

# Managing Clusters with Moab and SLURM

## May 2008



**Morris Jette (jette1@llnl.gov)**

**Donald Lipari (lipari1@llnl.gov)**

**S&T Principal Directorate - Computation Directorate**

# Disclaimer



This document was prepared as an account of work sponsored by an agency of the United States government. Neither the United States government nor Lawrence Livermore National Security, LLC, nor any of their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process discloses, or represents that its use would not infringe privately owned rights. References herein to any specific commercial product, process, or service by trade name, trademark, manufacture, or otherwise, does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States government or Lawrence Livermore National Security, LLC. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States government or Lawrence Livermore National Security, LLC, and shall not be used for advertising or product endorsement purposes.

This work was performed under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under Contract DE-AC52-07NA27344.



## Presentation Overview

- Description of SLURM
- How SLURM and Moab work together
- Workload Management at LLNL

## What is SLURM

- **S**imple **L**inux **U**tility for **R**esource **M**anagement, although it is no longer very simple at >230k lines of code and supports AIX and OS X in addition to Linux
- Performs resource management within a single cluster
- Arbitrates requests by managing queue of pending work
- Allocates access to computer nodes and their interconnect
- Launches parallel jobs and manages them (I/O, signals, time limits, etc.)
- Accounts for resource use by user and bank

## Key Features of SLURM

- Simple (relatively speaking)
  - Scheduling complexity largely external to SLURM (e.g. Moab)
- Open source: GPL
- Portable
  - C-language, *autoconf*, no kernel modifications required
- Fault-tolerant
  - For SLURM daemons and its jobs
- Secure
  - Authentication and cryptography plugins
- System administrator friendly
  - Simple and highly scalable configuration file
  - Supports heterogeneous clusters
- Scalable to the largest computers (64k nodes)

## Advanced SLURM Capabilities

- Highly modular design with flexible plugin mechanism
  - 51 different plugins of 12 different types (authentication mechanism, interconnect, mpi, etc.) available today
- Hierarchical communications with configurable fanout
  - Improves scalability with fault tolerance
  - Supports file tra

## Other Notable SLURM Features

- Widely deployed, running on about 1/3 of the Top500 systems
- Commercially distributed and supported HP, Bull, SiCortex and others
- Perl APIs and PBS/Torque command wrappers available for easy transition

## Scheduling Roles

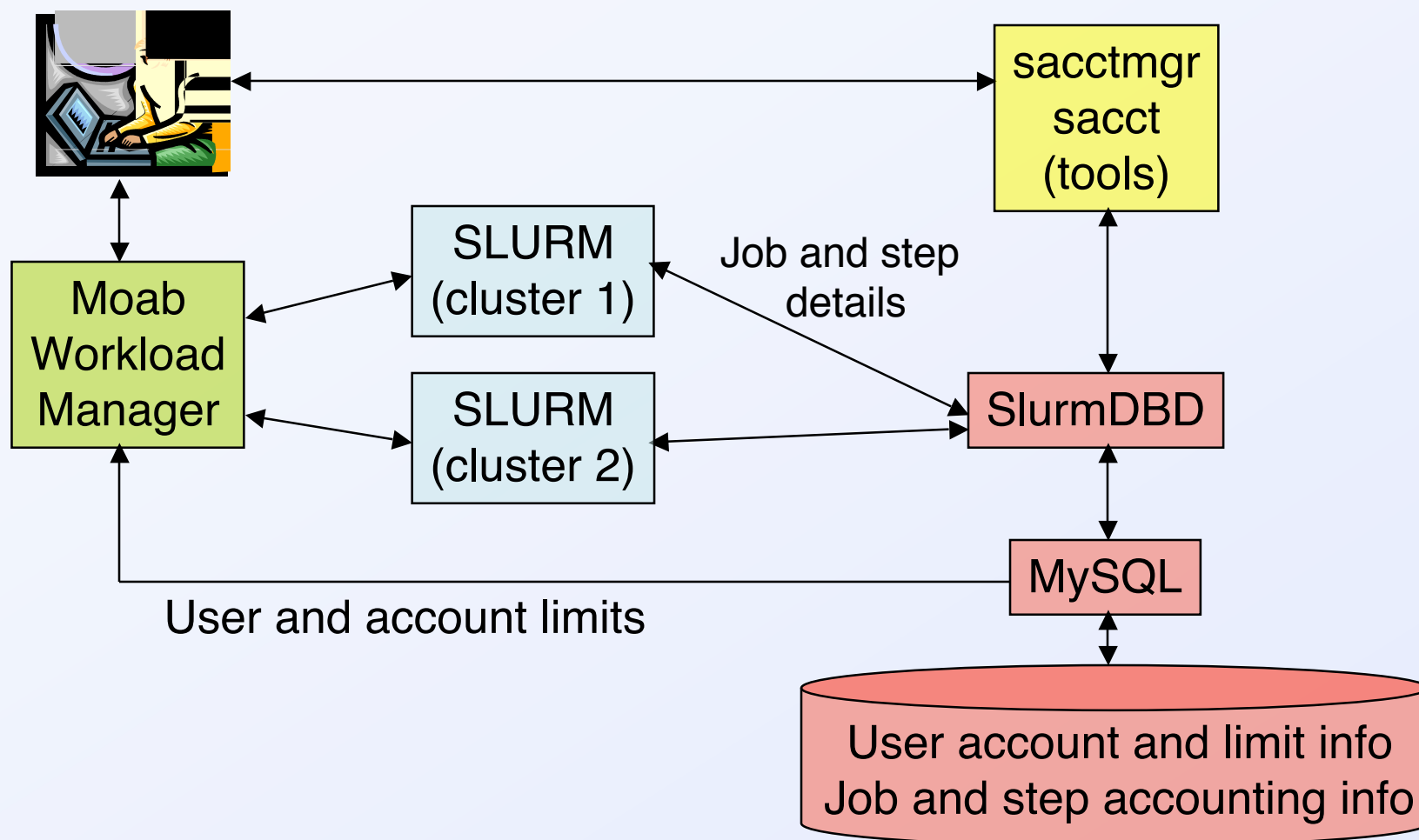
- Moab provides user resource limits, fair-share scheduling, and workload management across multiple clusters
- SLURM provides a highly configurable resource management infrastructure, most of which is well integrated with Moab
  - Job control over specific nodes to be required or excluded, node features, license requirements, etc.
  - Multiple queues with access control, priority value, job size and time limits, etc.
  - Manage the network, reboot nodes, etc.



## Communications between SLURM and Moab

- Users typically submit and manage jobs using Moab tools
  - Jobs can also be directly submitted to or managed using SLURM tools, but that provides a only cluster-centric perspective
- Moab submits jobs using a SLURM command (sbatch)
- Moab gathers job and node information using *wiki* interface (authenticated RPCs over socket using text messages)
  - SLURM commands are used for some system information
- Moab starts, stops, suspends, modifies and otherwise manages jobs using the *wiki* interface
- SLURM notifies Moab of state changes via event notification (job termination, node down, etc. by sending message over a socket)

# LLNL Workload Management Architecture



## Notes on LLNL Architectures

- Moab configured in the “grid mode” to manage workload on an enterprise-wide basis
- SLURM’s accounting information feeds into Moab’s fair-share scheduling mechanism
- Some clusters have whole nodes allocated to jobs (for best parallel job performance)
- Other clusters have individual processors (cores) allocated to jobs for non-parallel jobs
- Most clusters have a “debug” partition for small jobs and SLURM is configured to schedule these jobs independently, typically in <0.1 second, but without Moab or any of Moab’s scheduling policy support