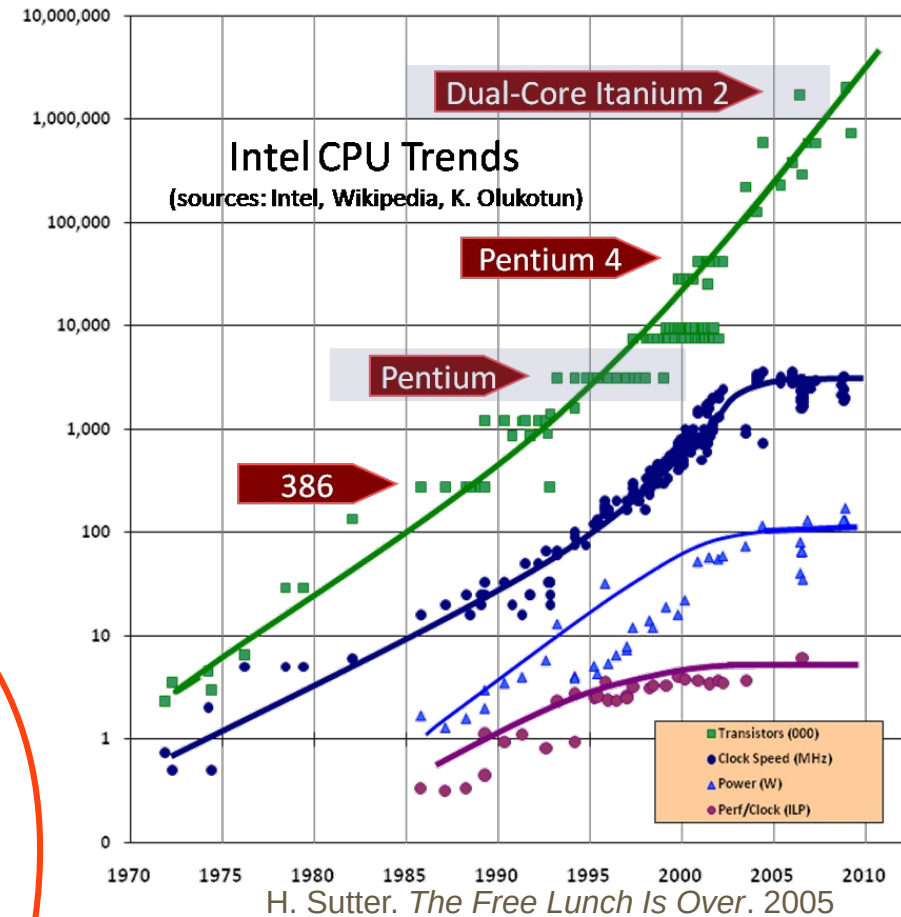
- **HPC Clusters and Supercomputers**

## Better use of the hardware

- **Identify the actual bottleneck** (CPU, memory, network, etc.)
- **Vectorization (SIMD)**

Not to forget: **Better algorithms**

**Parallel programming**



# How much faster is HPC?



Your laptop



CPU	8 cores	89,984 cores	1,051,392 cores
Memory	32 <b>GB</b>	488 <b>TB</b>	5.44 <b>PB</b>
Storage	1 <b>TB</b>	12.6 <b>PB</b>	TBA
Network	Ethernet 10 Gb/s	Infiniband 200 Gb/s	Slingshot 200 Gb/s
Accelerators	1 GPU	800 GPUs	43,808 GPUs
Theoretical Performance	350 <b>Gflops</b>	18 <b>Pflops</b>	2.82 <b>Eflops</b>



# HPC Platforms

Hardware and Software Views

SCynergy 2026



**EPICURE**  
Unlocking European-level HPC Support

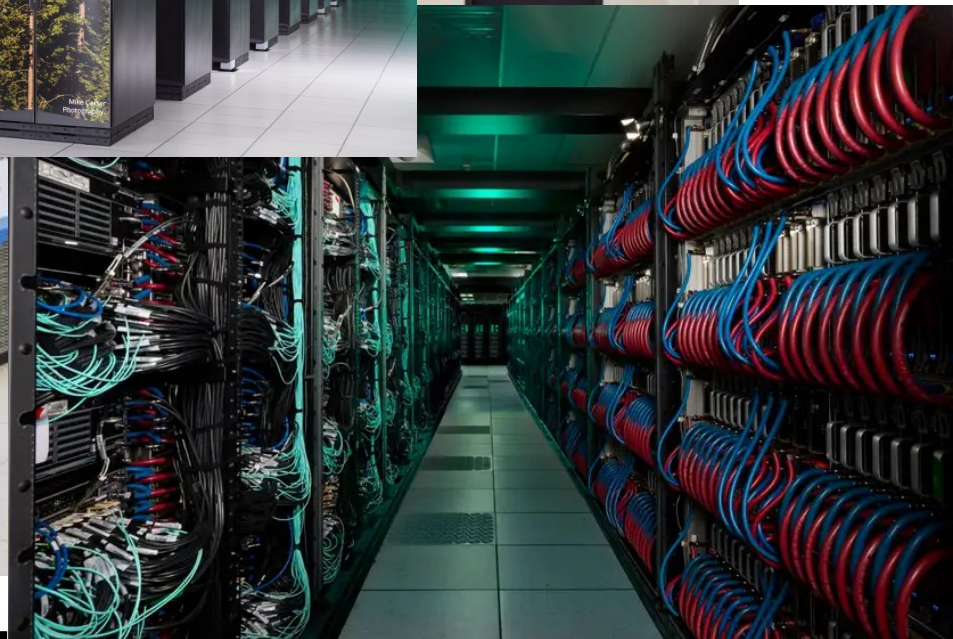
# What does an HPC platform look like?



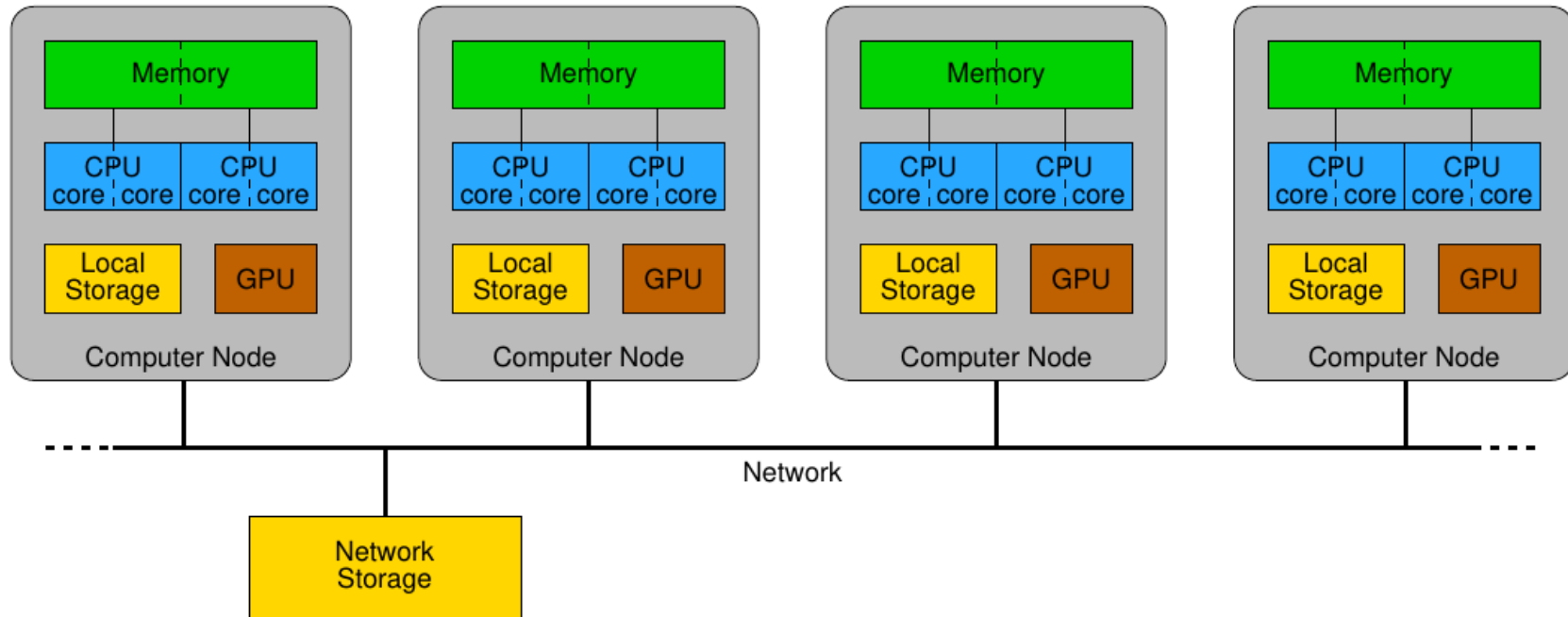
Photos: Garry McLeod/LLNL



**El Capitan, Lawrence Livermore  
National Laboratory, USA  
MeluXina, LuxProvide,**



# HPC platform: Hardware View



Different components or type of **resources**:

- Processors, GPU/Accelerators, Memory, Storage, Network, ...

# HPC platform: Software View

## Operating System

- Manage hardware resources
- Interface between userspace and hardware

## HPC Tools

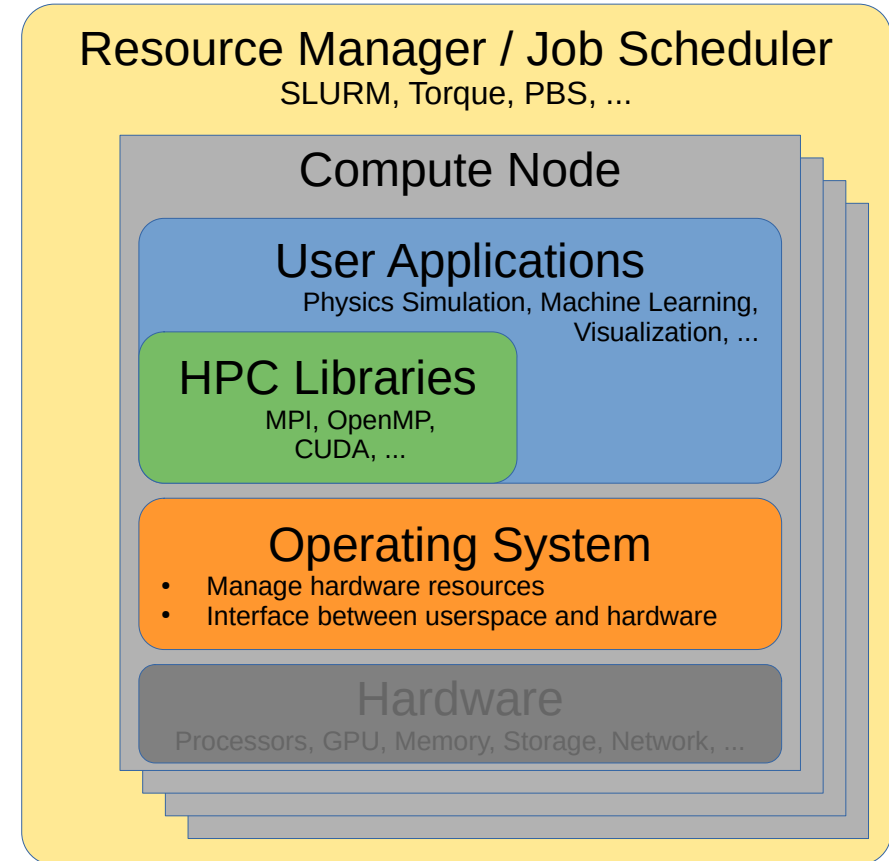
- Libraries, compilers and tools for developing and running parallel programs
- MPI, OpenMP, CUDA, TensorFlow, ...

## User Applications

- Scientific, engineering, and commercial
- Simulation, Machine Learning, Visualization, ...

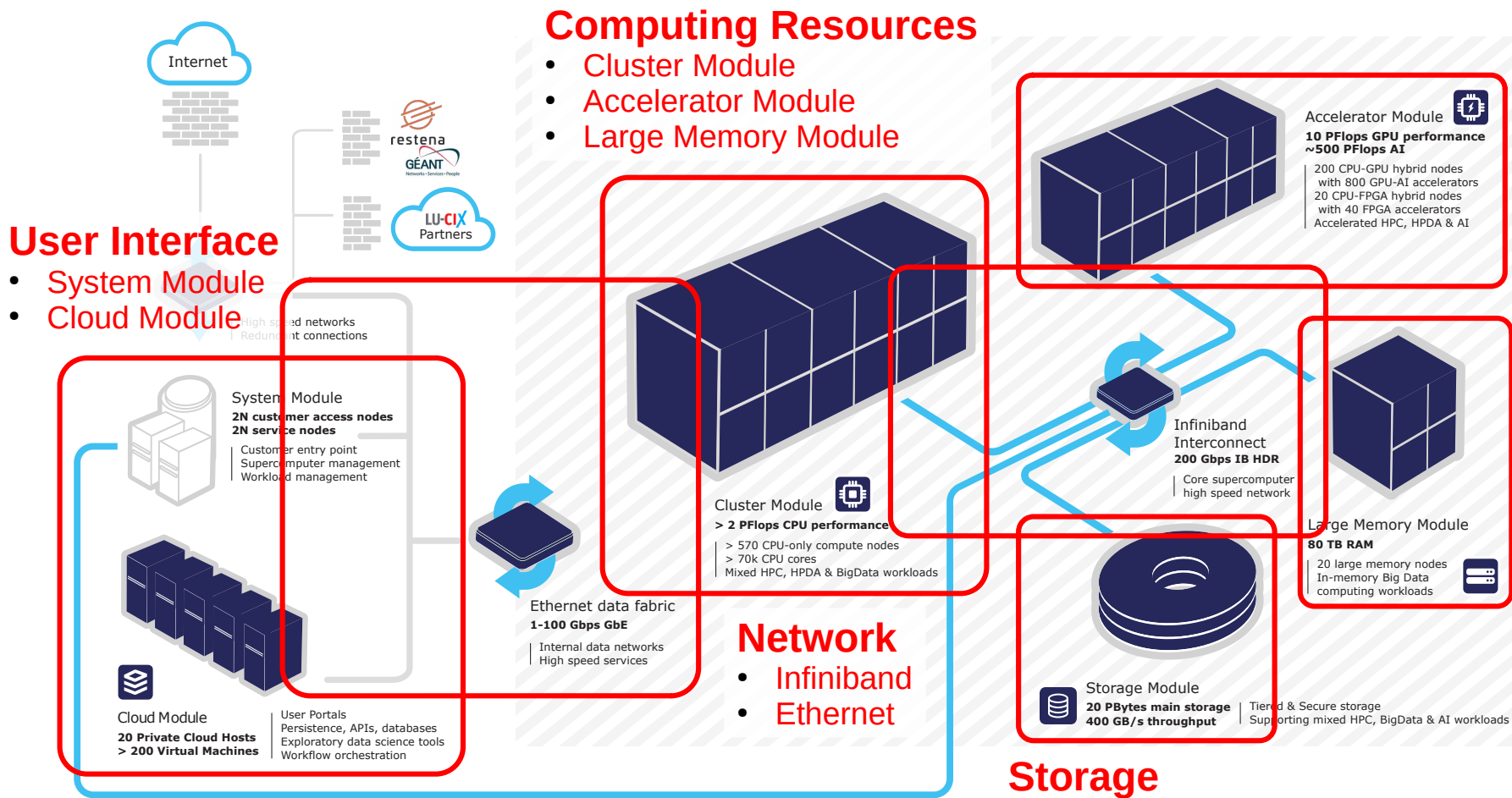
## Resource Manager / Job Scheduler

- Allocate resources and schedule user jobs
- Ensure efficient and fair resource sharing



# Example: MeluXina Supercomputer

Source: <https://docs.lxp.lu/system/overview/>





# Using an HPC Platform

HPC from the user point of view

SCynergy 2026



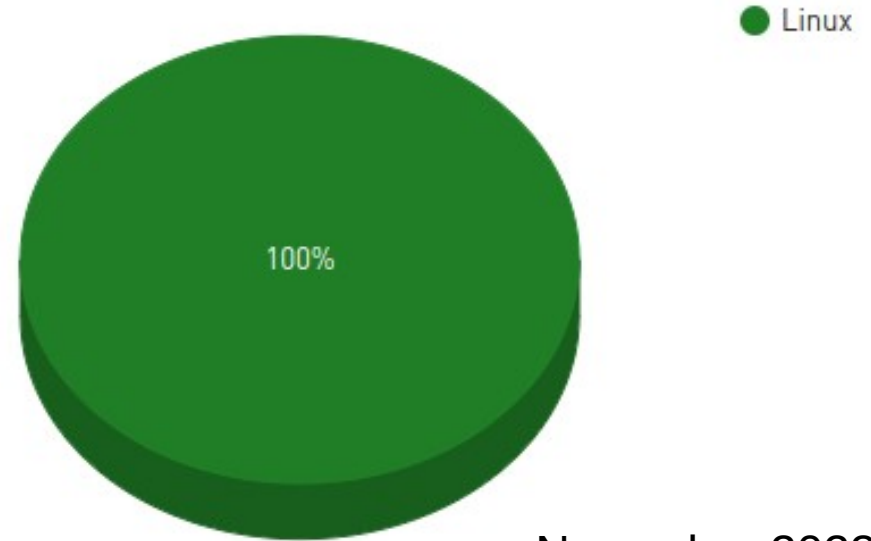
**EPICURE**  
Unlocking European-level HPC Support

# Quiz!

**What is the Operating System used on the Top500 supercomputers?**



Operating system Family System Share



November 2023

Source: <https://www.top500.org/statistics/list/>

# User Interfaces for HPC

## Command Line Interface (CLI)

- By far the most common

### Advantages

- Simple (always works)
- Easy to program/automatize
- Fast and low bandwidth requirements

### Disadvantages

- Not user friendly
- Requires knowledge and memorizing

## Alternatives to CLI

### Web portal

- Dedicated to some specific tasks

### Remote connection

- Client GUI and remote server for dedicated applications

### API for remote execution

### GUI for dedicated tasks

- For example, file transfer

# Command Line Interface

```
xbesson@access2:~/202401-Large_Midrex_Blast_Furnace_Testing/run_te...0_2024-02-02-17-39-11_LargeMidrexOF_OF128-simple-dt0.00001 — Konsole
DILUPBiGStab: Solving for CH4, Initial residual = 4.1610352e-07, Final residual = 4.1610352e-07, No Iterations 0
DILUPBiGStab: Solving for CO, Initial residual = 4.1610352e-07, Final residual = 4.1610352e-07, No Iterations 0
DILUPBiGStab: Solving for CO2, Initial residual = 4.1610352e-07, Final residual = 4.1610352e-07, No Iterations 0
DILUPBiGStab: Solving for H2, Initial residual = 4.1610352e-07, Final residual = 4.1610352e-07, No Iterations 0
DILUPBiGStab: Solving for H2O, Initial residual = 4.1610352e-07, Final residual = 4.1610352e-07, No Iterations 0
DILUPBiGStab: Solving for O2, Initial residual = 4.1610352e-07, Final residual = 4.1610352e-07, No Iterations 0
DILUPBiGStab: Solving for h, Initial residual = 0.00019233561, Final residual = 3.0765688e-09, No Iterations 1
min/max(T) = 1161.0753, 1175.5424
DICPCG: Solving for p, Initial residual = 8.4633275e-05, Final residual = 7.2521317e-07, No Iterations 1
diagonal: Solving for rho, Initial residual = 0, Final residual = 0, No Iterations 0
time step continuity errors: sum local = 1.5739e-11, global = 2.2920826e-13, cumulative = 3.8700406e-09
DICPCG: Solving for p, Initial residual = 7.2976685e-07, Final residual = 7.2976685e-07, No Iterations 0
diagonal: Solving for rho, Initial residual = 0, Final residual = 0, No Iterations 0
time step continuity errors: sum local = 1.5837742e-11, global = 2.1096539e-13, cumulative = 3.8702516e-09
DICPCG: Solving for p, Initial residual = 7.2983852e-07, Final residual = 7.2983852e-07, No Iterations 0
diagonal: Solving for rho, Initial residual = 0, Final residual = 0, No Iterations 0
time step continuity errors: sum local = 1.5839298e-11, global = 2.2920826e-13, cumulative = 3.8704808e-09
ExecutionTime = 113.93 s ClockT
```

```
xbesson@access2:~/202401-Large_Midrex_Blast_Furnace_Testing/run_te...0_2024-02-02-17-39-11_LargeMidrexOF_OF128-simple-dt0.00001 — Konsole
Courant Number mean: 5.3197731e-
Time = 0.00054
diagonal: Solving for rho, Init
PIMPLE: Iteration 1
[Progress bars for rho, p, and h variables]
Mem[
29.96/2510] Tasks: 102, 1307 thr, 1546 kthr, 128 running
0K/4.00M Load average: 69.75 18.72 6.45
Uptime: 3 days, 23:05:37

PID USER      PRI NI  VIRT  RES  SHR  CPU%MEM%  TIME Command
680447 xbesson   20  0  472M 189M 62476 R 101.1 0.1 0:44.69 |-| slurmstepd: [2146330.0]
680455 xbesson   20  0  476M 193M 62220 R 101.1 0.1 0:44.60 |-| slurmstepd: [2146330.0]
680456 xbesson   20  0  478M 194M 62312 R 101.1 0.1 0:44.67 |-| slurmstepd: [2146330.0]
680457 xbesson   20  0  477M 193M 62424 R 101.1 0.1 0:44.66 |-| slurmstepd: [2146330.0]
680461 xbesson   20  0  483M 202M 62452 R 101.1 0.1 0:44.36 |-| slurmstepd: [2146330.0]
680463 xbesson   20  0  480M 199M 62388 R 101.1 0.1 0:44.49 |-| slurmstepd: [2146330.0]
680471 xbesson   20  0  485M 202M 62320 R 101.1 0.1 0:44.40 |-| slurmstepd: [2146330.0]
680473 xbesson   20  0  481M 202M 62288 R 101.1 0.1 0:44.61 |-| slurmstepd: [2146330.0]
680482 xbesson   20  0  482M 203M 62324 R 101.1 0.1 0:44.33 |-| slurmstepd: [2146330.0]
F1[Help] F2[Setup] F3[Search] F4[Filter] F5[List] F6[Sort] F7[View] F8[Close] F9[Quit] F10[Quit]
```

```
u100054@login02:~ — Konsole
Meluxina

You are on a MeluXina login node

-----
System information: Compute
-----
Nodes | CPU | RAM | Accelerator | Disk
-----|-----|-----|-----|-----
573N | 2x AMD 7H12: 128c @2.6G | 512GB | - | -
200N | 2x AMD 7452: 64c @2.3G | 512GB | 4x NVIDIA A100-40 | 1.92G
20N | 2x AMD 7452: 64c @2.3G | 512GB | 2x Intel Stratix 10MX-16 | 1.92G
20N | 2x AMD 7H12: 128c @2.6G | 4096GB | - | 1.92G

-----
System information: Data
-----
Tler | Capacity | Speed | Type | Location on compute/login
-----|-----|-----|-----|-----
Scratch | 0.6PB | 400GB/s | NVMe | /project/scratch
Home/Project | 12.5PB | 180GB/s | HDD | /home/users, /project/home
Backup | 7.5PB | 30GB/s | HDD | -

-----
System information: Interconnect
-----
Fabric: Infiniband HDR, 200Gbps, DragonFly+ topology
Links : 1x on CPU nodes, 2x on GPU, FPGA & LargeMemory nodes

-----
System information: Software
-----
Production software stack: 2021.3, current default and obsolete on
14 Jan 2023
Production software stack: 2022.1, default from 15 Jan 2023, use it
now with: 'module load env/release/2022.1'
Modules system: LMod, use 'module av' on nodes to discover the environment

-----
Center information
-----
News & Events : luxprovide.lu
Documentation : docs.lxp.lu
System status : weather.lxp.lu
Support : servicedesk.lxp.lu, servicedesk@lxp.lu

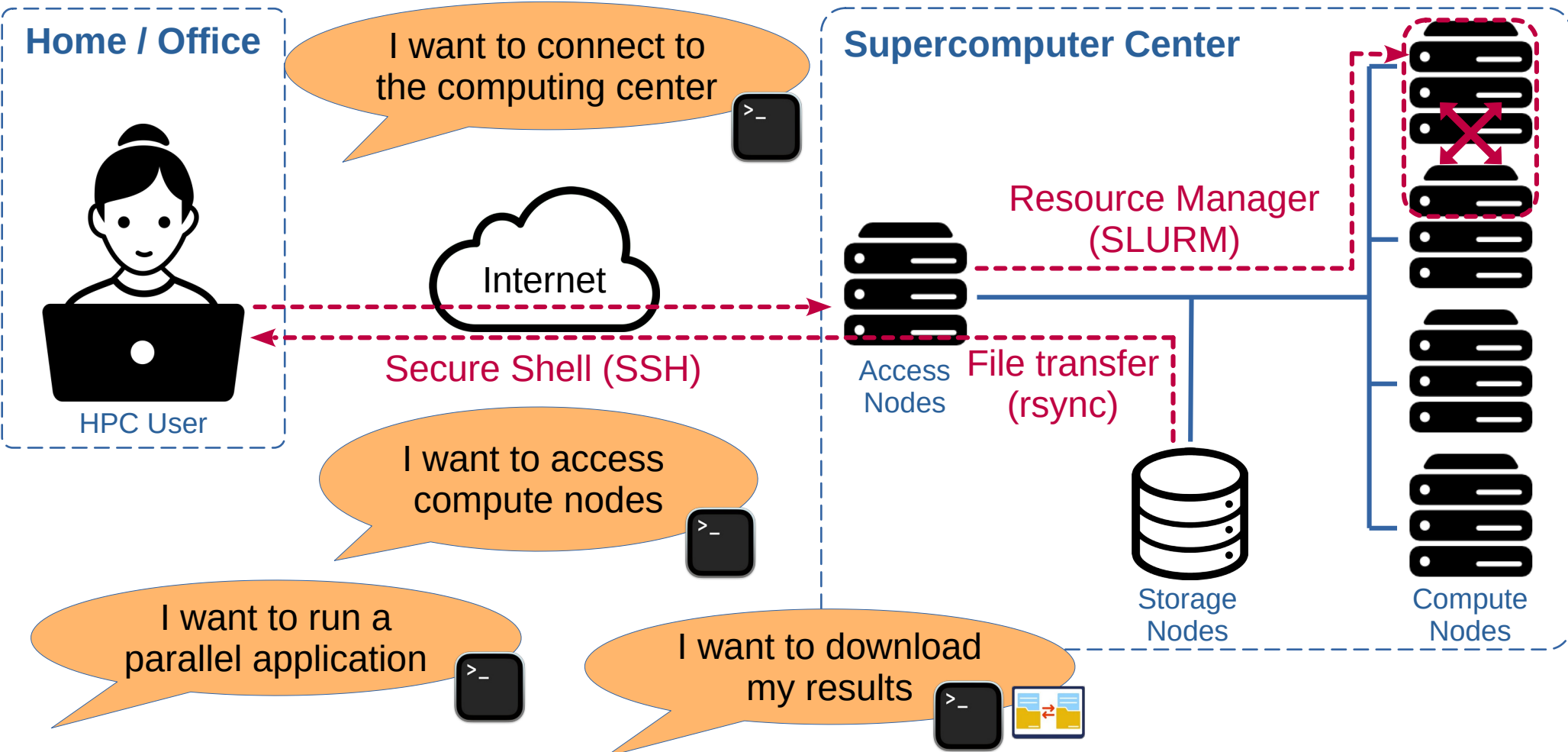
-----
LinkedIn & Twitter : @Luxprovide #meluxina #EuroHPC_JU

#####
# System events, in-progress and upcoming #
#####
# * 2024-02-15 00:00 CET: New 2023.1 software stack default on compute nodes #
# See https://docs.lxp.lu/system/whatsnew/#software-stacks for details #
#####
Last login: Tue Jan 30 16:11:10 2024 from 158.64.79.9
[u100054@login02 ~]$
```



# HPC platform: User point of view

Parallel Computing library (MPI, PyTorch)



# HPC usage: Interactive vs Batch

## Interactive jobs

- **Immediate** access to compute resources
- Commands are run manually (e.g. via an interactive shell, Jupyter notebook, ...)
- Used for debugging, development, testing, or small exploratory runs
- Usually limited to **small number of resources and a short time**

## Batch jobs (non-interactive)

- Execution is **deferred** until resources become available
- No user interaction during runtime
- Used for production workloads, long simulations, large parameter sweeps, ...
- Can use **large number of nodes and long execution time**
- ➔ This is the most standard usage

- In both cases, resource requests (or jobs) are submitted to the **resource manager**
  - Traditionally, HPC do offer infinite jobs or services



# SLURM: Simple Linux Utility for Resource Management



## What is SLURM?

- Open-source, highly configurable, and fault-tolerant workload manager for Linux clusters
- Widely used in HPC to manage and schedule large-scale computational jobs, including on MeluXina

## Main functions

- **Resource allocation:** It allocates resources like CPUs, memory, and storage to submitted jobs based on their requirements and configured policies.
- **Job scheduling:** It queues and prioritizes jobs, ensuring fair access and efficient utilization of resources.

# Main concepts of SLURM



## Partitions

- Group of **nodes with similar characteristics** (CPU type, memory capacity, available software)
- Users submit jobs to specific partitions based on their resource requirements

## QoS (Quality of Service)

- QoS defines **scheduling policies** for jobs within a partition, influencing priority and resource allocation
- For example, limits on job wait time and duration, pre-emption rules

## Jobs

- Jobs are the **units of work** that are submitted to SLURM.
- Each job has a specific resource requirement, such as number of CPUs, memory, and disk space.
- Jobs are submitted within specific partitions and QoS levels.

## Queue

- **Holding area** for submitted jobs within a partition and QoS
- Jobs are added to the queue according to arrival time and priority rules defined by the QoS
- Jobs can be PENDING, RUNNING, COMPLETED (and many more)

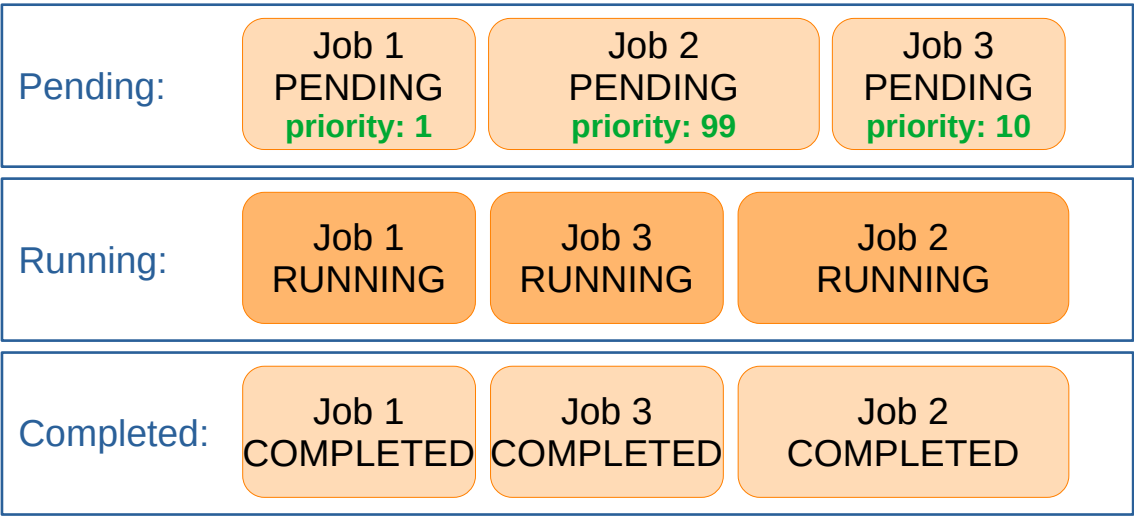
# Job Scheduling

## Simple Example

**QoS:**

<b>default</b>	<b>long</b>
- max 100 jobs	- max 1 job
- max 2 days	- max 6 days
- priority: normal	- priority: low

**Queues:**



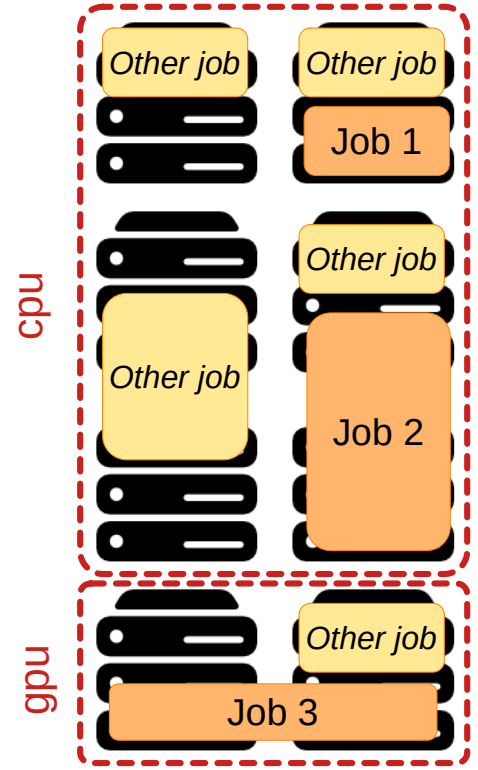
HPC User  
submits jobs

```
salloc -A p20XXXX -N 1 -t 2:00 -p cpu -q default
→ Job 1 for 1 CPU node for 2 hours

salloc -A p20XXXX -N 4 -t 6-0 -p cpu -q long
→ Job 2 for 4 CPU nodes for 6 days

salloc -A p20XXXX -N 2 -t 8:00 -p gpu -q default
→ Job 3 for 2 GPU nodes for 8 hours
```

**Partitions**



Compute Nodes

# Find and Using Software on HPC with

## Software Modules on HPC

- Manage **user environment** using Environment Modules
- Search and make software available to users
- Using `PATH`, `LD_LIBRARY_PATH` and other environment variables
- Various implementations: **Lmod**, C and Tcl Environment Modules

`module avail`: Lists all available modules

`module list`: Shows the loaded modules

`module load <module_name>`: Loads a module to make the software accessible

`module unload <module_name>`: Removes a module from your environment

Other useful commands: `overview`, `spider`, `swap`, `help`, `purge`, `keyword`, ...

→ [https://docs.lxp.lu/first-steps/module\\_system/](https://docs.lxp.lu/first-steps/module_system/)

```
[ @mel0319 ~]$ module avail GROMACS
----- /apps/USE/easybuild/release/2025.1/modules/all -----
GROMACS/2025.2-foss-2025a-CUDA-12.8.0
GROMACS/2025.2-foss-2025a (D)

Where:
D: Default Module
```



# MeluXina in practice

What you need to know to use MeluXina

SCynergy 2026



**EPICURE**  
Unlocking European-level HPC Support

# MeluXina user and project accounts

## User account: `u10XXXX`



It identifies a **person** (you)

- First and last name
- Email

It serves for authentication with

- Password
- One-Time Password (OTP)
- Authorized SSH keys

Datapath:

`/home/users/u10XXXX`

## Project account: `p20YYYY`



It identifies a **project** (that you can be part of)

- Group of users
- Start and end time

It tracks the resources

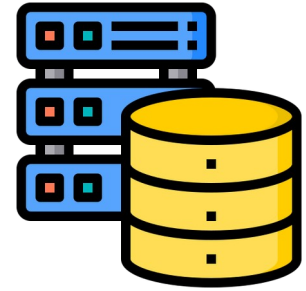
- **Computing resources:**  
cpu, gpu, fpga, largemem
- **Storage resources:**  
scratch (Tier1) and project (Tier2)

Datapath:

`/project/home/p20YYYY`

`/project/scratch/p20YYYY`

# MeluXina shared data storage



**Shared storage** accessible from any compute node

- Tier1: fast but small → temporary storage
- Tier2: slower but larger → permanent storage

**User storage** → personal and limited

`/home/users/u10XXXX` → located on Tier2

**Project storage** → shared with project members, size specific to each project

`/project/home/p20YYYY` → located on Tier2

`/project/scratch/p20YYYY` → located on Tier1

← Preferred location for your work files

**For multi-user projects** → create personal directory within project directory

`/project/home/p20YYYY/u10XXXX`

`/project/home/p201259/workspaces/u10XXXX`

← SCynergy 2026 trainings

# Tracking your resource usage → myquota

```
COMPUTE ALLOCATIONS FOR CURRENT MONTH
COMPUTE USAGE FROM 2026-04-01 to NOW
```

Project	CPU node-hours			GPU node-hours			FPGA node-hours		
	Used	Max	%used	Used	Max	%used	Used	Max	%used
p20 [redacted]	0			0			0		
p20 [redacted]	0	328	0.0%	0	65	0.0%	0		
p20 [redacted]	0			8	130	6.2%	0		

```
DATA ALLOCATIONS
```

Datapath	Data GiB			No. Files		
	Used	Max	Use%	Used	Max	Use%
/home/users/[redacted]	90	100	90%	94566	100000	94%
/project/home/p20 [redacted]	0	1000	0%	142	1000000	0%
/project/home/p20 [redacted]	2428	20000	12%	850390	2000000	42%
/project/scratch/p20 [redacted]	0	1024	0%	1	1000000	0%
/project/home/p20 [redacted]	0	10240	0%	1467	1000000	0%

Compute

Data

# Authentication & Access to MeluXina

## SSH Key

- Fast, lightweight, and secure
- Allows password-less access
- Most common on HPC platforms
- Requires an advanced configuration



Public SSH key to be upload via the service desk

Used for frequent and password-less operations

- **Shell access** via native terminal applications
- File transfer and synchronization

## Username + Password (+ OTP)

- Beginner friendly, easier to setup
- Similar to usual online web-service
- Manual authentication process

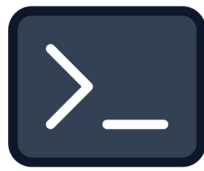


Username: `u10XXXX`  
Password: defined at service desk setup  
OTP: defined at first use

Used for interactive and graphical accesses

- **Web-portal access** (also includes a shell access)
- Simple file transfer only

# Shell Access via SSH



## Configuration

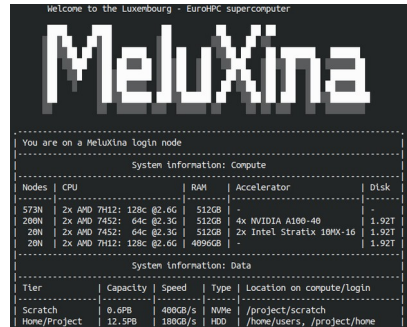
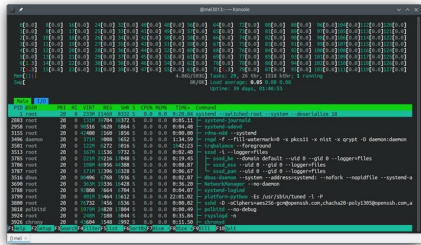
- Public SSH key to be added via service desk
- Configuration file `~/.ssh/config`

## Access

`ssh meluxina`

## Documentation

- [Generating an SSH key](#)
- [Uploading your public SSH key](#)
- [Connect to MeluXina](#)



# Web-portal Access



## Configuration

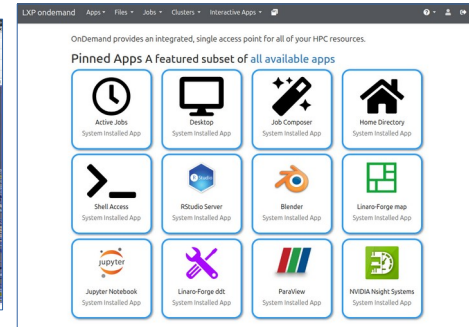
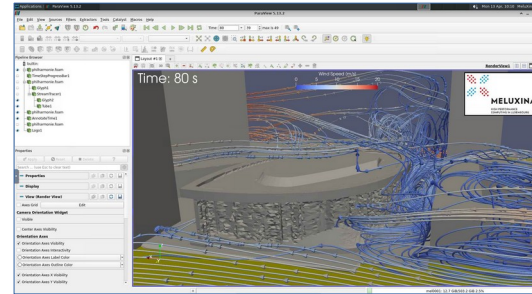
- Password setup in service desk
- One-Time Password (OTP) setup at first access

## Access

<https://portal.lxp.lu/>

## Documentation

- [How to connect to Open OnDemand](#)
- [Multi-factor authentication setup](#)





# Hands-on!

<https://luxprovide.github.io/SCynergy2026-GettingStartedWithMeluXina/>

- Connecting the MeluXina
- HPC workflow: Urban wind simulation
- AI workflow: Deep learning with PyTorch

