A brief introduction to using Kebnekaise at HPC2N

Birgitte Brydsø

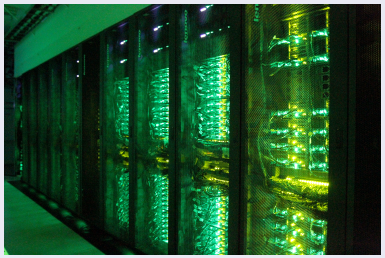
HPC2N, Umeå University

7-8 April 2022

Overview

- Kebnekaise
- Connecting to HPC2N's systems (Kebnekaise) with ThinLinc
- Editors
- The filesystem
- The module environment
- Example: NAMD
- Executing a job on Kebnekaise
  - SLURM
  - Jobscript examples
- Important information

Kebnekaise



1. 602 nodes / 19288 cores (of which 2448 are KNL)
   - 432 Intel Xeon E5-2690v4, 2x14 cores, 128 GB/node
   - 52 Intel Xeon Gold 6132, 2x14 cores, 192 GB/node
   - 20 Intel Xeon E7-8860v4, 4x18 cores, 3072 GB/node
   - 32 Intel Xeon E5-2690v4, 2x NVidia K80, 2x14, 2x4992, 128 GB/node
   - 4 Intel Xeon E5-2690v4, 4x NVidia K80, 2x14, 4x4992, 128 GB/node
   - 10 Intel Xeon Gold 6132, 2x NVidia V100, 2x14, 2x5120, 192 GB/node
   - 36 Intel Xeon Phi 7250, 68 cores, 192 GB/node, 16 GB MCDRAM/node
2. 501760 CUDA "cores" (80*4992 cores/K80+20*5120 cores/V100)
3. More than 136 TB memory total
4. Interconnect: Mellanox FDR / EDR Infiniband
5. Theoretical performance/HP Linpack: 984 TF / 791 TF
6. Date installed: Fall 2016 / Spring 2017 / Spring 2018

ThinLinc is a cross-platform remote desktop server developed by Cendio AB. It is especially useful for software with a graphical interface.

- Download the client and install it: `https://www.cendio.com/thinlinc/download`.
- Start the client. Enter the name of the server: **kebnekaise-tl.hpc2n.umu.se**. Enter your HPC2N username.
- (First time only) Go to "Options" $->$ "Security". Check that authentication is set to password.
- (First time only) Go to "Options" $->$ "Screen". Uncheck "Full screen mode".
- Enter your HPC2N password. Click "Connect"
- Click "Continue" when told that the server's host key is not in the registry. Wait for the ThinLinc desktop to open.

Editing your files

- Various editors: vi, vim, nano, emacs ...
- Example, vi/vim:
  - `vi <filename>`
  - Insert before: i
  - Save and exit vi/vim: `Esc :wq`
- **Example, nano:**
  - `nano <filename>`
  - Save and exit nano: `Ctrl-x`
- Example, Emacs:
  - Start with: emacs
  - Open (or create) file: `Ctrl-x Ctrl-f`
  - Save: `Ctrl-x Ctrl-s`
  - Exit Emacs: `Ctrl-x Ctrl-c`

More info here: http://www.hpc2n.umu.se/filesystems/overview

	Project storage	**$HOME (25GB)**	**/scratch**
Recommended for batch jobs	Yes	(No, size)	Yes
Backed up	No	Yes	No
Accessible by batch system	Yes	Yes	Yes (node only)
Performance	High	High	Medium
Default readability	Group only	Owner	Owner
Permissions management	chmod, chgrp, ACL	chmod, chgrp, ACL	N/A for batch jobs
Notes	Storage your group get allocated through the storage projects	Your home-directory	Per node

The filesystem, example

The course project has default storage here:
/proj/nobackup/snic2022-22-237

You should create a sub-directory under this directory, for your storing own files for this course, and for running the exercises:

- `cd /proj/nobackup/snic2022-22-237`
- `mkdir <username>` (or whatever you want to call your directory)

This storage will only be available for a few weeks (until around 1 May 2022).

The Module Environment

Most programs are accessed by first loading them as a 'module'

Modules are:

- used to set up your environment (paths to executables, libraries, etc.) for a particular (set of) software package(s)
- for helping users manage their Unix/Linux shell environment, allowing groups of related environment-variable settings to be made or removed dynamically
- allows having multiple versions of a program or package available by just loading the proper module
- installed in a hierarchial layout; thus some modules are only available after loading a specific compiler and/or MPI version

Compiler toolchains load software-bundles for a complete environment (compiling, using prebuilt software). Includes some/all of: compiler suite, MPI, BLAS, LAPACK, ScaLapack, FFTW, CUDA.

- **foss**: GCC, OpenMPI, OpenBLAS/LAPACK, FFTW, ScaLAPACK
- **intel**: icc, ifort, IntelMPI, IntelMKL

Finding software modules to load

- Listing all available modules:
  `module spider / ml spider`
- Listing all versions of a software:
  `module spider <software> / ml spider <software>`
  - Example: NAMD

```
b-an01 [~]$ ml spider NAMD

----------------------------------------------------------------------------
  NAMD:
----------------------------------------------------------------------------
    Description:
      NAMD is a parallel molecular dynamics code designed for
      high-performance simulation of large biomolecular systems.

     Versions:
        NAMD/2.14-mpi
        NAMD/2.14-nompi

----------------------------------------------------------------------------
  For detailed information about a specific "NAMD" package (including how to loa
d the modules) use the module's full name.
  Note that names that have a trailing (E) are extensions provided by other modu
les.
  For example:

     $ module spider NAMD/2.14-nompi
----------------------------------------------------------------------------
```

Loading and unloading software modules

- Loading a module:
  - Most software modules have prerequisites. Use 'ml spider' on a specific version to see how it is loaded
  - `ml spider <software>/version`
  - Then load it and any prerequisites `module load <pre-requisite module> / ml <pre-requisite module> module load <module> / ml <module>`
  - You can also load all on one line, listing the prerequisites first (`ml <pre-requisite module> <module>`)
- Unloading a module:
  `module unload <module>`
- Unloading all modules:
  `module purge / ml purge`
- Seeing which modules you currently have loaded
  `module list / ml`

Loading software modules: example

Loading NAMD/2.14-mpi:

```
b-an01 [~]$ ml spider NAMD/2.14-mpi

----------------------------------------------------------------------------
  NAMD: NAMD/2.14-mpi
----------------------------------------------------------------------------
    Description:
      NAMD is a parallel molecular dynamics code designed for
      high-performance simulation of large biomolecular systems.


    You will need to load all module(s) on any one of the lines below before the
 "NAMD/2.14-mpi" module is available to load.

      GCC/10.3.0  OpenMPI/4.1.1
      GCC/9.3.0  OpenMPI/4.0.3

    Help:
      Description
      ===========
      NAMD is a parallel molecular dynamics code designed for high-performance s
imulation of
       large biomolecular systems.


      More information
      ================
       - Homepage: https://www.ks.uiuc.edu/Research/namd/




b-an01 [~]$ ml GCC/10.3.0  OpenMPI/4.1.1 NAMD/2.14-mpi
b-an01 [~]$
```

Loaded software modules: example, NAMD/2.14-mpi

```
b-an01 [~]$ ml

Currently Loaded Modules:
  1) snicenvironment (S)   9) libxml2/2.9.10     17) OpenMPI/4.1.1
  2) systemdefault   (S)  10) libpciaccess/0.16  18) OpenBLAS/0.3.15
  3) GCCcore/10.3.0       11) hwloc/2.4.1        19) FlexiBLAS/3.0.4
  4) zlib/1.2.11          12) OpenSSL/1.1        20) FFTW/3.3.9
  5) binutils/2.36.1      13) libevent/2.1.12    21) ScaLAPACK/2.1.0-fb
  6) GCC/10.3.0           14) UCX/1.10.0         22) Tcl/8.6.11
  7) numactl/2.0.14       15) libfabric/1.12.1   23) NAMD/2.14-mpi
  8) XZ/5.2.5             16) PMIx/3.2.3

  Where:
   S:  Module is Sticky, requires --force to unload or purge
```

The Batch System (SLURM)

- Large/long/parallel jobs must be run through the batch system
- SLURM is an Open Source job scheduler, which provides three key functions
  - Keeps track of available system resources
  - Enforces local system resource usage and job scheduling policies
  - Manages a job queue, distributing work across resources according to policies
- In order to run a batch job, you need to create and submit a SLURM submit file (also called a batch submit file, a batch script, or a job script).
- Guides and documentation at: http://www.hpc2n.umu.se/support

Useful commands to the Batch System (SLURM)

- Submit job: `sbatch <jobscript.sh>` (successful submission returns a job-id number)

```
b-an01 [~/store/bbrydsoe/course-md/0.NAMD]$ sbatch namd-cpu.sh
Submitted batch job 18666958
b-an01 [~/store/bbrydsoe/course-md/0.NAMD]$
```

- As default, output/errors are found in `slurm-<job-id>.out`
- Get list of all jobs: `squeue`
- Get list of only your jobs: `squeue -u <username>`

```
b-an01 [~/store/bbrydsoe/course-md/0.NAMD]$ squeue -u bbrydsoe
             JOBID PARTITION     NAME     USER ST       TIME  NODES NODELIST(REASON)
          18666960     batch namd-cpu bbrydsoe PD       0:00      1 (Resources)
          18666959     batch namd-cpu bbrydsoe  R       0:13      1 b-cn0337
b-an01 [~/store/bbrydsoe/course-md/0.NAMD]$
```

- Adding the flag `--start` to `squeue` gives the estimated job start time. This can change depending on other people's jobs.
- Check on a specific job: `scontrol show job <job id>`
- Delete a specific job: `scancel <job id>`
- Delete all your jobs: `scancel -u <username>`

SLURM batch script for a standard NAMD job (CPU)

```
#!/bin/bash
#SBATCH -A SNIC2022-22-237 #Project id
#SBATCH -J my-namd-job #Name of job
#SBATCH --time=00:10:00 #Jobtime (HH:MM:SS) Max:  168H
#SBATCH -N 1 #Number of nodes.
#SBATCH -n 28 #Number of processes.

ml purge < /dev/null 2>&1
ml GCC/10.3.0 OpenMPI/4.1.1
ml NAMD/2.14-mpi

mpirun -np 28 namd2 config-file.inp > output_cpu.dat
```

NOTE if you are using the reservation for this course, you **have** to add `#SBATCH --gres=gpu:k80:2` (since the reservation is only for GPU nodes)

SLURM batch script for a standard NAMD job (GPU)

```
#!/bin/bash
#Project id
#SBATCH -A SNIC2022-22-237 #SBATCH -J my-namd-job #Name of
job
#SBATCH --time=00:10:00 #Jobtime (HH:MM:SS) Max:  168H
#SBATCH -N 1 #Number of nodes.
#SBATCH -n 28 #Number of processes.
#SBATCH --gres=gpu:k80:2 #Asking for 2 K80 GPUs
#SBATCH --exclusive #Asking for exclusive access to the
node

ml purge < /dev/null 2>&1
ml GCC/9.3.0 CUDA/11.0.2 OpenMPI/4.0.3
ml NAMD/2.14-nompi

namd2 +p28 config-file.inp > output_gpu.dat
```

Various useful info

- A project has been set up for the workshop: `SNIC2022-22-237`
- Use it by adding this to your submit file:
  `#SBATCH -A SNIC2022-22-237`
- The project is ONLY valid during the course and a few weeks after.
- Default storage is included with the project. It can be found here:
  `/proj/nobackup/snic2022-22-237`
- There is a reservation. To use, add this to the submit file:
  THURSDAY
  `#SBATCH --reservation=namd-gpu-day1`
  FRIDAY
  `#SBATCH --reservation=namd-gpu-day2`
- The reservation is ONLY valid for the specific day of the course.