SLURM
Simple Linux Utility for Resource Management

Pär Andersson

National Supercomputer Centre
Linköpings universitet

2009-05-28 / HEPiX Spring 2009

Resource Management?

Manage resources within a single cluster

- Nodes
    - state (up/down/idle/allocated . . .)
    - access
- Jobs
    - queued and running
    - start/stop jobs
    - job scheduling

Introduction

- Designed and built for scalability
  - BlueGene/P @ LLNL 147,456 processors
- Easy to install and manage
- Easy to use
- Very stable

Homepage and development

- Laurence Livermore National Lab https://computing.llnl.gov/linux/slurm/
- Well supported and actively developed
- slurm-dev@lists.llnl.gov
  - High SNR
  - Fast replies
  - Patches welcome

Source code

- GPL licensed
  - readable source code
  - modular design
- No public version control
- No road-map
  - Will be fixed

SLURM support

- HP - “XC Cluster”
- IBM - “IBM HPC Open Software Stack”
- Sun - “Sun HPC Software, Linux Edition”

Also packaged in Debian/Ubuntu as `slurm-llnl`

SLURM at NSC 2007

2007, we were buying a new 800+ node cluster.

- Was using Torque
  - Scalability issues with older 200 node cluster
  - Bugs
- Time to see if there were something better
- Found SLURM

SLURM at NSC 2009

cluster	nodes	SLURM version	Scheduler
neolith	805	1.3	Moab
bore	56	1.3	sched/backfill
gimle	84	1.3	Moab
vagn	6	1.3	Moab

Job scheduling

- Built-in
  - sched/builtin - FIFO
  - sched/backfill - FIFO+backfill
  - sched/gang - time slicing
- SLURM 1.3 needs an external scheduler for advanced job prioritization.
- External schedulers
  - sched/wiki - Maui
  - sched/wiki2 - Moab
    - Have caused many problems
    - Often bugs in Moab

Job scheduling

- Much improved in SLURM 2.0
  - Reservations
  - priority/multifactor
    - Age, Fair-share, Job size, Partition, QoS
  - Hierarchical Fair-Share
    - Accounts, sub accounts
    - Shares

What is a job?

- A job allocation is a set of resources (nodes/cores) available to a user for a specified time
- Programs started as “job steps”
- Batch scripts just a common special case

Runnig jobs

- srun - Run a job step, if necessary create allocation first
- salloc - Obtain allocation, run command (on current host), release allocation
- sbatch - Submit a batch script

srun

```
[paran@d2 ~]$ srun -N 2 hostname
n799
n798
```

salloc

```
[paran@d2 ~]$ salloc -N2
salloc: Granted job allocation 27
[paran@d2 ~]$ echo $SLURM_NODELIST
n[798-799]

[paran@d2 ~]$ srun hostname
n799
n798

[paran@d2 ~]$ exit
exit
salloc: Relinquishing job allocation 27
```

sbatch

```
[paran@d2 ˜]$ cat testjob.sh
#!/bin/sh
#SBATCH --nodes 2
echo “Script running on: $(hostname), allocation: ”\
“$SLURM_NODELIST”

[paran@d2 ˜]$ sbatch testjob.sh
sbatch: Submitted batch job 28

[paran@d2 ˜]$ cat slurm-28.out
Script running on: n798, allocation: n[798-799]
```

SLURM daemons

- slurmctld
  - Central management daemon
  - Master/Slave
- slurmdbd (optional)
  - Accounting database system
- slurmd
  - On every compute node
- slurmstepd
  - Started by slurmd for every job step

Daemons

- One initscript reads config file and starts slurmctld, slurmd, neither or both
- Communication is authenticated using MUNGE or OpenSSL
- Hierarchial communication
    - Hard to debug

Configuration

- One unified configuration file `/etc/slurm/slurm.conf`
  - Always need to be synchronized on all nodes!
- scontrol command

Getting information

- Get the information you need
- In the format you like
- Without using `sed` and/or `awk` one-liners

Getting information

- Get the information you need
- In the format you like
- Without using `sed` and/or `awk` one-liners

sinfo example

Show jobid and allocated nodes for running jobs of the user paran:

```
$ squeue -t running -u paran -o "%i %u %D %N"
JOBID USER NODES NODELIST
510857 paran 2 n[771-772]
510856 paran 4 n[4,52,320,411]
```

sinfo example

Same, without header:

```
$ squeue -t running -u paran -o “%i %u %D %N” -h
510857 paran 2 n[771-772]
510856 paran 4 n[4,52,320,411]
```

sinfo example

Show all idle nodes in the partition “neolith”:

```
$ sinfo -t idle -p neolith -h -o %N
n[418,773-774,778,794]
```

hostlist

- Hostlist syntax is used everywhere
- Same as in pdsh and many other LLNL utilities
- `n[1-805]` is nicer than `n1,n2,n3,n4.......,n804,n805`

python-hostlist
http://www.nsc.liu.se/˜kent/python-hostlist/

Migration

- When?
    - New systems
- Think of the users!
    - Do they even care?
    - Non-issue if grid
- Batch scripts
    - sbatch parses `#PBS`-lines
- Torque/PBS wrappers available
    - qstat, qsub, pbsnodes etc

SLURM 2.0

Used to be 1.4-pre

- 2.0.0 released 2009-05-20
- Reservations
- Power control
- Improved slurmdbd accounting
- Fair-share
- Topology awareness

Grid

- SLURM backend for NorduGrid ARC
  - Currently used on `ce01.titan.uio.no` as part of the NDGF Tier1
- Creating backends for other middlewares should be easy

Extending SLURM

- Write a plug-in
- SPANK
  SLURM Plug-in Architecture for Node and job (K)control
- C API slurm.h
- Perl API
- Python API
  - Separate project, under development

Summary

- SLURM 1.3 is working well
- SLURM 2.0 looks interesting