SLURM Administrators Tutorial Hands-ON

20/01/15

Yiannis Georgiou

Resource Management Systems Architect

Installation

Exercise 1:SLURM Installation and initial basic configuration upon personal environment using multiple slurmd

1.Install MUNGE for authentication. Make sure the MUNGE daemon, munged is started before you start the SLURM daemons.
2.Install SLURM either creating a tgz from git or downloading an existing tgz
3.cd to the directory containing the SLURM source and type ./configure with the following options –prefix=/usr/local/ --enable-multiple-slurmd
4.Type *make* to compile SLURM.
5.Type *make install* to install the programs, documentation, libraries, header files, etc.
6.Create the slurm User upon all nodes of the cluster.
7.Create parent directories for SLURM's log files, process ID files, state save directories, etc. are not created by SLURM. They must be created and made writable by SlurmUser as needed prior to starting SLURM daemons.
8. Create a basic slurm.conf file with FIFO prioritization and scheduling and start the deamons

Some Administrator Tips

- You can install and run different versions of slurm upon the same compute nodes of your cluster simultaneously by running the deamons on different ports
  - This will allow to experiment with upstreaming versions before production
  - Or test the resolution of a bug in real scale

Partitions / Reconfiguration

2) Set two different partitions that have the same resources but one enables Exclusive allocation and the other allows sharing of nodes. Observe the logging In the particular files

3) Start slurm services in foreground and observe the outputs. Verify that everything Is set correctly and restart in the background

4) Drain a node without stopping the deamons.

Accounting/QOS/Limitations

Exercise 5: Activate accounting using slurmdbd and mysql
-configure 3 users with different limitations on maximum allowed jobs
-and 2 QOS with different priorities and walltimes

1.Usage of sacctmgr command as root
2.Create an account for each user with *sacctmgr create account*
3.Update accounts including the limitations on maximum allowed jobs with sacctmgr update account name=x set GrpJobs=y
4.Create a QOS with sacctmgr create qos

Some Administrator Tips

- In case your production cluster has a lot of user accounts then a script would be a good idea for the creation of all accounts along with their customized limitations
  - New development efforts with layouts framework (to appear on upcoming version 15.08) will ease this process

Scheduling

Exercise 6: Activate :
-backfill scheduling and consider high throughput workloads
-fairsharing with priority on smaller jobs
-preemption on the QOS level

Allocation-Placement

Exercise 7: Activate :
-network topology aware scheduling
-internal node topology with possibilities to deal with memory and cores as seperate resources
-CPU binding
-possibilities to reserve GPUs with isolation

Power Management

Exercise 8: Activate :
-power management in a way that when nodes are idle for more than 10min they are turned off
-node power monitoring
-job energy accounting
-job power profiling
-experiment with real MPI application

Job Submission related exercises

9) Associate user jobs to projects: A user may only charge jobs to one of his projects

Solution with SLURM

- Execution Permissions through accounts/users associations in DB
- Usage of QOS with particular partition through JobSubmit Plugin

10) Memory limit: User specifies per-process (=per-core) memory limit => compute and enforce per-node memory limit

Solution with SLURM

Srun parameters –mem=<MB>,--mem-per-cpu=<MB>
Slurm.conf paramaters DefMemPerCPU, DefMemPerNode, MaxMemPerNode, MaxMemPerCPU

Activation of jobacct_gather plugin and memory polling date viewed through sstat and sacct

11) Reject job submission if prerequisites are not met, e.g. quota full, wrong project selected

Solution with SLURM

- Quota full and permissions through accounts/users associations in DB
- More Advanced checking through JobSubmit Plugins

Job susbmission related exercises

12) Place job into "idle" queue if project's applied-for computing time is exhausted

Solution with SLURM

- The Idle queue could be configured as a QOS with very low priority and a jobsubmit plugin could be used to check the quota and place the job on low – QOS that may be preempted if an normal job with good quota needs the resources

13) Interactive jobs possible: use dedicated nodes

Solution with SLURM

- Configure a partition with Shared=Exclusive and direct interactive jobs on that partition

14) Job separation:
a)Restrict each job's processes to its own processor cores, e.g. using CPUsets
b) Enforce memory limits (per-node)

Solution with SLURM

- Activation of TaskPlugin=task/affinity with TaskPluginParameters=Cpusets or TaskPlugin=task/cgroup in slurm.conf
- Usage of Slurm.conf paramaters DefMemPerNode, MaxMemPerNode,

15) Efficient memory-, core-, topology-aware (e.g., job within one switch) job placement

Solution with SLURM

- **In the slurm.conf** the **topology/tree** plugin may be activated by the admins to allow job placement according to network topology constraints
- **In the submission** commands the users may use the **--switches=<count>[@<max-time>]** for network-aware placement
- A lot of parameter for efficient memory/socket (internal node) topology aware parameters (see CPU Management chapter)

Job placement

16) Pin job's processes to processor cores
a) Define default placement strategy (pack to processor, strided, ...)
b) Allow user-selectable per-job placement strategy

Solution with SLURM

- A lot of parameters in slurm.conf for defaults and submission commands for user selection (see CPU Management chapter of this Tutorial)

17) Resource scheduling policy:
fill hosts first, use hosts first that have minimal needed memory available

Solution with SLURM

- Different possibilities for scheduling policies (see CPU Management guide and allocation strategies)
- Usage of features in slurm.conf and setting priorities in the paritions with nodes having minimal memory.

18) Placement strategy should use smallest number of nodes as possible and prefer to allocate whole nodes for parallel jobs

Solution with SLURM

- Default placement strategy in SLURM is based on best-fit block allocation algorithm to minimize fragmentation.

an atos company