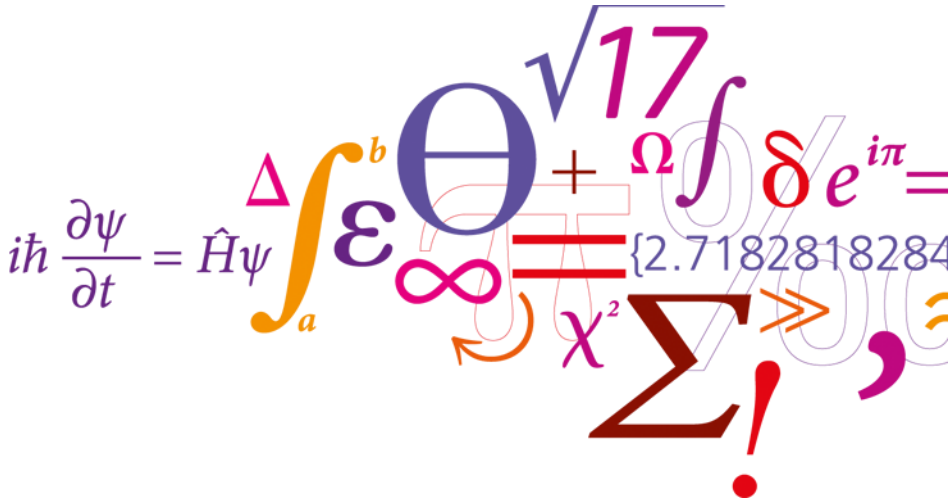
Slurm Wiki and Tools – a *Niflheim* site report

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/
Tools: https://github.com/OleHolmNielsen/Slurm_tools



DTU Fysik
Institut for Fysik

Slurm documentation and our Wiki

- *SchedMD* offers many excellent pages for *Slurm* administrators in the **Documentation pages**, and everyone is recommended to consult these pages for authoritative information.

- Our site started working with *Slurm* in 2015, but found the documentation back then scant and lacking both in overview and in the details.

- As a *Slurm* newbie, I started **writing down everything** I did to get *Slurm* up and running on our CentOS 7 based cluster, including **exact Linux commands** and configuration files.

- This became a public ***Slurm* Wiki**, now at https://wiki.fysik.dtu.dk/Niflheim_system/ (mirrored at *ReadTheDocs* https://niflheim-system.readthedocs.io/en/latest/)
  The Wiki has apparently been useful to other *Slurm* sites as well.

- A user-driven Wiki may significantly **ease the learning curve for deploying Slurm**.

Slurm support

- Our ***Slurm* support contract** with *SchedMD* has been important for improving the information in the Wiki, and conversely helped to improve a few of *Slurm's* documentation pages.

- The **excellent *Slurm* support from *SchedMD*** has helped us a lot in providing **trouble-free operations** and a high productivity of our cluster employing only a **very minimal IT staff**.

Slurm Wiki overview

- The Wiki discusses **selected *Slurm* features** needed by relatively simple small-to-medium sized clusters (10s to a few 1000s of nodes).

- **Exact Linux commands** for EL8 and EL9 (*RHEL*, *RockyLinux*, *AlmaLinux*, etc.) are given, in contrast to generic instructions which are valid for any Linux distribution, but have to be interpreted by the user to obtain specific OS commands.

- This Wiki is based on **actual user experiences** and requirements as we upgraded *Slurm* through **every major version** since 15.08, and started to exploit new *Slurm* features along the way.

- **Support cases** with *SchedMD*, as well as discussions on the **slurm-users** mailing list, are incorporated into the Wiki as reference information.

Wiki highlights: Installation and Upgrading

- **Wiki page**: https://wiki.fysik.dtu.dk/Niflheim_system/Slurm_installation/
- Setting up system accounts, how to install *Munge,* and other prerequisite RPM packages.
- Building and installing **RPMs** on EL8 and EL9 systems.
- *Slurm* **log file rotation**.
- **Upgrading major releases** should be done carefully:
  - *Slurm database* upgrade dry-run on a test node (lots of details provided).
  - Upgrade of `slurmctld` and possibly migrating it to a new server.
  - Upgrade of `slurmd` is usually straightforward, also on a running production system.

Wiki highlights: Configuration

- **Wiki page**: https://wiki.fysik.dtu.dk/Niflheim_system/Slurm_configuration/
- "***Configless Slurm***" testing.
- Configuring the *LBNL* **Node Health Check** (NHC) package.
- Building FreeIPMI **power control and monitoring** into *Slurm*.
- **Kernel configuration**: ARP cache, maximum number of open files.
- **PAM** configuration for pam_slurm_adopt SSH login restrictions.
- Temporary **job scratch directories** (`job_container/tmpfs` plugin).
- EL8/EL9 `firewalld` configuration.

Wiki highlights: Database

- **Wiki page**: https://wiki.fysik.dtu.dk/Niflheim_system/Slurm_database/
- **Initial setup** of *MariaDB*.
- Setting **database purge parameters** for old records.
- **Database backup** (using `logrotate`) and database restore.
- **Migration** of `slurmdbd` to a new server.

Wiki highlights: Slurm Operations

- **Wiki page**: https://wiki.fysik.dtu.dk/Niflheim_system/Slurm_operations/
- Configure *Slurm* **RPC rate limiting**.
- Expanding and collapsing **host lists**.
- **Passwordless SSH** configuration.
- **ClusterShell** parallel commands: Installation, configuration and usage.
- Compute node OS and firmware **rolling upgrades** during normal operations.

Wiki highlights: Power Saving Configuration

- **Wiki page**: https://wiki.fysik.dtu.dk/Niflheim_system/Slurm_cloud_bursting/
- Power saving can be used both for **on-premise nodes** as well as **cloud nodes**.
- Shutting down ("suspend") nodes for **saving electricity and cloud costs**.
- See also the previous presentations SLUG'22 Pathfinding into the clouds and SLUG'23 Saving Power with Slurm.
- Compute **nodes are now turned off and on dynamically** by `slurmctld` as required, enabling significant cost savings.

Wiki highlights: Power monitoring

- **Wiki page**: https://wiki.fysik.dtu.dk/Niflheim_system/Slurm_configuration/#power-monitoring-and-management

- After configuring FreeIPMI, you can enable **Power monitoring** in *slurm.conf* using `AcctGatherEnergyType=acct_gather_energy/ipmi`
  - A serious **IPMI-related bug in *slurmd*** was fixed in 23.02.7 (Bug 17639).

- Enable *IPMI Data Center Manageability Interface (**DCMI**)* power monitoring.
  - Problematic **non-DCMI-compliant BMCs** (e.g., *Xfusion*, *Huawei* – Bug 17704).

- When IPMI power monitoring has been enabled, it becomes possible (in principle, and for jobs using nodes exclusively) to make **energy accounting of individual jobs**.
  - Job energy accounting is still not fully reliable as of *Slurm* 23.11.10/24.05.3 due to a number of *slurmd* issues (bug 20207).

Slurm tools from the *Niflheim* GitHub site

- ***Niflheim* tools** are available from https://github.com/OleHolmNielsen/Slurm_tools
- These tools were developed over time when we needed to **monitor** the cluster operations, when we needed to **configure** *Slurm*, and when we needed to implement **additional functionality**.
- These tools mostly employ standard *Slurm* commands such as `sinfo`, `squeue`, `scontrol`, `sacctmgr`, etc.
- The most important tools are discussed in the following pages:

How many jobs and users are in the queue?

```
$ showuserjobs
Batch job status for cluster niflheim at Thu Aug 29 16:34:03 CEST 2024

Node states summary:
allocated     678 nodes ( 96.86%)  29360 CPUs ( 94.78%)
drained         3 nodes (  0.43%)    272 CPUs (  0.88%)
draining        4 nodes (  0.57%)    288 CPUs (  0.93%)
idle            7 nodes (  1.00%)    528 CPUs (  1.70%)
idle~           5 nodes (  0.71%)    288 CPUs (  0.93%) Powered off
mixed           2 nodes (  0.29%)    160 CPUs (  0.52%)
planned         1 nodes (  0.14%)     80 CPUs (  0.26%)
Total         700 nodes (100.00%)  30976 CPUs (100.00%)

Job summary: 3582 jobs total (max=20000) in all partitions.

Username/                 Runnin          Limit Pendin
Totals       Account        Jobs   CPUs    CPUs   Jobs   CPUs Further info
========== ========== ====== ====== ====== ====== ====== ==============================
GRAND_TOTAL ALL           667  29576    Inf   2915 119943 Running+Pending=149519 CPUs, 52 users
ACCT_TOTAL  camdvip       257  11192    Inf    884  37346 Running+Pending=48538 CPUs, 9 users
ACCT_TOTAL  catvip        153   7226    Inf    147   6248 Running+Pending=13474 CPUs, 6 users
ACCT_TOTAL  ecsvip        163   6934    Inf   1786  70285 Running+Pending=77219 CPUs, 18 users
manja       camdvip       107   4984   5000     74   4272
olich       catvip        117   4968   5000    143   5560
kartsh      camdvip        86   3632   5000    324  13248
```

User full names…

What is the status of Slurm partitions?

```
$ showpartitions
Partition statistics for cluster niflheim at Thu Aug 29 16:13:16 CEST 2024
           Partition     #Nodes     #CPU_cores  Cores_pending    Job_Nodes MaxJobTime Cores Mem/Node
           Name State Total   Idle   Total    Idle Resorc  Other    Min    Max  Day-hr:mn /node      (GB)
   xeon24el8:*    up   186      0    4464       0       0  11847      1 infin     2-02:00    24      256+
xeon24el8_test    up     2      2      48      48       0      0      1 infin          30    24       256
xeon24el8_week    up    20      0     480       0       0      0      1 infin     7-00:00    24       256
 xeon24el8_512    up    12      0     288       0       0      0      1 infin     2-02:00    24       512
     xeon40el8    up   320      4   12800     160       0  21720      1 infin     2-02:00    40       384
 xeon40el8_768    up    12      2     480      80       0   1640      1 infin     2-02:00    40       768
 xeon40el8_clx    up   128      2    5120      80       0      0      1 infin     2-02:00    40       384
        xeon56    up    96      0    5376       0       0  69384      1 infin     2-02:00    56       512
     sm3090el8    up     7      4     560     464       0    320      1 infin     7-00:00    80      191+
 sm3090el8_768    up     4      1     320     224       0      0      1 infin     7-00:00    80       768
  sm3090_devel    up     1      0      80      72       0      0      1 infin       12:00    80       191
   xeon32_week    up     2      0      64       0       0     32      1 infin     7-00:00    32      4096
   xeon32_4096    up     4      2     128      64       0      0      1 infin     2-02:00    32      4096
     a100_week    up     2      1     256     224       0      0      1 infin     7-00:00   128       512
          a100    up     4      3     512     480       0      0      1 infin     2-02:00   128       512
         epyc96    up    68      0    6528       0       0   6432      1 infin     2-02:00    96       768
Note: The cluster default partition name is indicated by :*
```

What are the nodes and jobs doing?

- The `pestat` command can give **many different kinds of node and job overviews**.
- The usual *Slurm* options (and more) can be used: -p, -u, -A, -q, -t, -w, -j, …
- For example:

```
$ pestat -F -p epyc96
Print only nodes that are flagged by * (RED nodes)
Print only nodes in partition epyc96
Hostname       Partition     Node Num_CPU  CPUload  Memsize  Freemem  Joblist
                            State Use/Tot  (15min)     (MB)     (MB)  JobID User ...
e005              epyc96   drain*   0  96    0.00   768000   769821
e006              epyc96   drain*   0  96    0.00   768000   769315
e009              epyc96    idle    0  96   96.24*  768000   671486
e018              epyc96   alloc   96  96  104.17*  768000   742671  7625705 yundi
e024              epyc96   alloc   96  96  103.66*  768000   446522  7630058 laumu
e027              epyc96   alloc   96  96  101.91*  768000   551690  7630058 laumu
e028              epyc96   alloc   96  96  102.26*  768000   751016  7625705 yundi
e064              epyc96   alloc   96  96  103.70*  768000   740925  7625706 yundi
e065              epyc96   alloc   96  96  102.28*  768000   750932  7625706 yundi
```

When will *draining* nodes become idle?

- The **`pestat`** command's "–E" (Job *EndTime*) option can be combined with the "-t draining" option.
- Sorting on the job *EndTime* (in column 11).
- For example:

```
$ pestat -E -t draining -C | sort -k 11
Force colors ON in output
Hostname       Partition     Node Num_CPU  CPUload  Memsize  Freemem  Joblist
Job EndTime is printed after each JobID/user
Select only nodes with state=draining
                           State Use/Tot  (15min)     (MB)     (MB)  JobID User EndTime ...
x192          xeon24el8*    drng*  24  24   24.31   256000   223429  7637111 olich 2024-08-31T01:48:46
a128           xeon40el8    drng*  40  40   40.15   384000   350799  7637214 olich 2024-08-31T15:31:25
e001              epyc96    drng*  96  96   97.65   768000   746859  7637913 s222468 2024-08-30T15:55:48
sd651          a100_week    drng*  32 128    1.72*  512000   502586  7630936 magstr 2024-09-04T09:10:57
```

What are the user processes in a job or a node?

```
$ psjob 7606946
JOBID      PARTITION NODES TASKS USER      ARRAY_JOB_ID ARRAY_TASK_ID START_TIME          TIME          TIME_LIMIT
7606946    xeon32_wee1     32    mhfga     7606946      N/A            2024-08-28T05:16:33 1-11:11:06    6-21:00:00
NODELIST: b021
---------------
b021
---------------
    PID NLWP S USER      STARTED     TIME %CPU   RSS COMMAND
 220255    1 S mhfga       Aug 28 00:00:00  0.0  3872 /bin/bash /var/spool/slurmd/job7606946/slurm_sc
 220346    1 S mhfga       Aug 28 00:00:00  0.0  3108 /bin/bash /home/energy/modules/software/QChem/6
 220359   95 S mhfga       Aug 28 41-08:57:08 2822 3190520496 /home/energy/modules/software/QChem/6.0
Total: 3 processes and 97 threads
Uptime: 16:27:39 up 10 days,  5:59,  0 users,  load average: 31.92, 32.26, 32.45
```

```
$ psnode s006
Node s006 information:
NODELIST   PARTITION  CPUS CPU_LOAD  S:C:T    MEMORY  STATE       REASON
s006       sm3090el8  80   2.00      4:10:2   768000  mixed       none
s006       sm3090el8_ 80   2.00      4:10:2   768000  mixed       none
Jobid list: 7639340 7642783
Node s006 user processes:
    PID NLWP S USER      STARTED     TIME %CPU   RSS COMMAND
   3772    1 S magstr   10:01:13 00:00:00  0.0  3260 /bin/bash /var/spool/slurmd/job7639340/slurm_sc
   3777    4 R magstr   10:01:13 06:26:53 99.6 3082444 python fast_molvae/sample.py --nsample 50000
   7155    1 S magstr   13:31:47 00:00:00  0.0  3296 /bin/bash /var/spool/slurmd/job7642783/slurm_sc
   7343    6 R magstr   13:41:22 02:47:46 99.7 3378856 python -u /home/energy/magstr/git/FastJTNNpy3
Total: 4 processes and 12 threads
```

Notifying users about badly behaving jobs

- Sending an **E-mail alert to users** when the *Slurm* administrator believes the job is using resources in an inefficient or incorrect manner:

```
$ notifybadjob 7642902

Please select one of the following reasons why you want to Notify about this job:
  1. Your job is doing no useful work and is essentially dead.
  2. Your job has grossly exceeded the available physical RAM memory and is very inefficient.
  3. Your job has grossly exceeded the physical RAM memory available per CPU core.
  4. Your job is running too many processes/threads and is overloading the CPU(s).
  5. Your job is using more CPU cores than your job has requested.
  6. Your job is not using all of the CPU cores or GPUs that you have requested.
  7. Your job is not using all of the GPUs that you have requested.
```

Job submit plugin for checking job sanity

- The ***Slurm job_submit* plugin** is a very useful feature for making sure that user jobs are checked for sanity (e.g., that numbers of CPU cores and/or GPUs requested correspond to the requested node hardware).
- Only trivial *job_submit* plugins can be found in the *Slurm* documentation or by searching the internet ☹

- We provide an **example *job_submit* Lua plugin** with a number of configurable checks that can be customized according to the **site's job policies**: https://github.com/OleHolmNielsen/Slurm_tools/tree/master/plugins
  - This `job_submit.lua` plugin has been extremely useful for **catching user mistakes** when submitting to the wrong partitions, or request an incorrect number of CPUs/GPUs, etc.

Managing accounts when adding/removing users

- When system *passwd* and *group* databases change, how do we **synchronize** this with the *Slurm* **accounts**?
  - We propose to use the already existing UNIX *passwd* and *group* information to define a mapping onto the *Slurm* account tree hierarchy. See details in SLUG'19 Slurm Account Synchronization with UNIX Groups and Users

- The `slurmusersettings` tool can **create, update or delete users** in the *Slurm* database based on the system *passwd* database. It can be used to **set or update user limits**.

- The `showuserlimits` tool **displays user limits** by parsing the highly convoluted output from the `scontrol show assoc_mgr` command.

- The `showjobreasons` tool shows a **summary of reasons** for jobs being in the *Pending* state.

Slurm accounting

- The `slurmreportmonth` tool conveniently generates **monthly, weekly, and yearly accounting statistics** from *Slurm* using the `sreport` command.

- The `slurmacct` and `topreports` tools (which use `sacct`) have some **advantages** over the `sreport` command:

  – **Partition specific accounting** is possible.

  – **Average CPU count** (job parallelism) is printed.

  – **Average waiting time** in the queue is printed (answer to "*My jobs wait for too long*").

  – **Users' full name** is printed (useful to managers).

Conclusions

- **SchedMD** offers a lot of excellent *Slurm* documentation which should be consulted first.
- However, there still is a need for **exact Linux commands**, exact RPM package versions, and configuration file details for clusters that employ an "*Enterprise Linux"* family operating system (*RHEL, RockyLinux, AlmaLinux,* etc.).
- Our **Wiki site** offers many additional details regarding *Slurm* installation and upgrading, configuration of services, and procedures for daily operations.
- **Energy and cloud savings** are important and can be configured with *Slurm*.
- We provide ***Slurm* Tools** on *GitHub* that significantly improves the management of jobs, nodes, power consumption, *Slurm* accounts, users, and user limits.
- Some of the tools provide (possibly more useful) *Slurm* accounting reports.