Slurm Overview

Brian Christiansen, Marshall Garey, Isaac Hartung
SchedMD

SC17

Outline

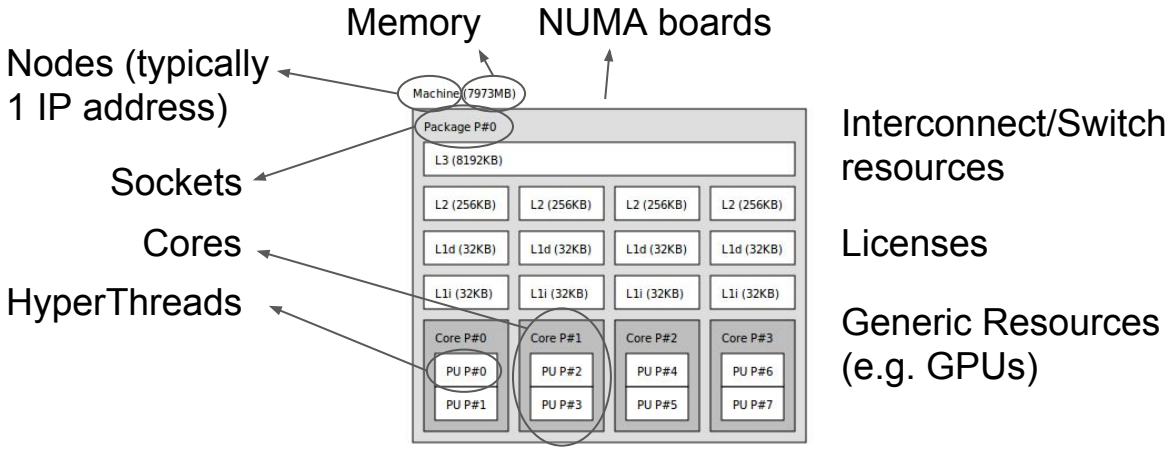
- Roles of a resource manager and job scheduler
- Slurm description and design goals
- Slurm architecture and plugins
- Slurm configuration files and commands
- Accounting

Outline

- Roles of a resource manager and job scheduler
- Slurm description and design goals
- Slurm architecture and plugins
- Slurm config files and commands
- Accounting

Roles of a Resource Manager

- Allocate resources within a cluster

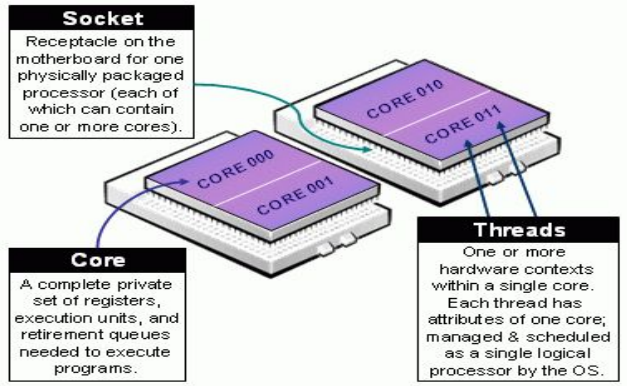


Interconnect/Switch resources

Licenses

Generic Resources (e.g. GPUs)

Can require extensive knowledge about the hardware and system software (e.g. to alter network routing or manage switch window)

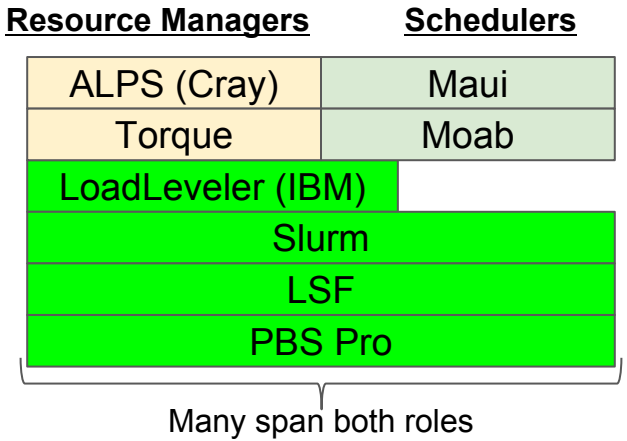


- Launch and otherwise manage jobs

Role of a Job Scheduler

- Prioritizes jobs based on policies
- Allocates time on resources
- Enforce resource limits
- Coordinates with Resource Manager

Examples



Slurm started as a resource manager (the “rm” in Slurm) and added scheduling logic later

Outline

- Roles of a resource manager and job scheduler
- Slurm description and design goals
- Slurm architecture and plugins
- Slurm config files and commands
- Accounting

What is Slurm?

- Historically Slurm was an acronym standing for
  - **S**imple **L**inux **U**tility for **R**esource **M**anagement
- Development started in 2002 at Lawrence Livermore National Laboratory as a resource manager for Linux clusters
- Sophisticated scheduling plugins added in 2008
- About 500,000 lines of C code today (plus test suite and doc)
- Used on many of the world's largest computers
- Active global user community

Slurm Design Goals

- Highly scalable (managing 3.1 million core Tianhe-2, tested to much larger systems using emulation)
- Open source (GPL version 2, available on Github)
- System administrator friendly
- Secure
- Fault-tolerant (no single point of failure)
- Portable - targeting POSIX2008.1 and C99

Slurm Portability

- *Autoconf* configuration engine adapts to environment
- Provides scheduling framework with general-purpose plugin mechanism. System administrator can extensively customize installation using a building- block approach
- Various system-specific plugins available
  - (e.g. *select/cray*)
- Huge range of use cases:
  - Sophisticated workload management at HPC sites
  - Scalable HTC environments (**14k jobs/minute sustained)**

Outline

- Roles of a resource manager and job scheduler
- Slurm description and design goals
- Slurm architecture and plugins
- Slurm config files and commands
- Accounting

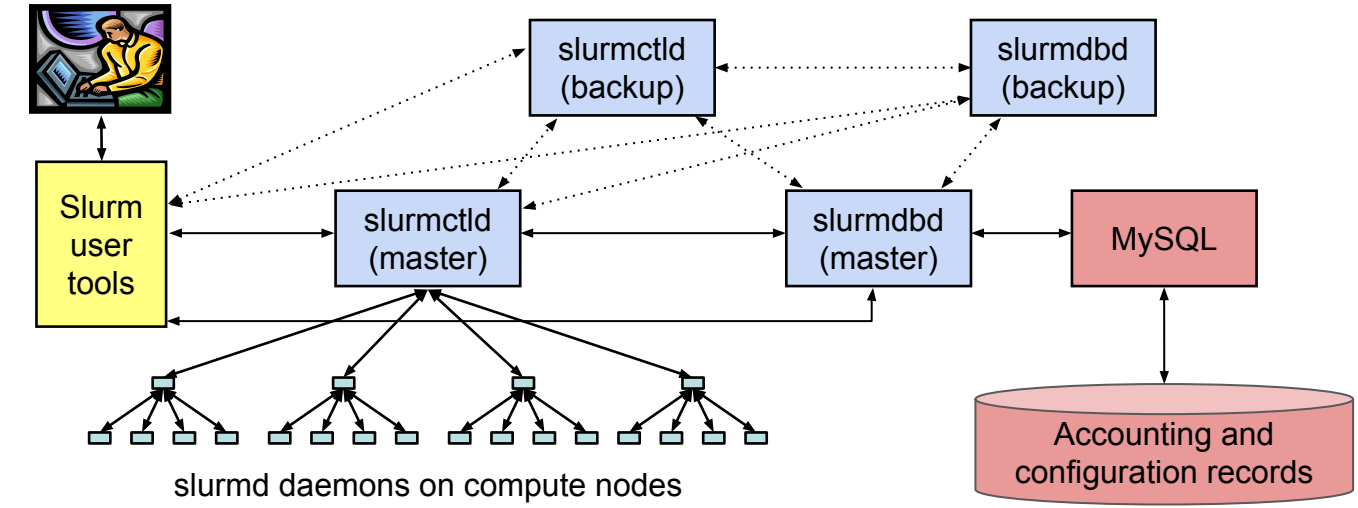
Daemons

- **slurmctld** – Central controller (typically one per cluster)
  - Monitors state of resources
  - Manages job queues
  - Allocates resources
- **slurmdbd** – Database daemon (typically one per enterprise)
  - Collects accounting information from controller(s)
  - Manages accounting configuration (e.g. limits, fair-share, etc.)
    - Pushes to controller(s)

Daemons

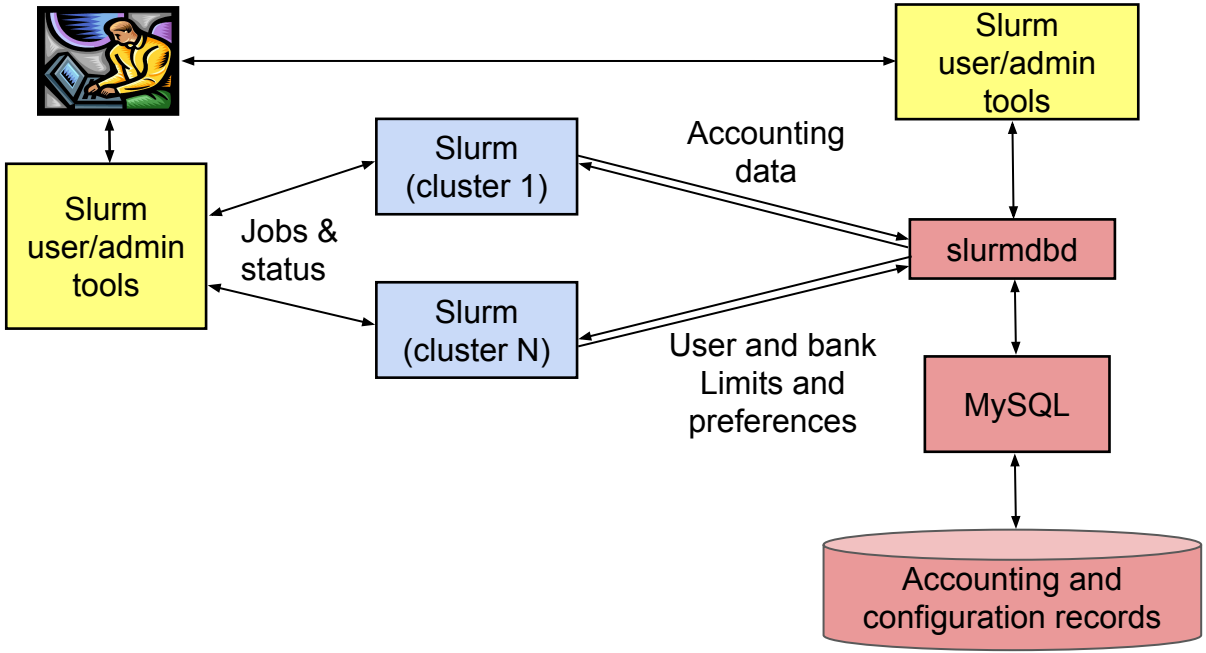
- **slurmd** – Compute node daemon (typically one per compute node)
  - Launches and manages slurmstepd (see below)
  - Small and very light-weight
  - Supports hierarchical communications with configurable fanout
- **slurmstepd** – Job step shepherd
  - Launched for batch job and each job step
  - Launches user application tasks
  - Manages accounting, application I/O, profiling, signals, etc.

Cluster Architecture



slurmd daemons on compute nodes
(Note hierarchical communications with configurable fanout)

Typical Enterprise Architecture



Job Queues (Slurm Partitions)

- Resource allocation requests (jobs) are placed in priority-ordered queues
- Resources (compute nodes) can be in one or more queues
- Dozens of limits available on a queue, both per-job and aggregate
- Jobs can be submitted to multiple queues at the same time

Job Steps

- A *job* can spawn one or more *job steps* within its allocation
  - Job steps can run sequentially or in parallel
  - Think of it as a job-specific resource management mechanism
  - Jobs spawning tens of thousands of job steps are common

Job Priority Factors

- Fair-share (how over- or under-served a user/group is)
- Age (how long queued)
- Size (favor larger or smaller jobs)
- Queue/partition priority factor
- Quality Of Service (QOS) priority factor

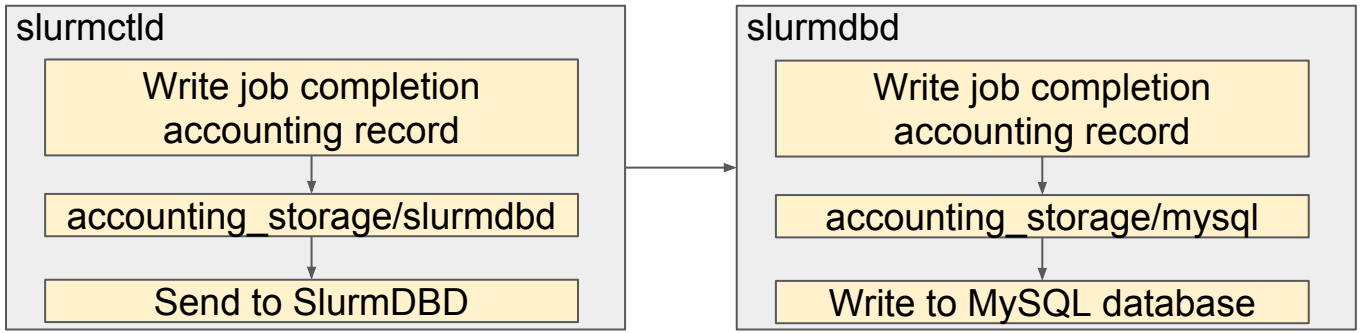
Plugins

- Dynamically linked objects loaded at run time based upon configuration file and/or user options
- 100+ plugins of 32 different varieties currently available
  - Network topology: 3D torus, tree, etc
  - MPI: OpenMPI, PMI2, PMIX
  - External sensors: Temperature, power consumption, etc.

Slurm Kernel (65% of code)				
Authentication Plugin	MPI Plugin	ProcTrack Plugin	Topology Plugin	Accounting Storage Plugin
Munge	pmi2	cgroup	Tree	SlurmDBD

Plugin Design

- Plugins typically loaded when the daemon or command starts and persist indefinitely
- Provide a level of indirection to a configurable underlying function



Plugin Development

- APIs are all documented for custom development
- Most APIs have several examples available
- Some plugins have a LUA script interface
  - Job submit plugin

Job Submit Plugin

- Called for each job submission or modification
- Can be used to set default values or enforce limits using functionality outside of Slurm proper

Two functions need to be supplied:

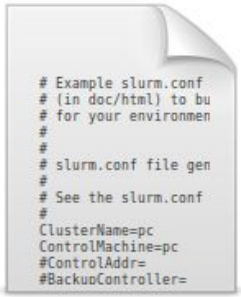
```
int job_submit(struct job_descriptor *job_desc, uint32_t submit_uid);
int job_modify(struct job_descriptor *job_desc, struct job_record *job_ptr);
```

Outline

- Roles of a resource manager and job scheduler
- Slurm description and design goals
- Slurm architecture and plugins
- Slurm config files and commands
- Accounting

Slurm Configuration

slurm.conf



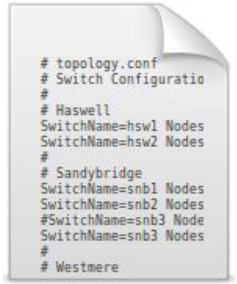
- General conf
- Plugin activation
- Sched params
- Node definition
- Partition conf

slurmdbd.conf

- Describes slurmdbd
- Archive/Purge parameters
- Storage options

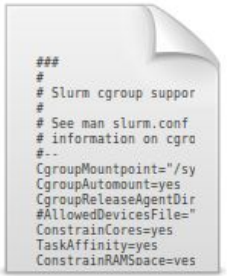
Slurm Configuration

topology.conf



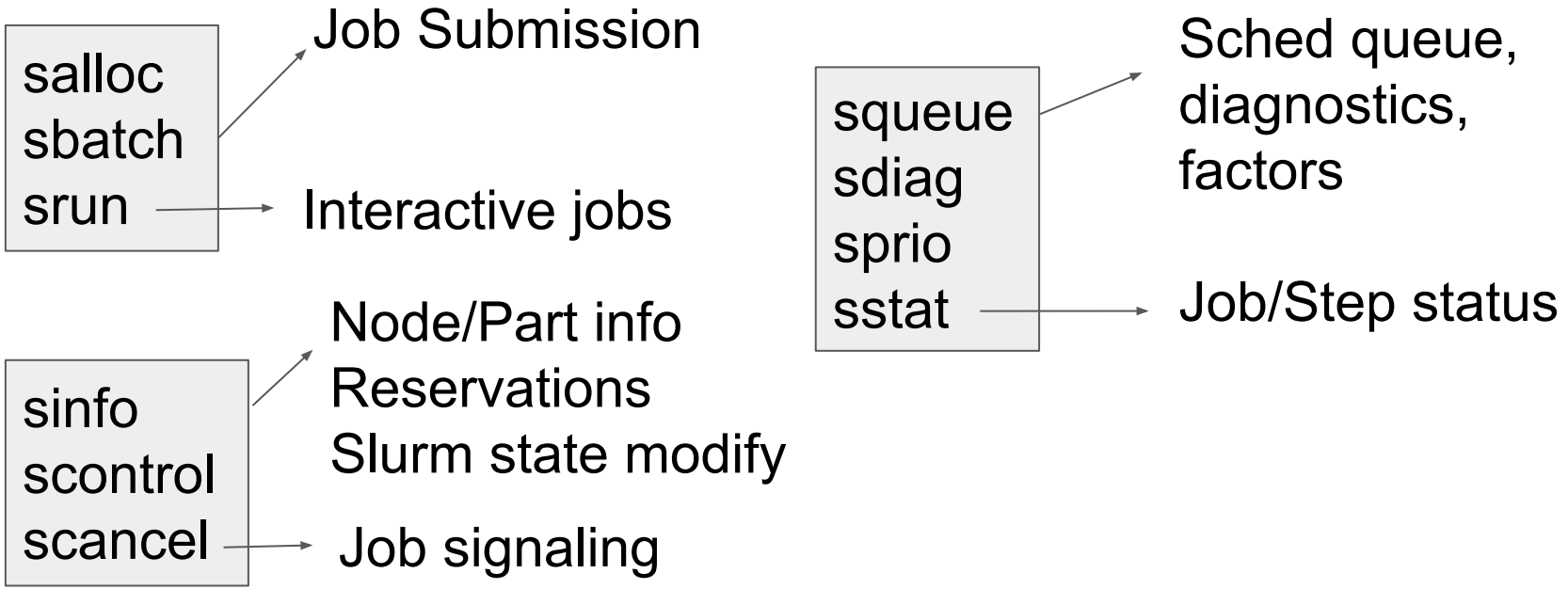
gres.conf

cgroup.conf



- Others: burst_buffer.conf, acct_gather.conf, knl.conf, etc.

Commands Overview



Commands Overview

sacct
sacctmgr
sshare
sreport

→ Accounting data
view/modify
FairShare info
Report generation

sattach
sbcast
strigger

→ I/O attach to jobs,
file transmission
to nodes, events
triggering

sview
smap

→ Graphical
interfaces

- --help, --usage
- man pages
- APIs make new tool development easier

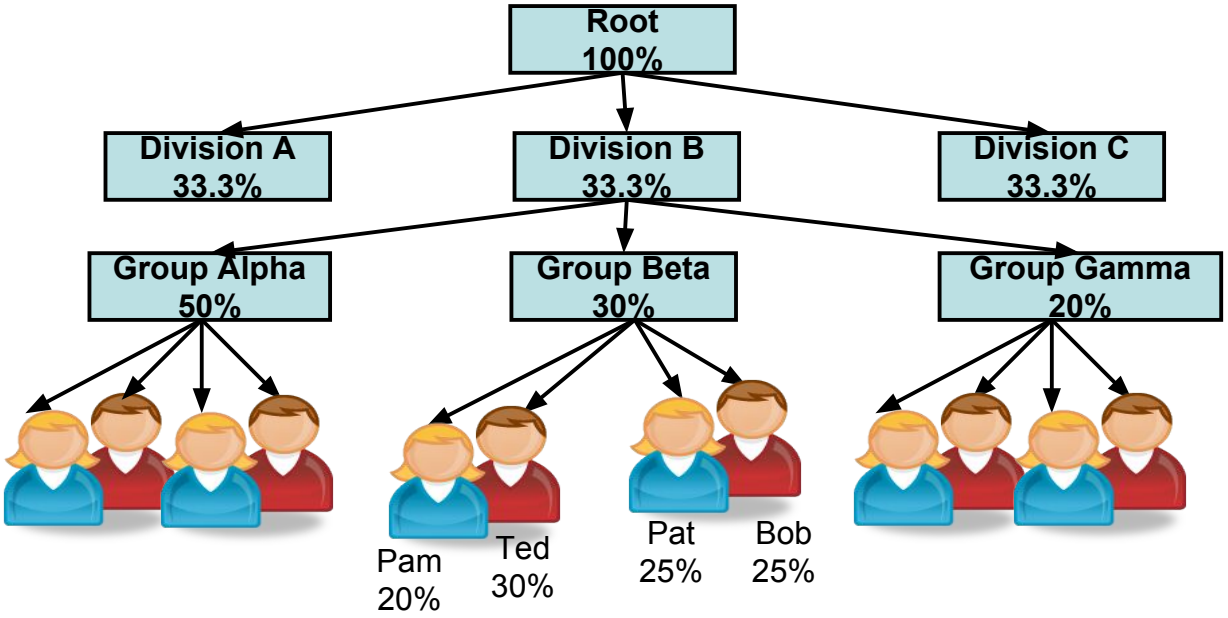
Outline

- Roles of a resource manager and job scheduler
- Slurm description and design goals
- Slurm architecture and plugins
- Slurm config files and commands
- Accounting

Database Use

- Job accounting information
- Quality of Service (QOS) definitions
- Fair-share resource allocations
- Configuring limits (max job count, max job size, etc.)
  - Per Job limits (e.g. MaxNodes)
  - Aggregate limits by user, account or QOS (e.g. GrpJobs)
- Based upon hierarchical accounts
  - Limits by user AND by accounts
- Information pushed out live to scheduler daemons

Hierarchical Account Example



And More ...

- Job dependencies
- Fine-grained task layout
- Wrappers for other workload manager commands
- Burst Buffers
- Job arrays
- KNL support
- PAM support
- cgroup support

Questions