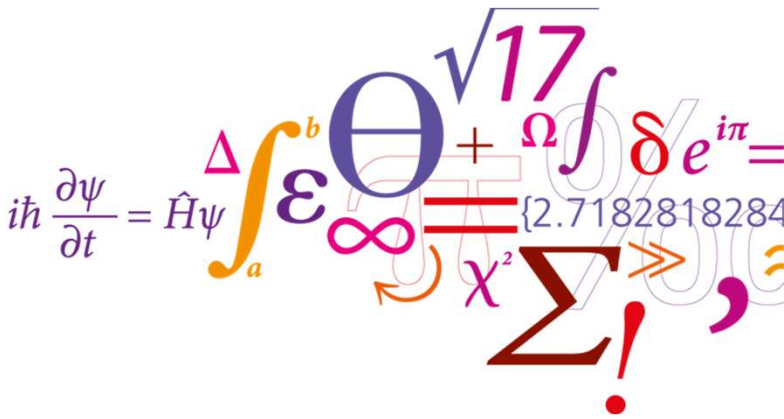
Installing and managing an HPC center by two people

Dr. Ole Holm Nielsen
Department of Physics
Technical University of Denmark (DTU)

Email: Ole.H.Nielsen@fysik.dtu.dk
Wiki: https://wiki.fysik.dtu.dk/Niflheim_system/
Slurm: https://github.com/OleHolmNielsen/Slurm_tools



DTU Fysik
Institut for Fysik

Who are we? The *Niflheim* HPC center at Technical University of Denmark (DTU)

- A **university departmental HPC facility** operating since 2002.
- Funded by university professors' **external grants** (university co-funding until 2021).
- General **national HPC resources in Denmark** are tiny:
  Small allocations are available at Danish university HPC centers and at LUMI (EuroHPC).
- *Niflheim* delivered **244 million CPU core-hours** in 2025 (~10x national resources).

Why operate our own HPC center?

- Our professors decide group allocations and user priorities **with Zero bureaucracy**! Planning is easy for 3-5 year research projects without making grant applications.
- Our users are primarily from **established research groups** at the university.
- Each group has "power users" and professors with **discipline-specific knowledge** about the software tools needed. New users receive introduction to HPC from their group.
- Our **user support** is therefore mainly limited to account management, installing software, and solving system related issues.
- **Best value for money:**
  Our internal **full-cost calculation**: **7 Euro** / 1000 CPU core-hours.
  - *Azure* D96ads V6 (AMD Genoa 96-core, 384 GB, disk): **90 Euro** / 1000 CPU core-hours using **Pay as you go** (Norway) + network + storage + cloud infrastructure. Alternatively using **Spot price: 16 Euro.**

Description of *Niflheim*

- **Total investment** in servers of ~7 million Euro.
- Provided by our university: Server room incl. environment monitoring, electricity, cooling. Internet connection. Central user database. EU-tender assistance.
- We manage by ourselves: System configuration, procurements, physical installation, automated OS installation, and cluster operations by **only 2 people** (both are Physics PhDs).

- **HPC system:** ~700 compute nodes, 13 NFS file servers (~2 PB data) with daily backups.
- Several CPU generations from 96-core AMD Genoa 9474F to old 24-core Intel Broadwell. A total of 31.700 CPU cores with **2.3 PFLOPS theoretical peak performance**. Compute nodes are usually retained for up to 10 years (space permitting).
- GPUs: 8*H200, 16*A100, 80*RTX3090, 2*AMD MI210.
- ***Direct Water Cooled*** Lenovo ”Neptune” servers.
- Omni-Path and NVIDIA/Mellanox interconnects. Ethernet 100 Gb/s backbone.
- **No** Lustre or other complex storage systems.
- Power usage: About 400 kW.

Niflheim design principles

- **Keep It Simple, Stupid** (the well-known ”KISS principle”)!
- We don’t buy any ”fancy” hardware or software unless it is absolutely required.
- Configuration of compute node hardware is optimized for our users’ codes:
  Fewer and faster CPU cores with a large RAM memory (256 GB, 768 GB, or up to 4096 GB).
- We almost entirely use Open Source tools. A few commercial codes are procured by users.
- Maximizing the application code performance per amount of investment is
  very important for scientific productivity.

What does *Niflheim* offer to the end users?

Our ~250 end users are offered this software stack:

- RockyLinux 8 OS.

- Software environment modules provided by **EasyBuild** (EB).
  We build application code EB modules by requests from users.
  The 1000s of modules provided by EasyBuild is a **really fantastic concept**!

- Many users create a Python *venv* which loads optimized EB modules.
- Other users with Machine Learning (ML) projects may prefer to install their own personal Python *venv* or *Conda environment* (for PyTorch etc.) in stead (**not** using EB).

- Each group has home-directories in a dedicated NFS file server. Scratch file servers as well.

- A user documentation Wiki page is at https://wiki.fysik.dtu.dk/Niflheim-users/
  Mirrored at https://niflheim-users.readthedocs.io/en/latest/

EasyBuild deployment at *Niflheim*

- We deployed EasyBuild inspired by the tutorial session held at the SC17 conference.
- Module hierarchy: We chose a **flat layout**.
- We decided to make **separate module trees for every CPU architecture** in our cluster.
  - Note: The **cpuid** tool from https://etallen.com/ is the simplest and most reliable way to detect your CPU architecture: `cpuid -1 | grep '(synth)'`
  - Nodes make NFS automount of EB modules for the appropriate CPU architecture.
- When deploying EB on CentOS (and later on RockyLinux) compute nodes, **every single installation step was documented** in our Wiki page at https://wiki.fysik.dtu.dk/Niflheim_system/EasyBuild_modules/
- All our EB modules are served to the cluster by a single **dedicated fast NFS server** (SSD disks, 25 Gb/s Ethernet).

EasyBuild experiences

- We usually build modules with the FOSS and INTEL toolchains, trying to keep up with new versions, but being conservative due to frequent bugs in the very latest versions.
- Our group contributes some EB modules of Open Source codes developed at our lab:
  - GPAW, ASE, and ASAP.
  - Our in-house EB expert is Prof. Jakob Schiøtz.
- Rebuilding EB modules is simple in principle, but could be disruptive during production.
- We build some local custom modules which have trivial version updates in the `.eb` files.
- We build many modules with "`eb –from-pr=NNNNN`" because PRs are only merged slowly and must await new EB releases.

Documentation, documentation, documentation!

- In order to manage a complex HPC cluster with numerous hardware and software components, we decided many years ago to **write documentation for every single item**!

- Our Wiki pages are written in the **reStructuredText** (**RST**) markup language and stored in a GitHub repo. The *Sphinx* documentation engine generates HTML pages for the internet.

- Key management principle: **Everything must be documented**!
  Hardware (servers, storage, network), firmware configurations, OS installation with Kickstart or Ansible, the software stack (EasyBuild), and Slurm.
- These Wiki pages serve as our **collective memory** (and perhaps also as disaster recovery documentation) of the HPC cluster setup.

- We share (most of) our pages on the Internet via Wikis in https://wiki.fysik.dtu.dk/

Slurm documentation and tools

- **Slurm** by *SchedMD* is a crucial batch job system software for most HPC clusters.

- Since we first deployed Slurm in 2017, **everything we did with Slurm has been documented** in the Wiki pages at https://wiki.fysik.dtu.dk/Niflheim_system/
  - These Wiki pages may possibly have become a **significant Slurm resource** in addition to the documentation provided by *SchedMD* itself.

- Once your HPC cluster is up and running Slurm, many questions come up:
  - What are users' jobs and the compute nodes actually doing? Etc. Etc.
  - Answering such questions using only the provided Slurm tools can be non-trivial!

- To satisfy the needs for insights, we developed a **large number of add-on Slurm tools** available from https://github.com/OleHolmNielsen/Slurm_tools.

Conclusions

- **Two people** can handily install and operate a medium-sized HPC cluster (~700 nodes), where the research groups on-board their own new HPC users.
- **Cost efficiency** of a local HPC cluster is excellent.
- **EasyBuild** is the **crucial foundation** under the software stack which we provide.
- **Documentation of everything** is extremely important for managing the HPC cluster.
- Our **Wiki site** documents many details regarding **Slurm** installation and upgrading, configuration of services, and procedures for daily operations.
  We provide some **Slurm Tools** on GitHub that may significantly assist the management of Slurm clusters.