CSCS
Centro Svizzero di Calcolo Scientifico
Swiss National Supercomputing Centre

ETH zürich

Sarus Suite and **Podman:**

Native HPC Speed Meets **Cloud-Native Productivity**

HPC-AI Swiss Conference 2026

Gwangmu Lee, CSCS

22.04.2026

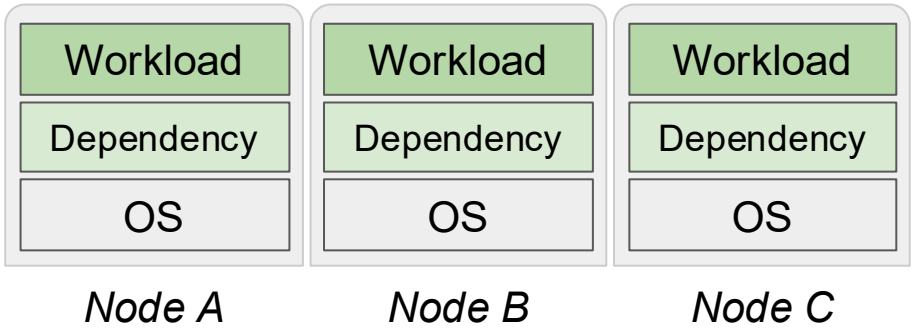
We at CSCS…

- National **supercomputing** centre.
- Since 1991, serving…
  - Large-scale simulation
  - Scientific modeling
  - AI/ML training
    (relatively recent addition)

"Workloads"

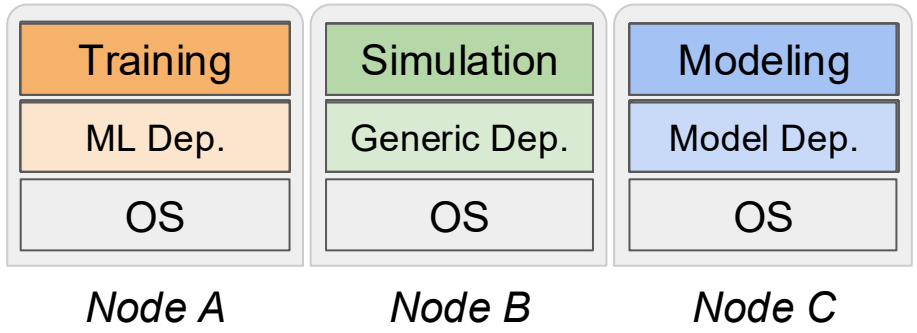
HPC Software Stacks

- Back in the days…



  - **Very static and limited sets.**
  - "Let's install them all!"
  - That's what "Modules" is for:
    Install everything, and let users choose.

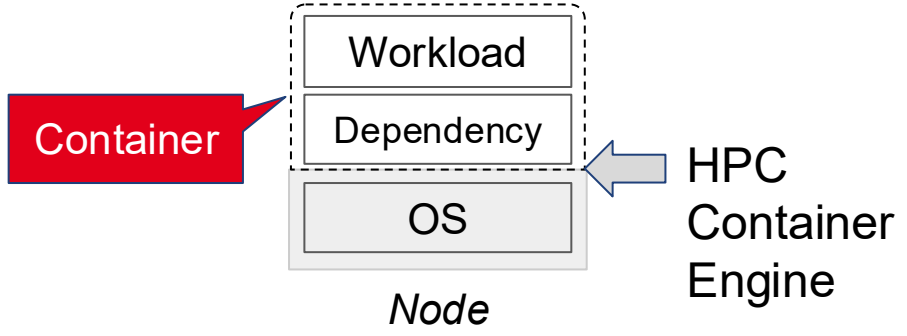
- Nowadays…



  - **Highly variable and customized.**
  - We can't just install them all.
    (Too many *new* workloads every day)
  - What should we do?

Solution: Containers for HPC?

- Containers: great for custom SW stacks.
- That's why we have **HPC container engines** (e.g., Enroot).
- Caveat: **ground-up** developed for HPC.
- Meanwhile, **general-purpose engines**:
  - Evolving, growing, and improving.
  - Heading toward **cloud-native computing**.



e.g., NVIDIA Enroot

Benefits of Cloud-Native Computing

- Integration with dev. productivity solutions

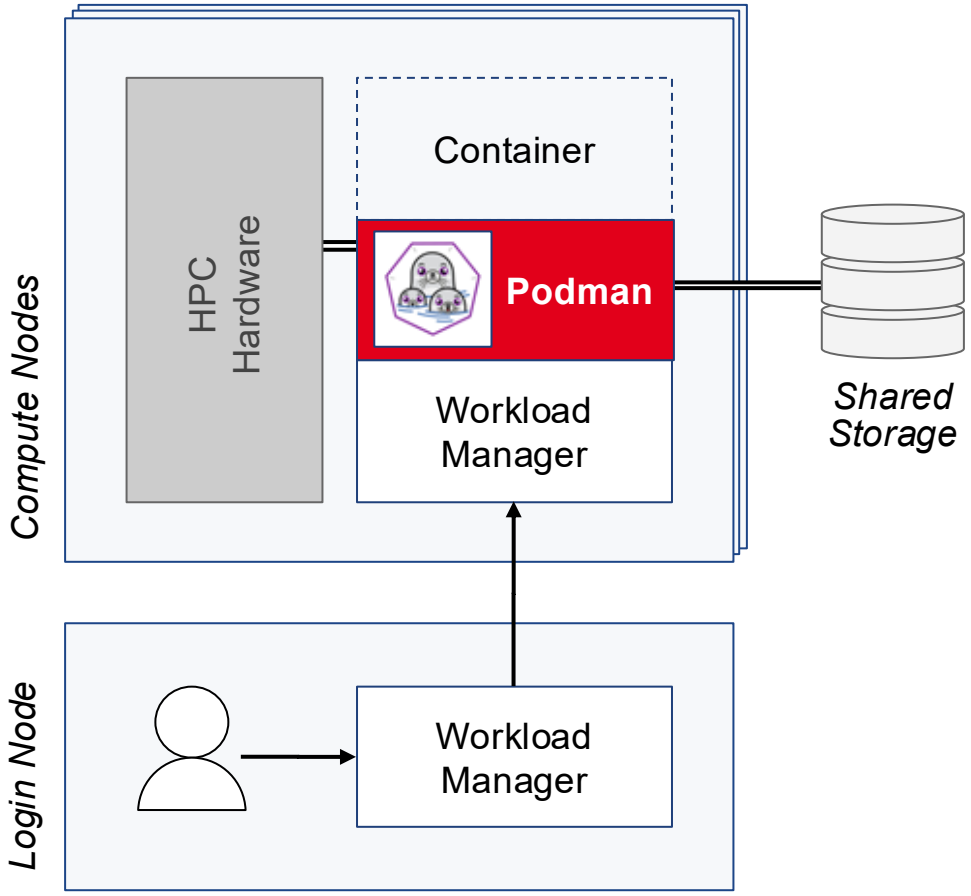
- Alignment to external projects

HPC container users would want them, too.

How would we support them?

- Open to a wider ecosystem
  (community, standards, features, …)

Solution?: Using General-Purpose Engine As-Is?

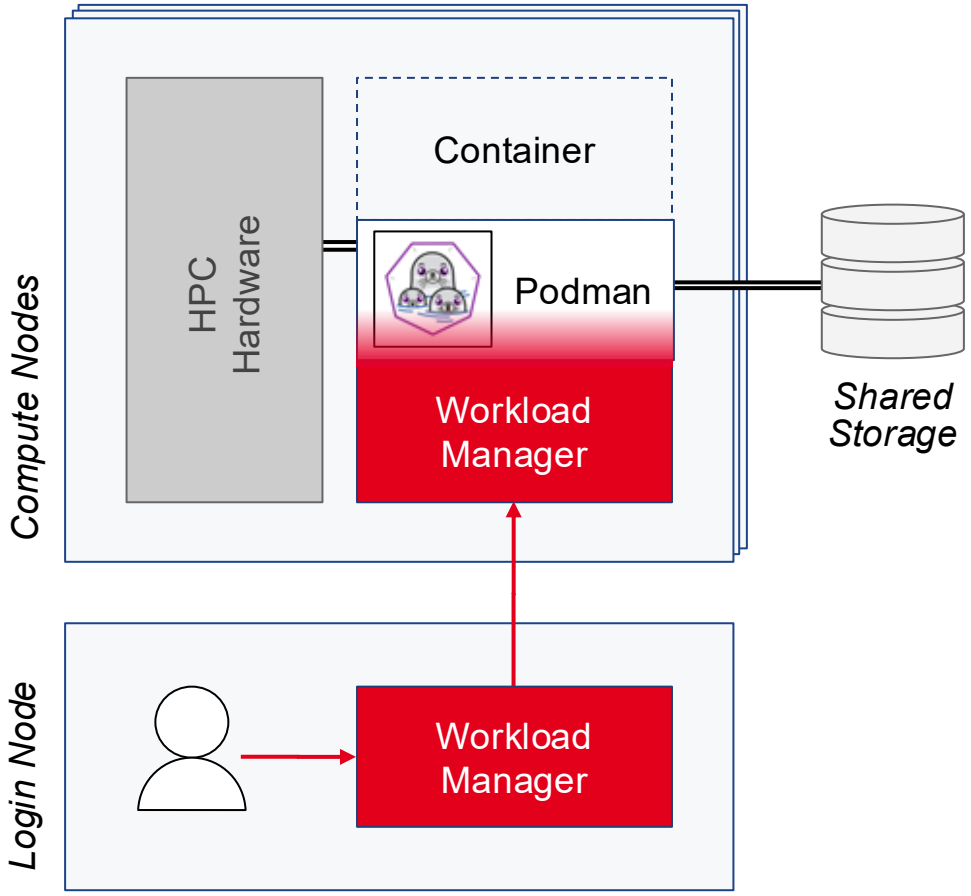


- Candidate: **Podman**
  - Modular & Configurable
  - Secure (rootless)
  - Easy to manage (daemonless)
  - Standard (OCI) compliant
  - Large community
  - Active development
  - …
- Q: Then why do we have HPC container engines?

HPC Problems with General-Purpose Engines

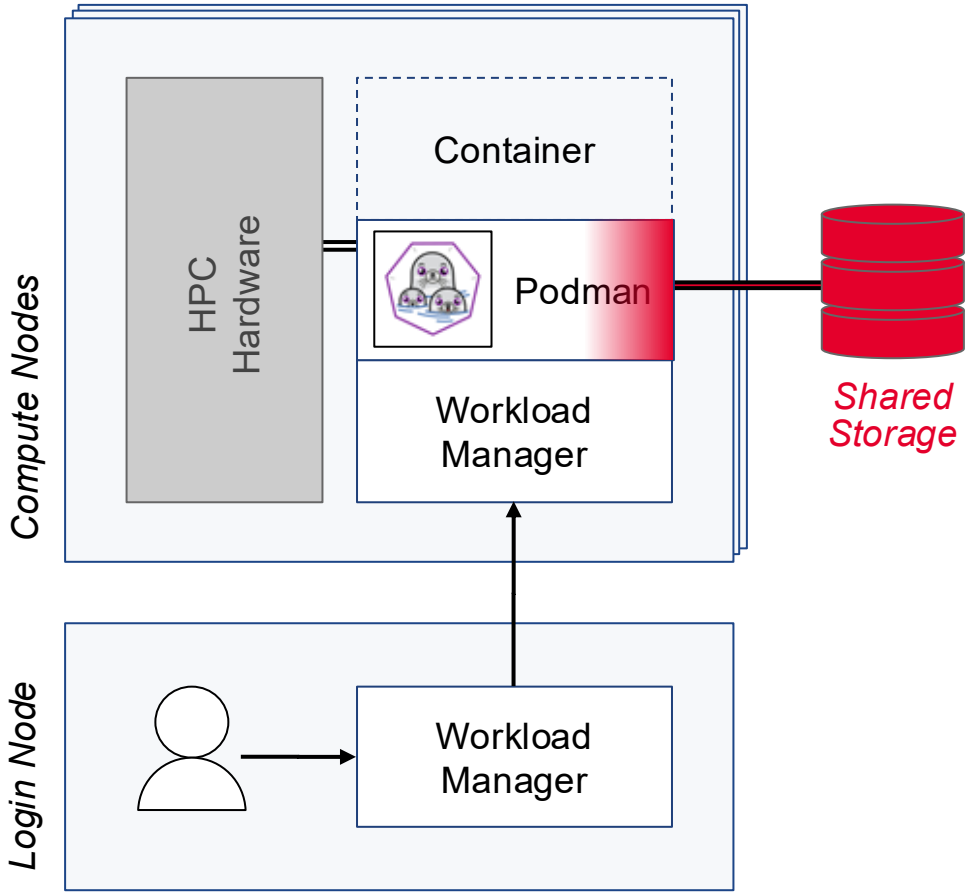
1. Integration with HPC workload manager (e.g., Slurm)
2. Compatibility with HPC shared storage.
3. Enabling HPC-grade performance.

HPC Problem 1: Integration with Workload Manager (WLM)



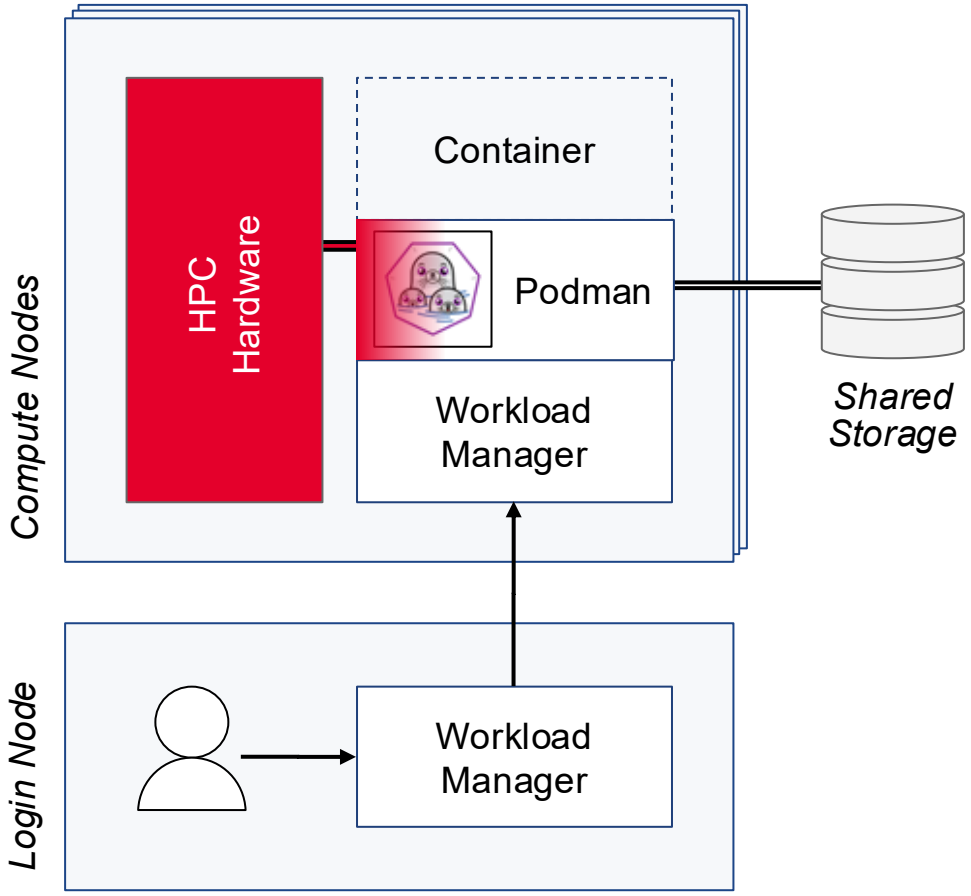
- HPC workloads: **batch-job** centric.
- WLM: launch batch jobs **on nodes**.
- Containers: should launch too.

HPC Problem 2: Compatibility with Shared Storage



- HPC workloads: uses **shared stor.**
- General-purpose containers: uses **multi-layered filesystem**.
- Multi-layer filesystem + shared stor: Awful performance.

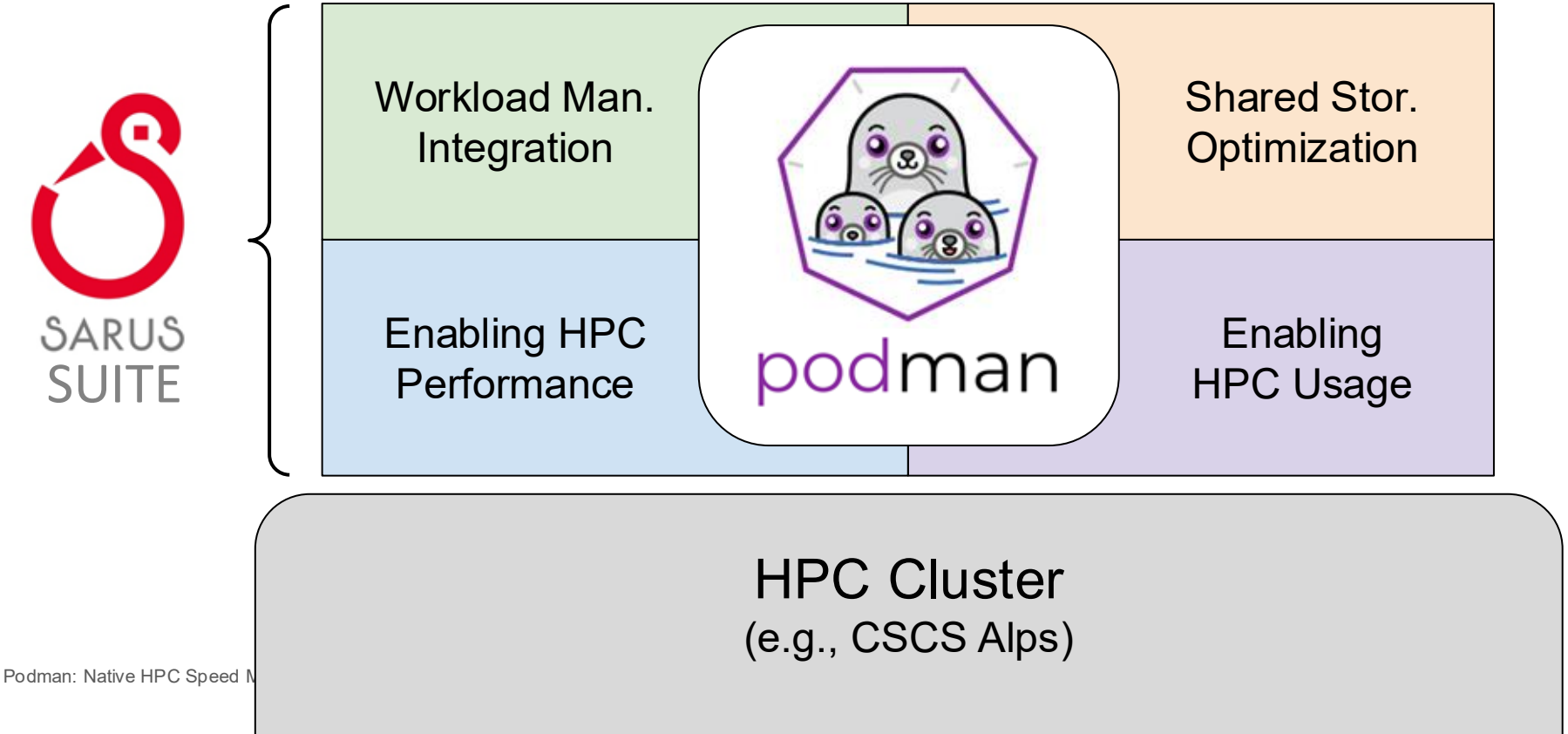
HPC Problem 3: Enabling HPC Performance



- General-purpose containers: **strict isolation** by default.
- For HPC workloads: special HW/libs on the host, relaxed isolation (e.g., network)…

Sarus Suite: Summary

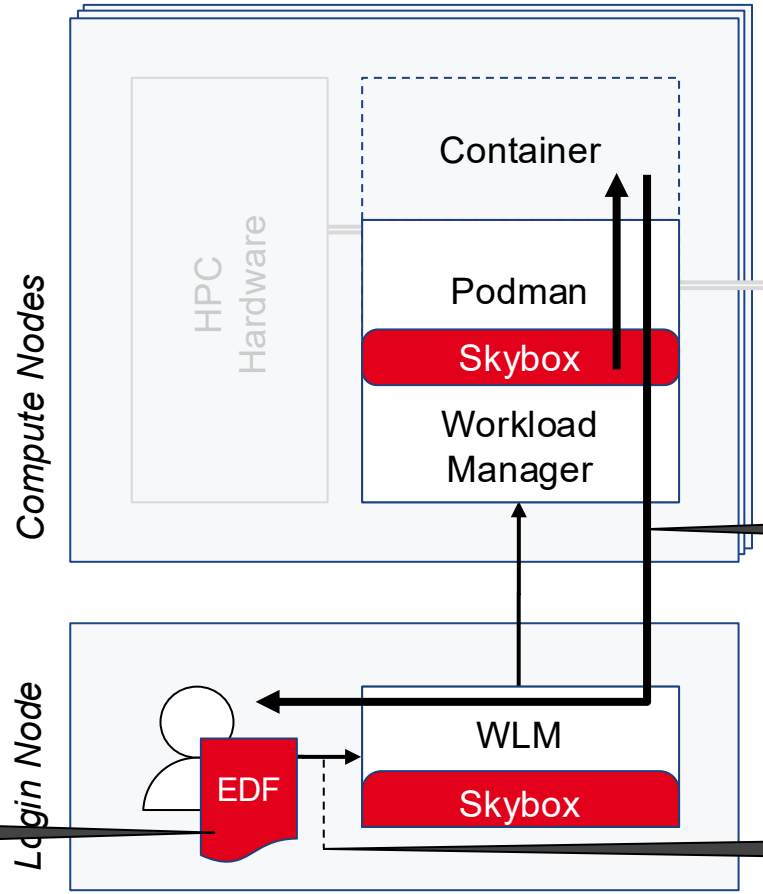
- Put Podman (General-purpose engine) at the center.
- Augment (not modify) it to address HPC problems.
- Plug it into HPC clusters.



Sarus Suite: Integration with Workload Manager (WLM)

(HPC Problem 1)

- **EDF (Environment Definition File)**: Defines container environment.
- **Skybox**: WLM plugin. Launches containers.




```
iteration 1/ 25 | consumed samples: 128 | throughput per GPU (TFLOP/s/GPU): 281.5 |
iteration 2/ 25 | consumed samples: 256 | throughput per GPU (TFLOP/s/GPU): 613.9 |
iteration 3/ 25 | consumed samples: 384 | throughput per GPU (TFLOP/s/GPU): 630.9 |
...
```

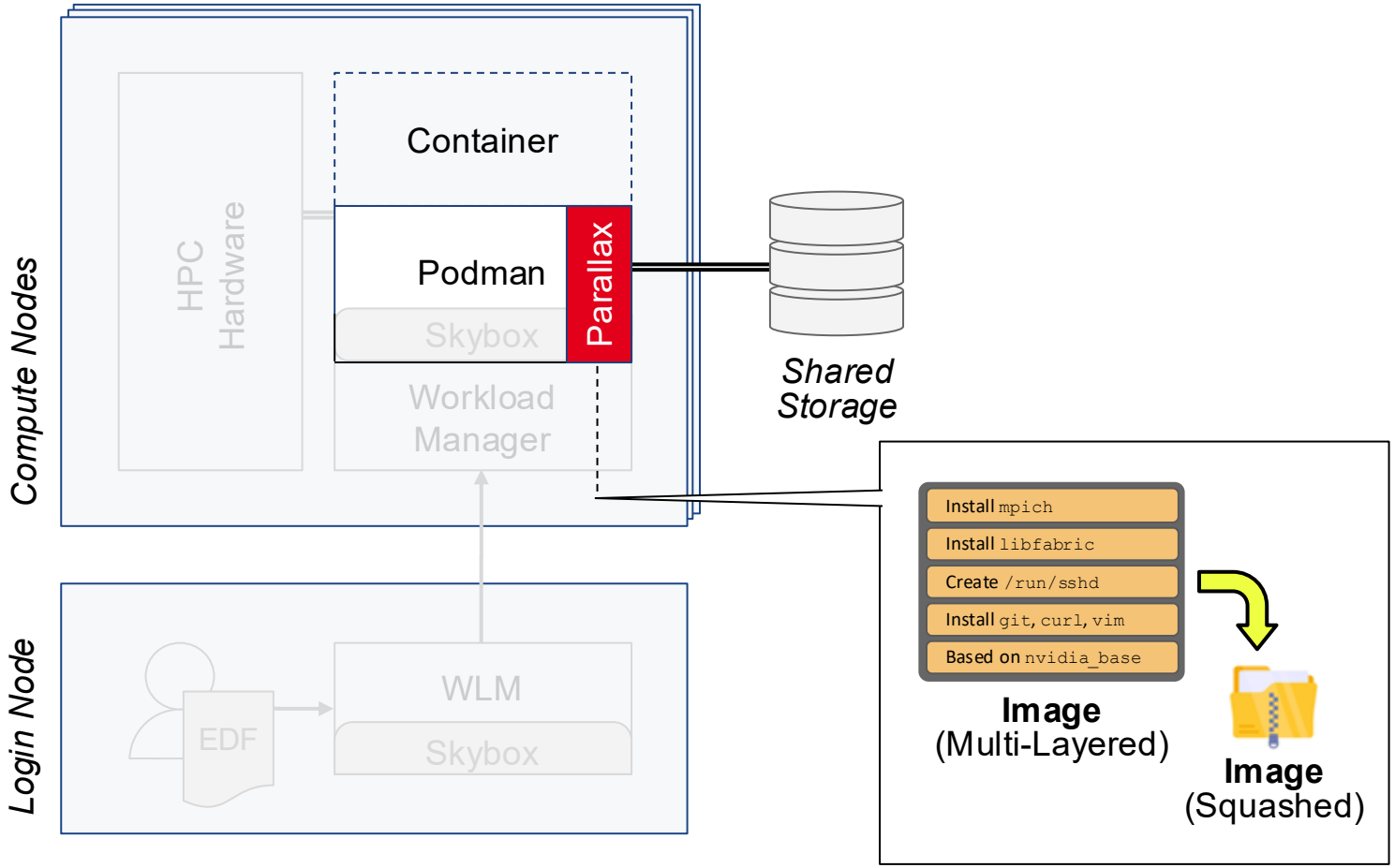
```
image = "ngc-pytorch"
mounts = [<shared-stor>]

[annotations]
nccl.enabled = "true"
```

```
user@login_node:~$ srun \
  --environment=<EDF>   \
  torchrun …
```

Sarus Suite: Compatibility with Shared Storage

(HPC Problem 2)

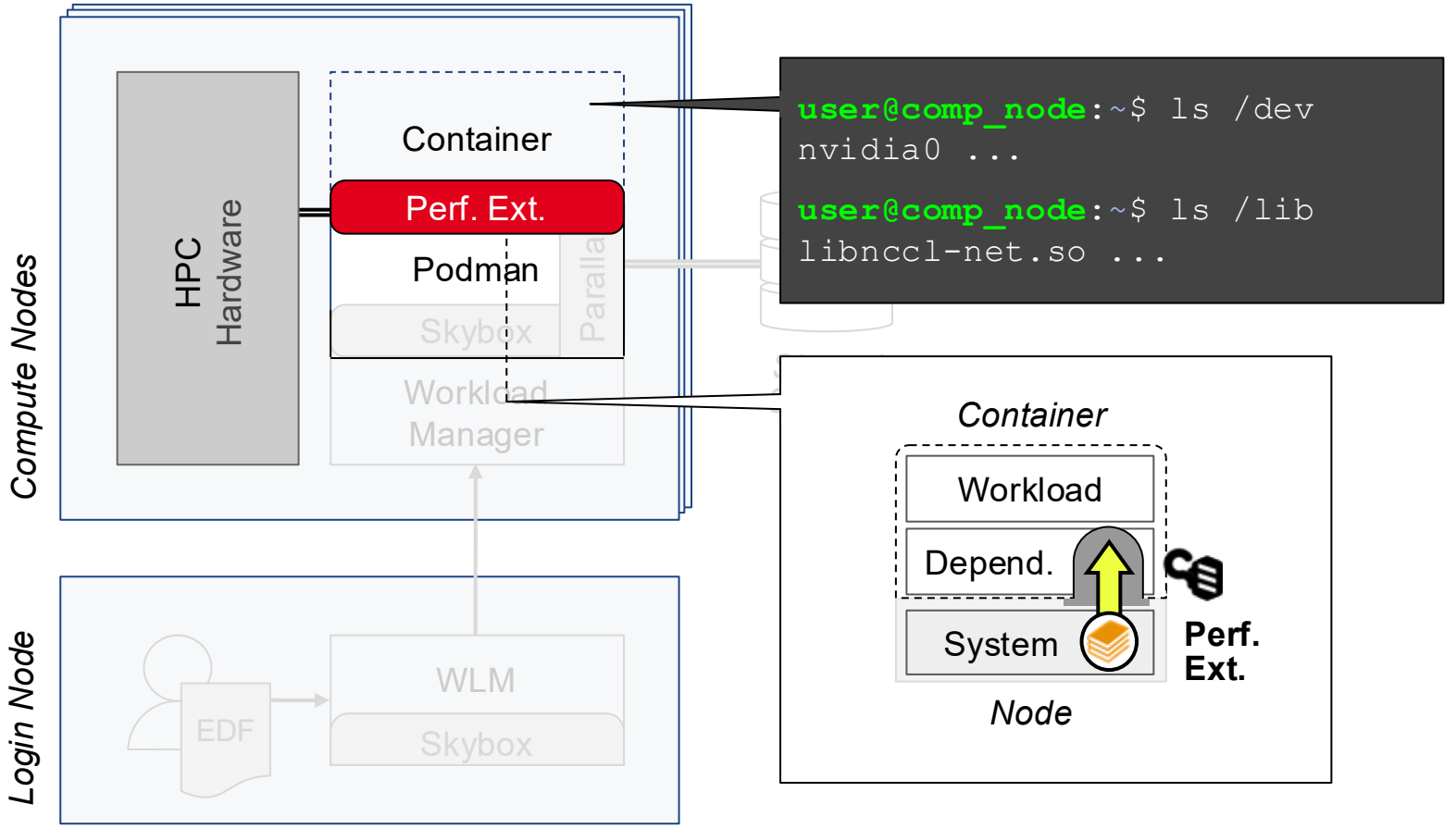
- **Parallax**:
  utility for Podman-compatible image store on shared stor.
  (“SquashFS”)



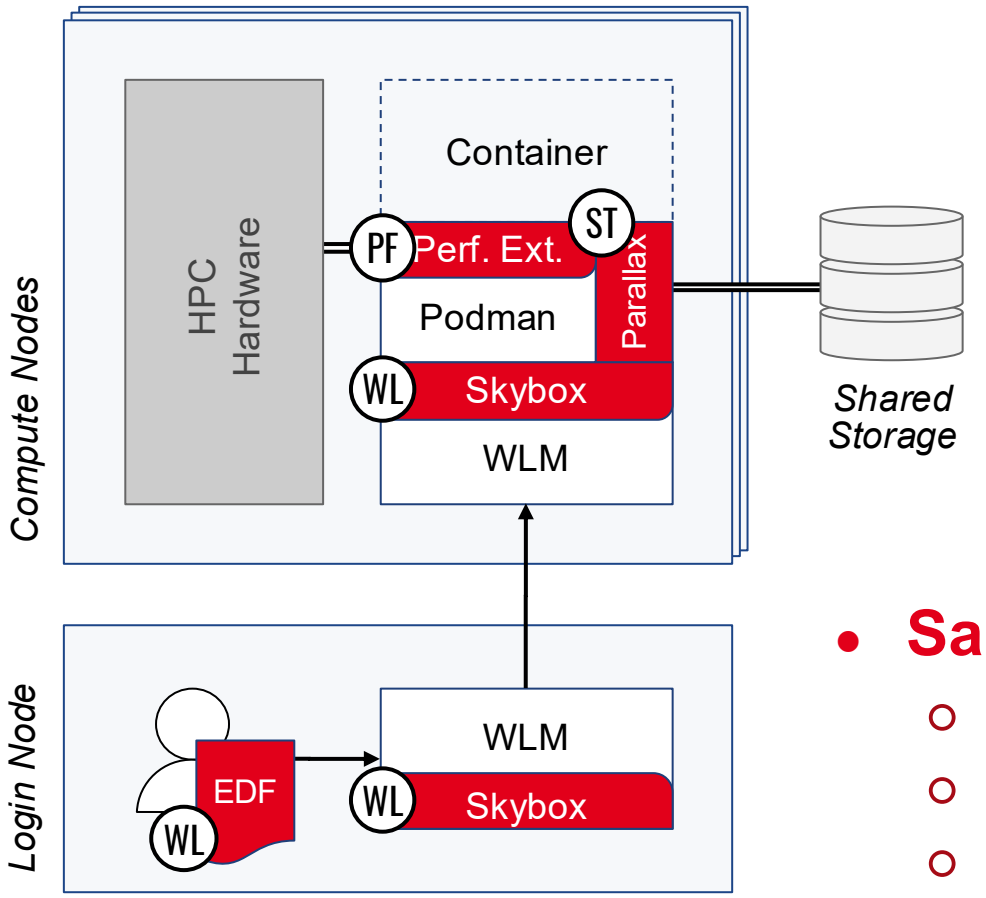
Sarus Suite: Enabling HPC Performance

(HPC Problem 3)

- **Performance Extension**: HPC-friendly configuration and HW/library injection



Sarus Suite: Technical Overview



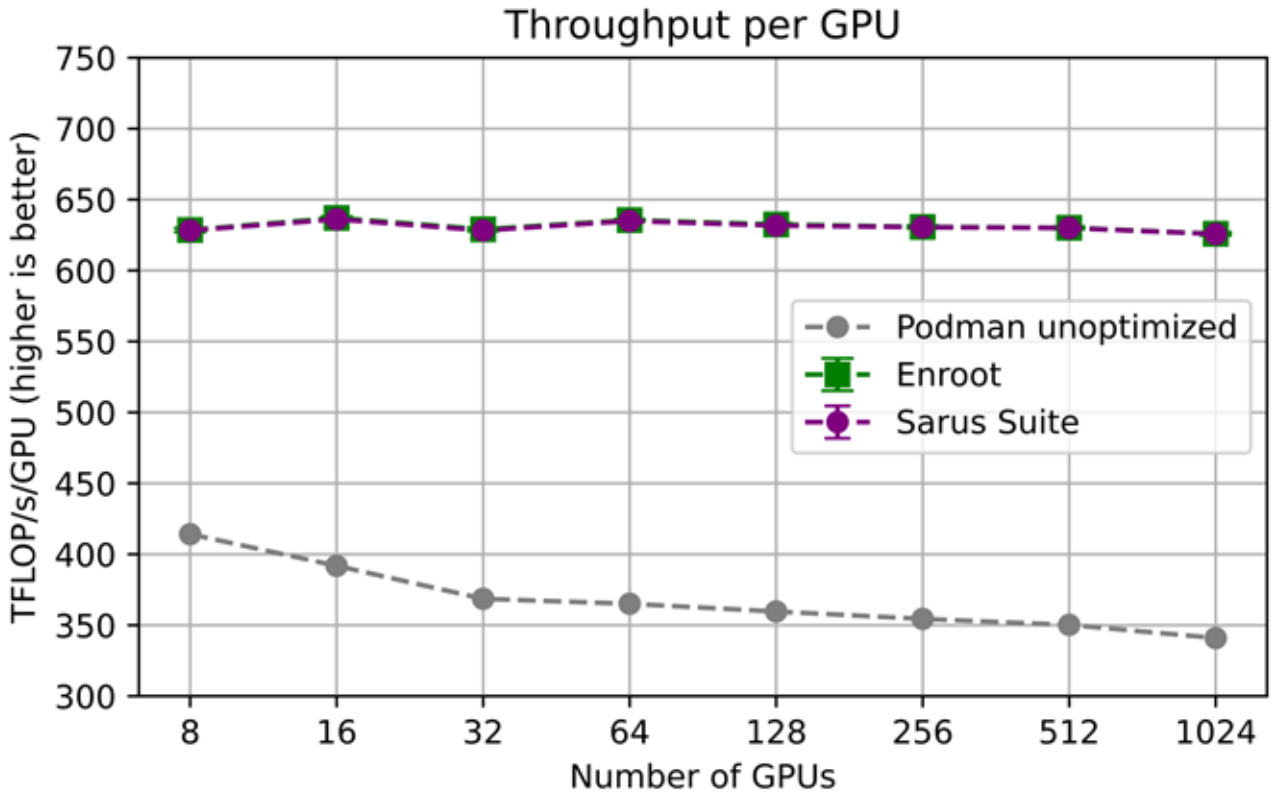
- **Sarus Suite** components
  - (WL): Integration with WLM
  - (ST): Compatibility with shared stor.
  - (PF): Enabling HPC performance

Evaluation

- **Value** of Sarus Suite
  - HPC containers *augmenting* general-purpose engine.

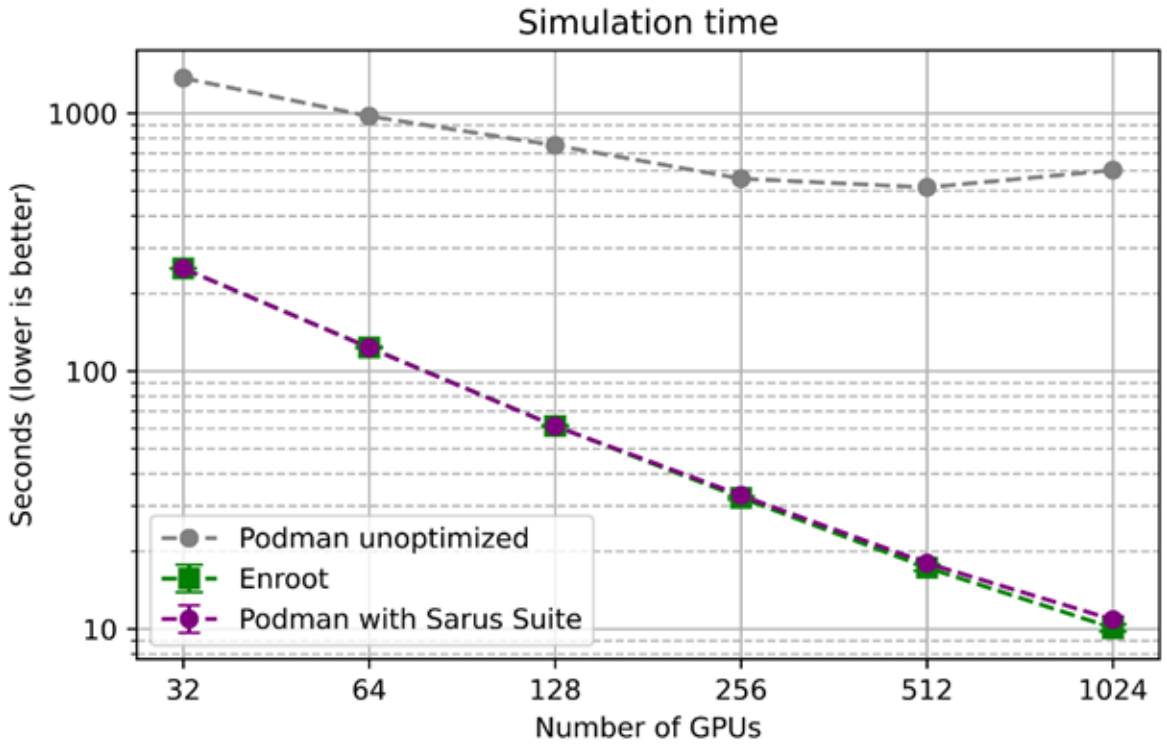
- **Points of interest**
  - Enables HPC workload?
    → Using 3 representative HPC workloads. (AI/ML, Simulation, HPC benchmark)
  - Enables HPC performance?
    → Spoiler: **same** as the state-of-the-art HPC container engine, Enroot.
  - Enables HPC usages?
    → Evaluation done on the current production env. at CSCS.

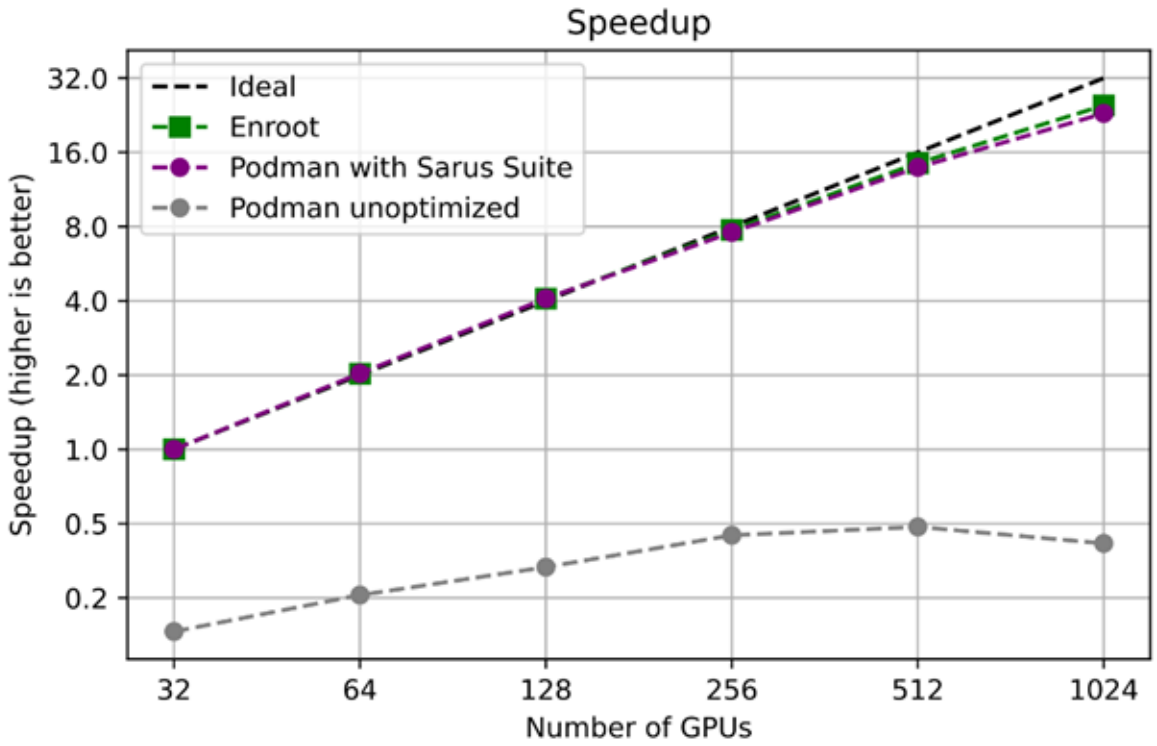
Evaluation: Megatron-LM (LLM pre-training)



Container: NGC PyTorch 25.11 + Megatron-LM 0.15.2
Test case: LLama 3 8B pre-training example from Megatron-LM repository (4 ranks per node, 10 repetitions)
System: Alps Infrastructure - Machine Learning vCluster (4 x NVIDIA GH200, Slingshot 11 200Gb)

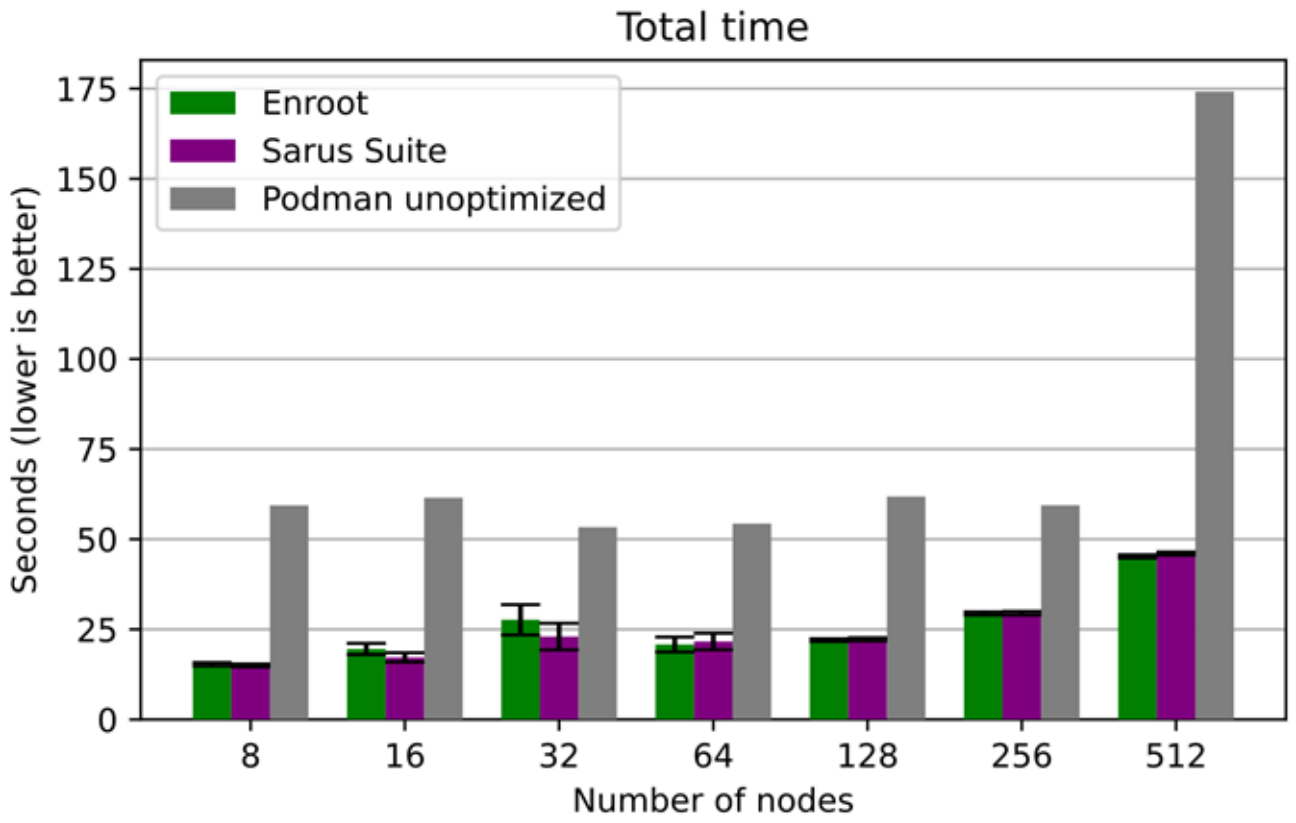
Evaluation: PyFR (Flux Reconstruction)





Container: PyFR 2.1, OpenMPI 5.0.7, CUDA 12.8, Ubuntu 24.04
Test case: 3D Taylor-Green vortex (4 ranks per node, 10 repetitions)
System: Alps Infrastructure - User Lab vCluster (4 x NVIDIA GH200, Slingshot 11 200Gb)

Evaluation: Pynamic (Storage Filesystem Benchmark)



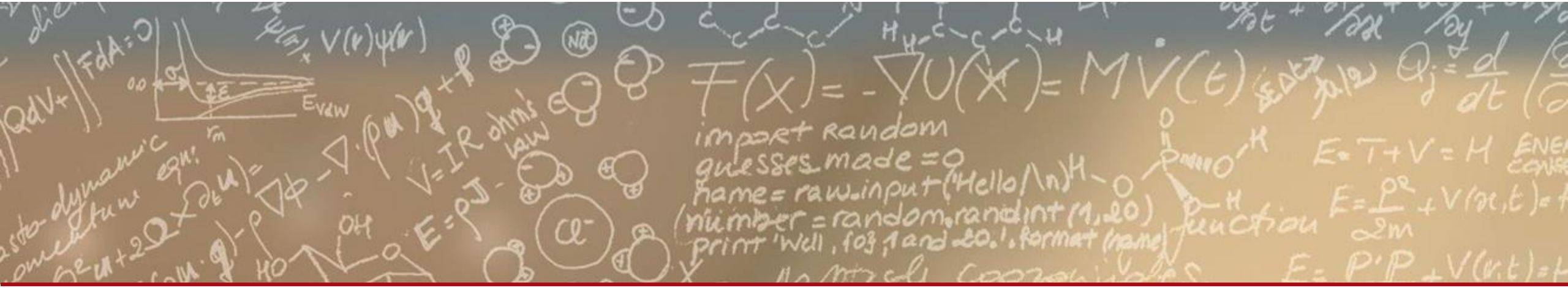
Container: Pynamic w/ Python 3, OpenMPI 5.0.7, Debian 12 (Bookworm)
Test case: Simulated import and visit of 495 Python modules with avg. 1850 functions each (4 ranks per node, 10 repetitions)
System: Alps Infrastructure - Machine Learning vCluster (4 x NVIDIA GH200, Slingshot 11 200Gb)

Conclusion

- HPC-specific container engines:
  impossible to follow general-purpose ones.
  (Cloud-native support -- gone)

- **Sarus Suite's approach**
  - HPC container solution that augments general-purpose engine (Podman).
  - Augmentations for HPC:
    - WLM integration
    - HPC performance
    - Shared stor. support
    - HPC usage

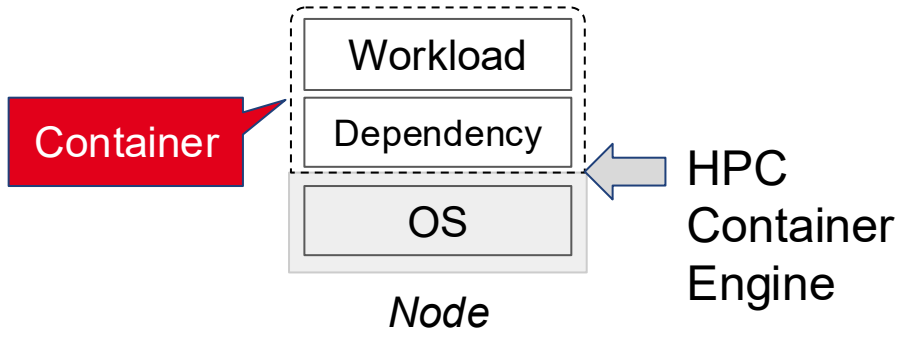
- Planned adoption in CSCS in a few months.



Backup Slides

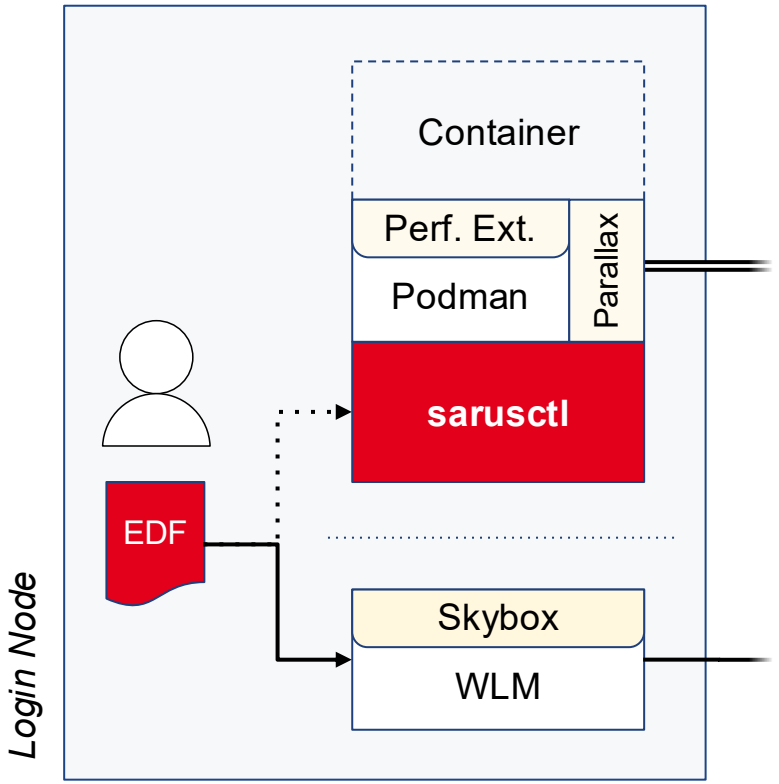
Solution: Containers for HPC?

- Containers: great for custom SW stacks.
- That's why we have many **HPC container engines**.
- Caveat: ground-up developed for HPC.
- Meanwhile, **general-purpose engines**:
  - Evolving features and standard
  - Growing community support
  - Improved integration with external components
- HPC users would want them, too.
  How would you keep up with them?



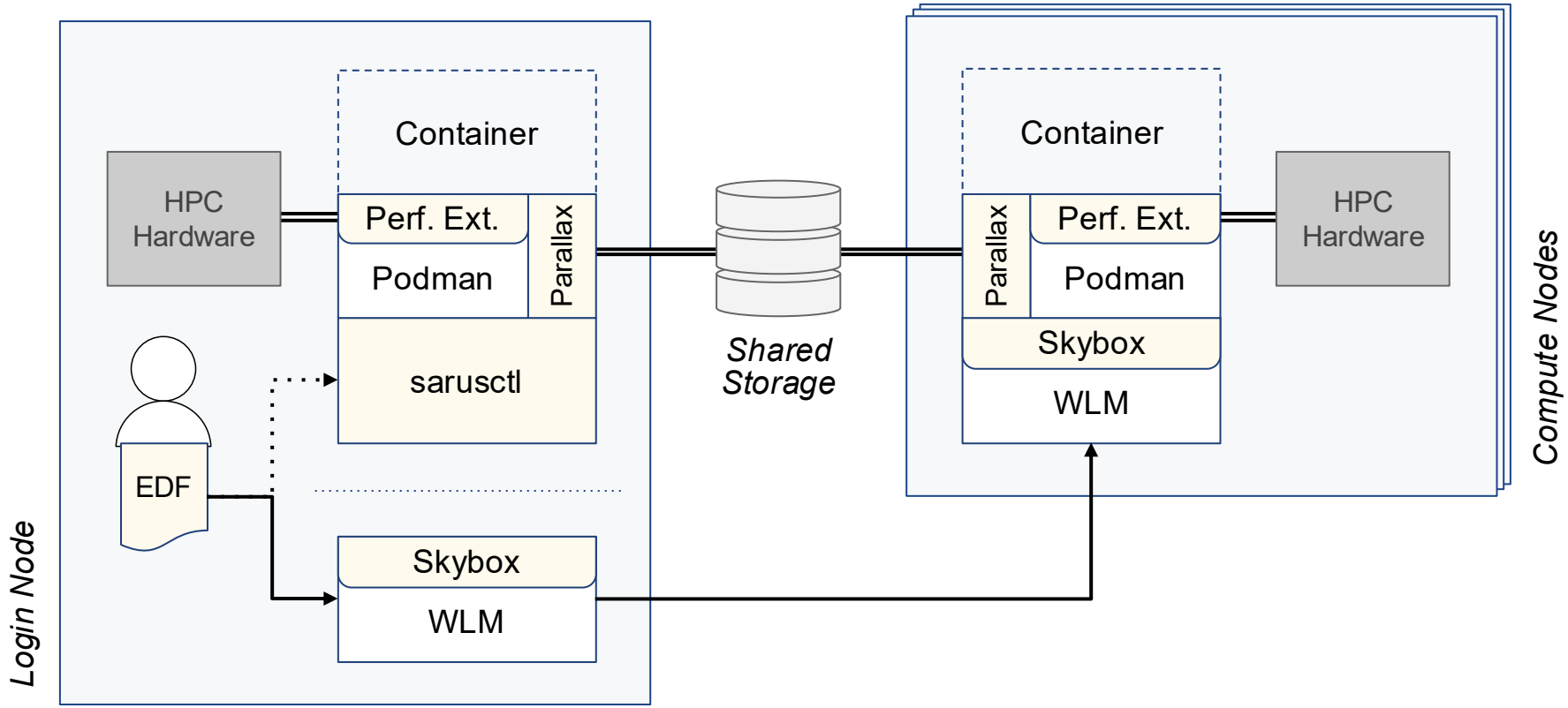
e.g., NVIDIA Enroot

Enabling HPC Usage



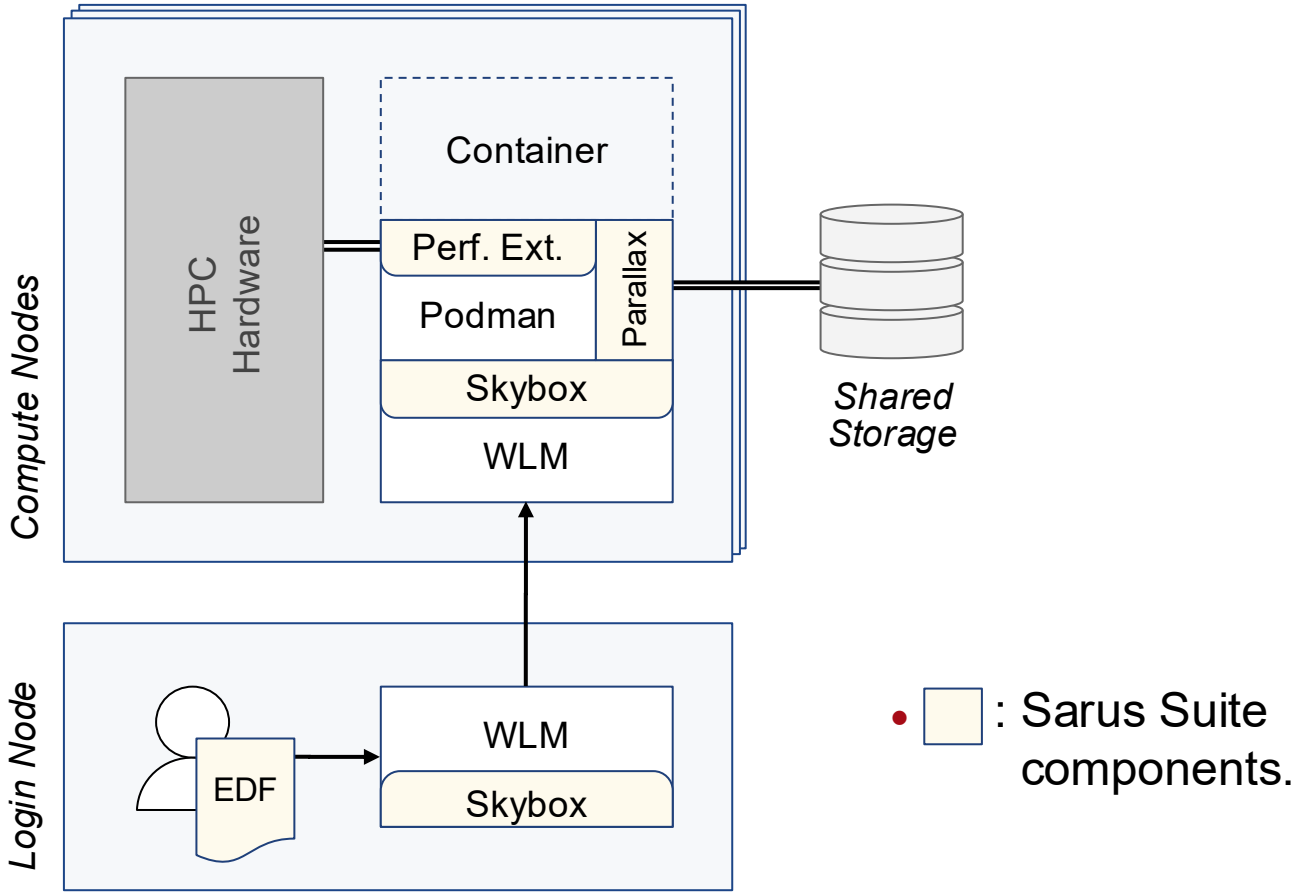
- Rationale
  - HPC workloads: batch-job centric.
  - “I want this environment in my batch.”
  - “I want to check my environment before.”
- Sarus Suite components
  - **Environment Definition File (EDF)**: specifies the environment *inside* the container (e.g., image name, mounts, …)
  - **sarusctl**: provides a preview of the environment defined by EDF.

Sarus Suite Overview



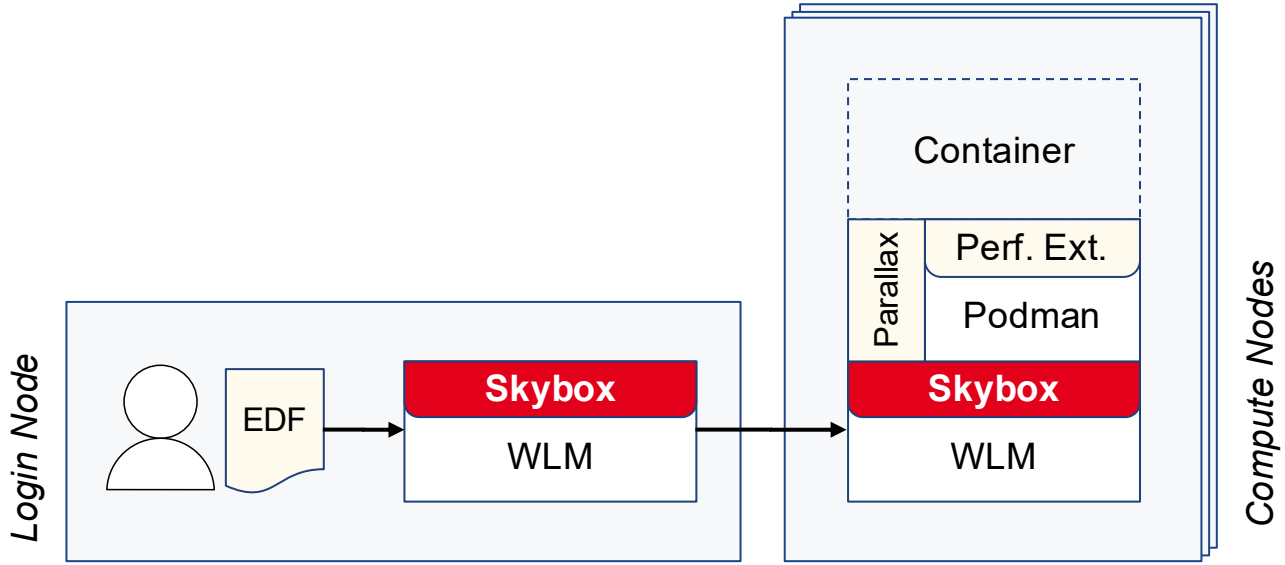
- : Sarus Suite components.

Sarus Suite Overview (w/o sarusctl)



Sarus Suite: Integration with Workload Manager (WLM)

(HPC Problem 1)

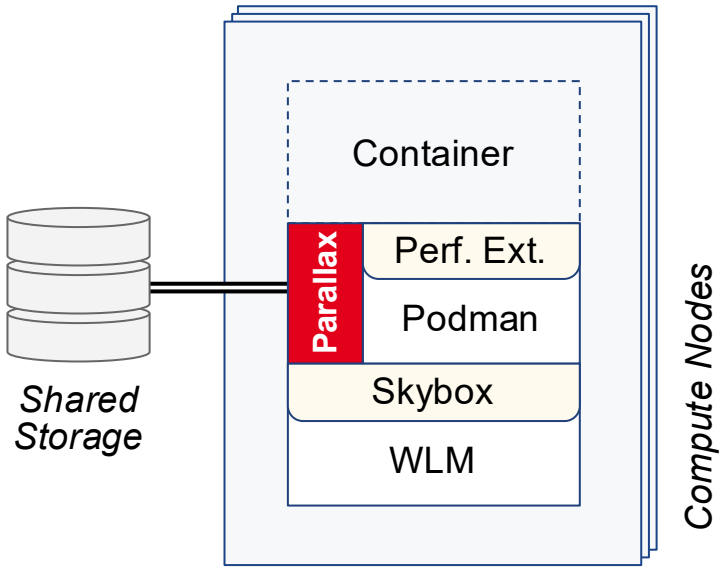


- Rationale
  - HPC workloads: batch-job centric.
  - WLM: *launch* batch jobs on nodes.
  - **Containers: should be launched, too.**

- Sarus Suite components
  - **Skybox**: WLM plugin.
    (Launches containers on nodes.)

Sarus Suite: Compatibility with Shared Storage

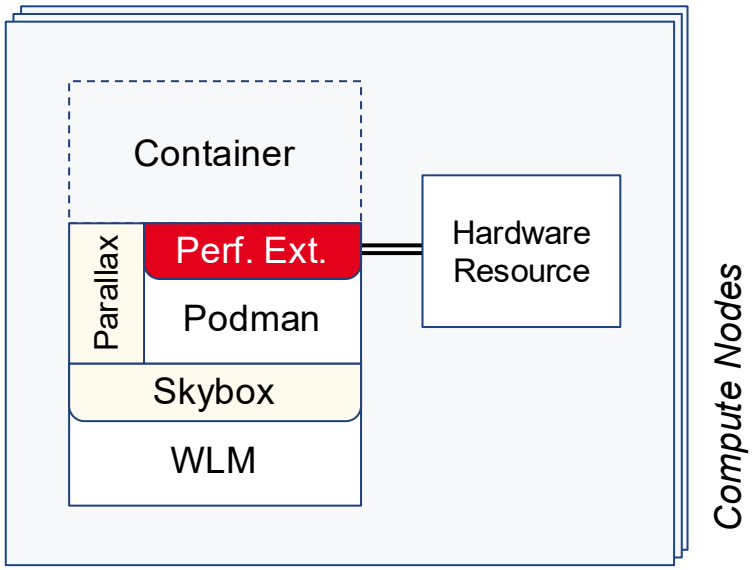
(HPC Problem 2)



- Rationale
  - HPC workloads: uses parallel storage.
  - General-purpose containers: uses multi-layered filesystem.
  - Horrible performance on parallel stor.
- Sarus Suite components
  - **Parallax**: parallel-storage-friendly container image converter. (multi-layered to layer-less)

Sarus Suite: Enabling HPC Performance

(HPC Problem 3)



- Rationale
  - General-purpose containers: strict isolation by default.
  - For HPC workloads: special HW resources, configs, libs, …
- Sarus Suite components
  - **Performance Extension**: HPC-friendly configuration and binary injection logic (“hooks”)