bullX instruments for innovation



Table of Contents

Bull Extreme Computing

- Overview.
- Principal concepts.
- Architecture.
- Scheduler Policies.





Introduction

- Simple Linux Utility for Resource Manager.
- Merely grouping together several machines on a network is not enough to constitute a real cluster.
- Resource Management software is required to optimize the throughput within the cluster, according to specific scheduling policies.
- A resource manager is used to allocate resources, to find out the status of resources, and to collect task execution information.
- From this information, the scheduling policy can be applied.
- Bull Extreme Computing platforms use SLURM, an Open-Source, scalable resource manager.



Introduction

- Simple Linux Utility for Resource Manager.
- SLURM development has been a joint effort of:
 - Lawrence Livermore National Laboratory (LLNL).
 - SchedMD.
 - HP.
 - Bull.
 - Linux NetworX
 - Many other contributors.
- Development began in 2002.
- Current SLURM development staff:
 - Morris Jette (LLNL, Project leader).
 - Danny Auble (LLNL).
 - Don Lipari (LLNL).
- Bull has more than 10 contributors to the project.



Top500 Clusters with SLURM

- **Tianhe-1A:** designed by The National University of Defence Technology (NUDT) in China with 14,336 Intel CPUs and 7,168 NVDIA Tesla M2050 GPUs. The world's fastest super computer with a peak performance of 2.507 Petaflops (Nov '10).
- **Tera 100:** at CEA with 140,000 Intel Xeon 7500 processing cores, 300TB of central memory and a theoretical computing power of 1.25 Petaflops. Europe's most powerful supercomputer.
- **Dawn:** a BlueGene/P system at LLNL with 147,456 PowerPC 450 cores with a peak performance of 0.5 Petaflops.
- **EKA:** at Computational Research Laboratories, India with 14,240 Xeon processors and Infiniband interconnect.
- **MareNostrum:** Linux cluster at Barcelona Supercomputer Center with 10,240 PowerPC processors and a Myrinet switch.



Features and Functionalities

Features:

- Job scheduler for small and large HPC clusters.
- Open-Source.
- Highly scalable.
- Fault-tolerant.

Key functionalities:

- Exclusive and shared resource allocation of computes nodes with a time limit quantum.
- Provides a framework for starting, executing and monitoring jobs on a set of allocated nodes.
- Arbitrate contention for resources by managing a queue of pending work.



Plugins

Optional plugins for:

- Accounting.
- Differntes scheduling policies (backfill, gang, etc.).
- Power saving.
- Resouce limits by user.
- Multifactor job prioritization algorithms.
- Topology optimized resource selection.
- Advanced reservation.



SLURM key functions

- As a cluster resource manager, SLURM has three key functions:
 - Firstly, it allocates exclusive and/or non-exclusive access to resources (Compute Nodes) to users for some duration of time so they can perform work.
 - Secondly, it provides a framework for starting, executing, and monitoring work (normally a parallel job) on the set of allocated nodes.
 - Finally, it arbitrates conflicting requests for resources by managing a queue of pending work.
- Optional plug-ins can be used for accounting, advanced reservation, backfill scheduling, resource limits by user or account, and sophisticated multifactor job prioritization algorithms.





Principal Concepts

A general purpose plug-in mechanism.

Nodes

Compute resource.

Partitions

- Groups of nodes into logical sets.
- Could be overlapped sets.

Jobs

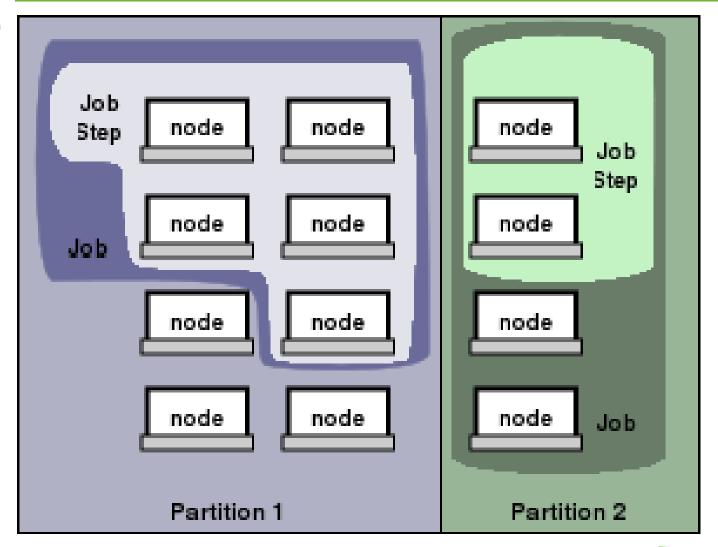
 Allocations of resources assigned to a user for a specified amount of time.

Job steps

- Sets of tasks within a Job.
- One queue of pending work.



Basic Concepts





©Bull, 2011

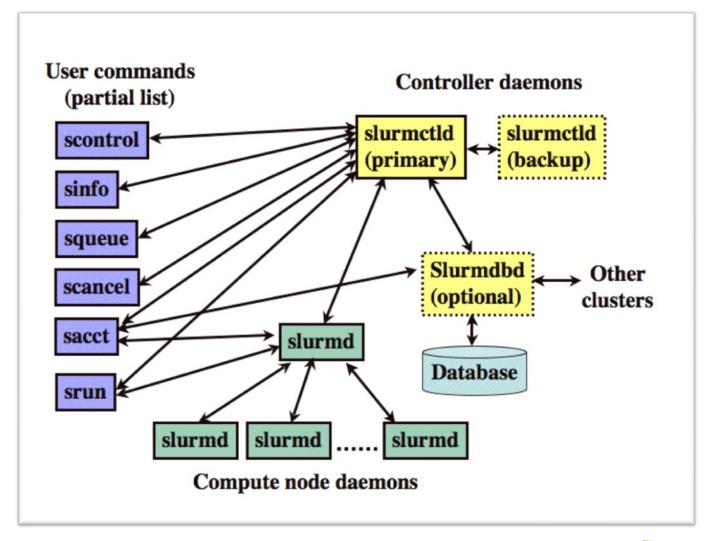
Internal-Node Topology aware Placement

- Two different plugins with specific functions:
 - Select/linear plugin which considers nodes to be one single resource for exclusive use.
 - Select/cons_res takes into account finer granularities like, socket, core and thread.
- The first one is more scalable, but the second one is indispensable for use upon SMP or NUMA architectures in order to avoid internal fragmentation.



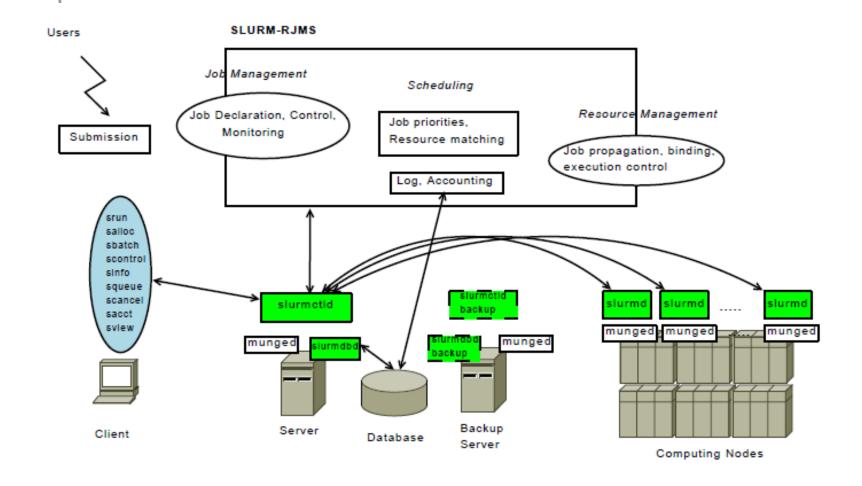


Architecture diagram





General Organization





SLURM Daemons and Commands

SLURM has the followings daemon process:

- SLURMCTLD.
- SLURMD.
- SLURMDBD.

SLURM CLI commands:

- SRUN to submit a job for execution.
- SCANCEL to terminate a pending or running job.
- SQUEUE to monitor job queues.
- SINFO to monitor partition and the overall system state.
- SACCTMGR to view and modify SLURM account information. Used with the slurmdbd Daemon.
- SACCT to display data for all jobs and job steps in the SLURM accounting log.



SLURM Daemons and Commands

SLURM CLI commands (cont.)

- SBATCH for submitting a batch script to SLURM.
- SALLOC for allocating resources for a SLURM job.
- SATTACH to attach to a running SLURM job step.
- STRIGGER used to set, get or clear SLURM event triggers.
- SVIEW used to display SLURM state information graphically. (Requires an Xwindows capable display).
- SREPORT used to generate reports from the SLURM accounting data when using an accounting database.
- SSTAT used to display various status information of a running job or step.



SLURMCTLD

- The central control daemon for SLURM is called SLURMCTLD.
- SLURMCTLD is multi-threaded; thus, some threads can handle problems without delaying services to normal jobs that are also running and need attention.
- SLURMCTLD runs on a single management node (with a fail-over spare copy elsewhere for safety), reads the SLURM configuration file, and maintains state information on:
 - Nodes (the basic compute resource).
 - Partitions (sets of nodes).
 - Jobs (or resource allocations to run jobs for a time period).
 - Job steps (parallel tasks within a job).



SLURMCTLD subsystems

Node Manager

- Monitors the state and configuration of each node in the cluster.
- It receives state-change messages from each Compute Node's SLURMD daemon asynchronously, and it also actively polls these daemons periodically for status reports.

Partition Manager

- Groups nodes into disjoint sets (partitions) and assigns job limits and access controls to each partition.
- The partition manager also allocates nodes to jobs (at the request of the Job Manager) based on job and partition properties.
- SCONTROL is the (privileged) user utility that can alter partition properties.



SLURMCTLD subsystems

Job Manager

- Accepts job requests (from SRUN/SBATCH or a metabatch system), places them in a priority-ordered queue, and reviews this queue periodically or when any state change might allow a new job to start.
- Resources are allocated to qualifying jobs and that information transfers to (SLURMD on) the relevant nodes so the job can execute.
- When all nodes assigned to a job report that their work is done, the Job Manager revises its records and reviews the pending-job queue again.



SLURMD

- The SLURMD daemon runs on all the Compute Nodes of each cluster that SLURM manages and performs the lowest level work of resource management.
- Like SLURMCTLD, SLURMD is multi-threaded for efficiency; but, unlike SLURMCTLD, it runs with root privileges (so it can initiate jobs on behalf of other users).
- SLURMD carries out five key tasks and has five corresponding subsystems.



SLURMD - subsystems

Machine Status

 Responds to SLURMCTLD requests for machine state information and sends asynchronous reports of state changes to help with queue control.

Job Status

 Responds to SLURMCTLD requests for job state information and sends asynchronous reports of state changes to help with queue control.

Remote execution

- Starts, monitors, and cleans up after a set of processes (usually shared by a parallel job), as decided by SLURMCTLD (or by direct user intervention).
- This can often involve many changes to processlimit, environmentvariable, working-directory, and user-id.



SLURMD - subsystems

Stream Copy Service

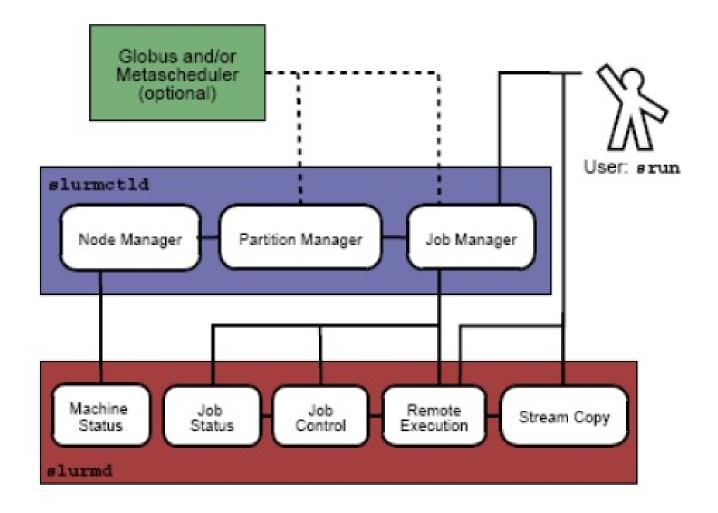
- Handles all STDERR, STDIN, and STDOUT for remote tasks.
- This may involve redirection, and it always involves locally buffering job output to avoid blocking local tasks.

Job Control

 Propagates signals and job-termination requests to any SLURM managed processes (often interacting with the Remote Execution subsystem).



SLURM - subsystems

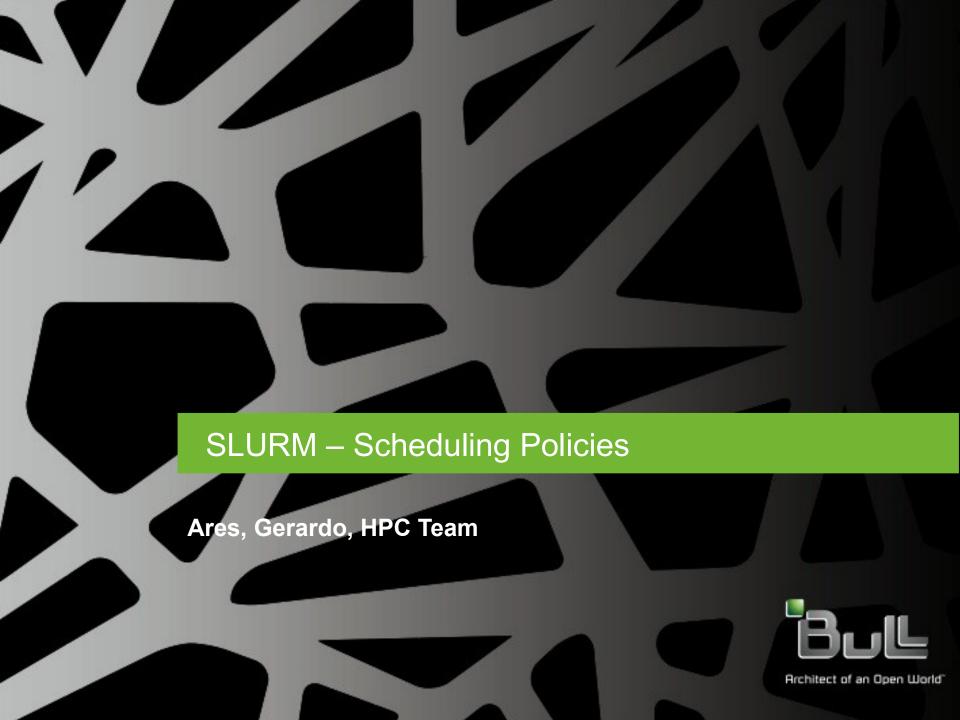




SLURMDBD

- The SlurmDBD daemon stores accounting data into a database.
- Storing the data directly into a database from SLURM may seem attractive, but requires the availability of user name and password data, not only for the SLURM control daemon (slurmctld), but also user commands which need to access the data (sacct, sreport, and sacctmgr).
- Making possibly sensitive information available to all users makes database security more difficult to provide, sending the data through an intermediate daemon can provide better security and performance (through caching data) and SlurmDBD provides such services.
- SlurmDBD is written in C, multi-threaded, secure and fast.





Scheduler Policies

Builtin (default)

- A first-in-first-out scheduler. SLURM executes jobs strictly in the order in which they were submitted (for each resource partition), unless those jobs have different priorities.
- Even if resources become available to start a specific job, SLURM will wait until there is no previously-submitted job pending (which sometimes confuses impatient job submitters).

Backfill

- Modifies strict FIFO scheduling to take advantage of resource islands that may appear as earlier jobs complete.
- SLURM will start jobs submitted later out of order when resources become available, and if doing so does not delay the execution time in place for any earlier-submitted job.



Scheduler Policies

Backfill (cont.)

- To increase the job's chances of benefiting from such backfill scheduling:
 - (1) Specify reasonable time limits (the default is the same time limit for all jobs in the partition, which may be too large), and
 - (2) Avoid requiring or excluding specific nodes by name.

Gang

 Multiple jobs may be allocated to the same resources and are alternately suspended/resumed letting only one of them at a time have dedicated use of those resources, for a predefined duration.



Scheduler Policies

Fairshare

 Take into account past executed jobs of each user and give priorities to users that have been less using the cluster.

Preemption

 Suspending one or more "low-priority" jobs to let a "high-priority" job run uninterrputed until it completes.



bullX instruments for innovation

