SLURM

Resource Management at LLNL
SLURM Version 1.2

April 2007

Morris Jette (jette1@llnl.gov)
Danny Auble (auble1@llnl.gov)
Chris Morrone (morrone2@llnl.gov)
Lawrence Livermore National Laboratory

UCRL-PRES-230170

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 the University of California 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 the University of California. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or the University of California, and shall not be used for advertising or product endorsement purposes.

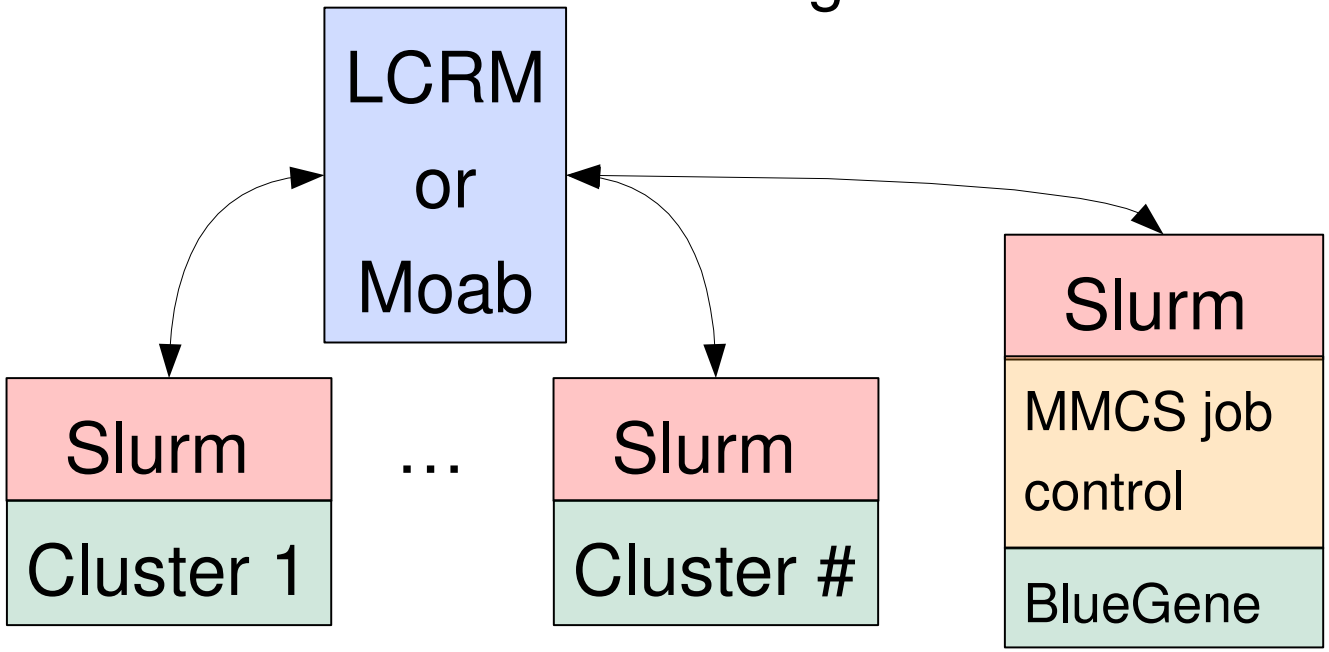
This work was performed under the auspices of the U.S. Department of Energy by the University of California, Lawrence Livremore National Laboratory under Contract No. W-7405-Eng-48.

What is SLURM's Role

- Performs resource management within a single cluster
- Arbitrates requests by managing queue of pending work
- Allocates access to resources (processors, memory, sockets, interconnect resources, etc)
  - Boots nodes and reconfigure network on BlueGene
- Launches and manages tasks (job steps) on most clusters
  - Forwards stdin/stdout/stderr
  - Enforce resource limits
  - Can bind tasks to specific sockets or cores
  - Can perform MPI to initialization (communicate host, socket and task details)
- Supports job accounting
- Supports file transfer mechanism (via hierarchical communications)

Job Scheduling at LLNL

- LCRM or Moab (http://www.clusterresources.com) provide highly flexible enterprise-wide job scheduling and reporting with a rich set of user tools
- Slurm (http://www.llnl.gov/linux/slurm) provides highly scalable and flexible resource management within each cluster

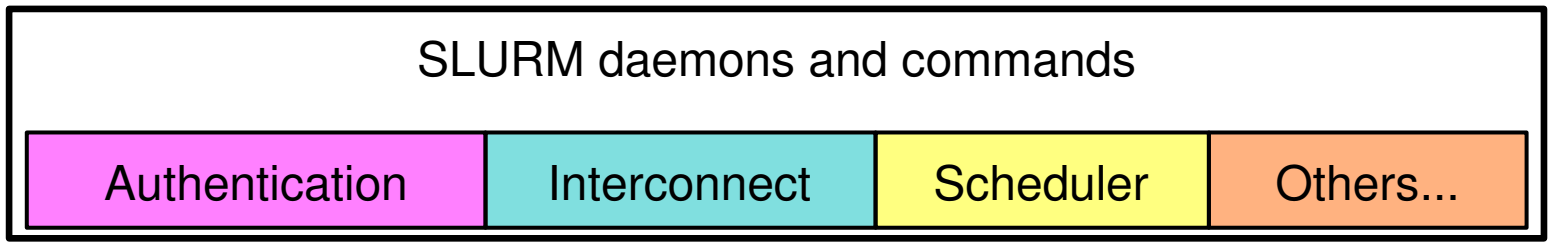


Both Moab and Slurm are production quality systems in widespread use on many of the worlds most powerful computers

SLURM Plugins

(A building-block approach to design)

- Dynamically linked objects loaded at run time per configuration file
- 34 different plugins of 10 different varieties
  - Interconnect
    - Quadrics Elan3/4, IBM Federation, BlueGene or none (for Infiniband, Myrinet and Ethernet)
  - Scheduler
    - Maui, Moab, FIFO or backfill
  - Authentication, Accounting, Logging, MPI type, etc.



SLURM's Scope

(It's not so simple any more)

- Over 200,000 lines of code
- Over 20,000 lines of documentation
- Over 40,000 lines of code in automated test suite
- Roughly 35% of the code developed outside of LLNL
  - HP
    - Added support for Myrinet, job accounting, consumable resource, and multi-core resource management
    - Working on gang-scheduling support
  - Bull
    - Added support for CPUsets
    - Working on expanded MPI support for MPICH2/MVAPICH2
  - Dozens of additional contributors at 15 sites world-wide

SLURM is Widely Deployed

- SLURM is production quality and widely deployed today (best guess is ~1000 clusters)
- ~7 downloads per day from LLNL and SourceForge
  - To over 500 distinct sites in 41 countries
- Directly distributed by HP, Bull and other vendors to many other sites
- No other resource manager comes close to SLURM's scalability and performance

SLURM

SLURM 1.0 Architecture on a Typical Linux Cluster

Compute Nodes

(Point-to-point communications)

Administration Nodes

slurmd

slurmd

slurmd

slurmctld
(primary)

slurmctld
(optional backup)

Flatfile

srun
(submit job or
spawn tasks)

sacct
(accounting)

sinfo
(system status)
squeue
(status jobs)
scancel
(signal jobs)
scontrol
(admin tool)
smap
(topology info)

SLURM

SLURM 1.2 Architecture on a Typical Linux Cluster

Compute Nodes

(Hierarchical communications)

slurmd

slurmd

slurmd

Administration Nodes

slurmctld
(optional backup)

slurmctld
(primary)

Flatfile

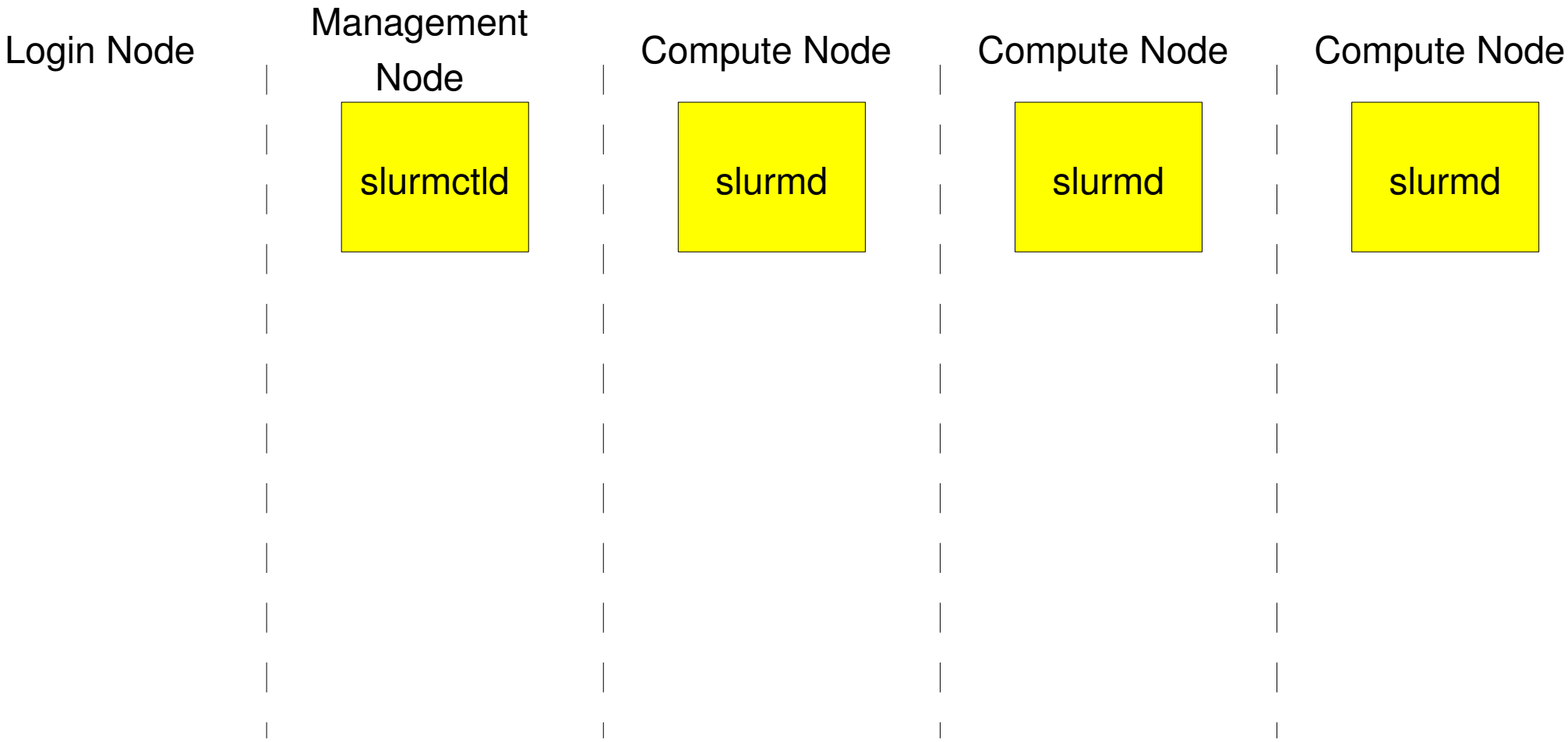
sacct
(accounting)

sbcast
(file broadcast)

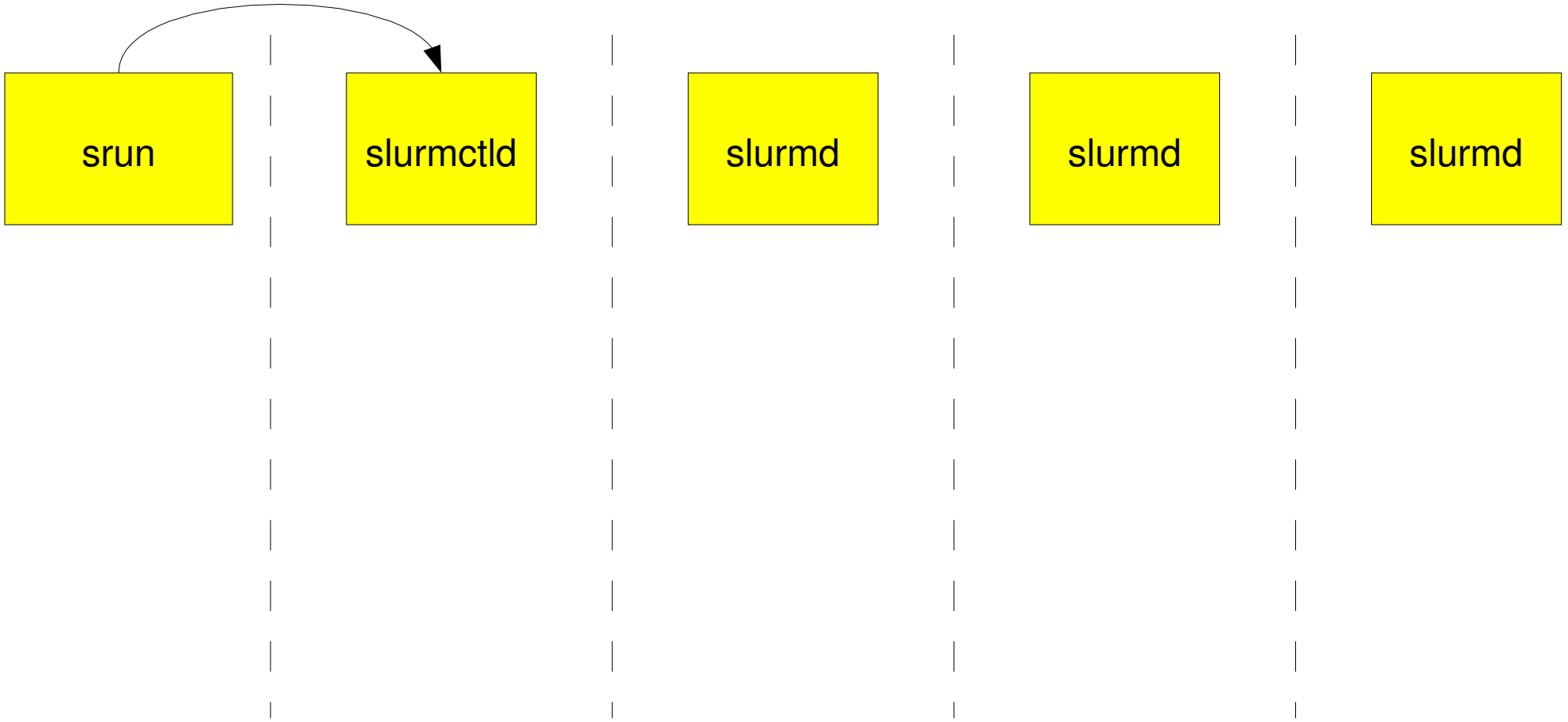
salloc
(get an allocation)
sbatch
(submit a batch job)
sattach
(attach to an existing job)
srun
(submit job or spawn tasks)

sinfo
(system status)
squeue
(status jobs)
scancel
(signal jobs)
scontrol
(admin tool)
smap
(topology info)
sview
(GUI for system information)
strigger
(event manager)

SLURM on Linux – launch

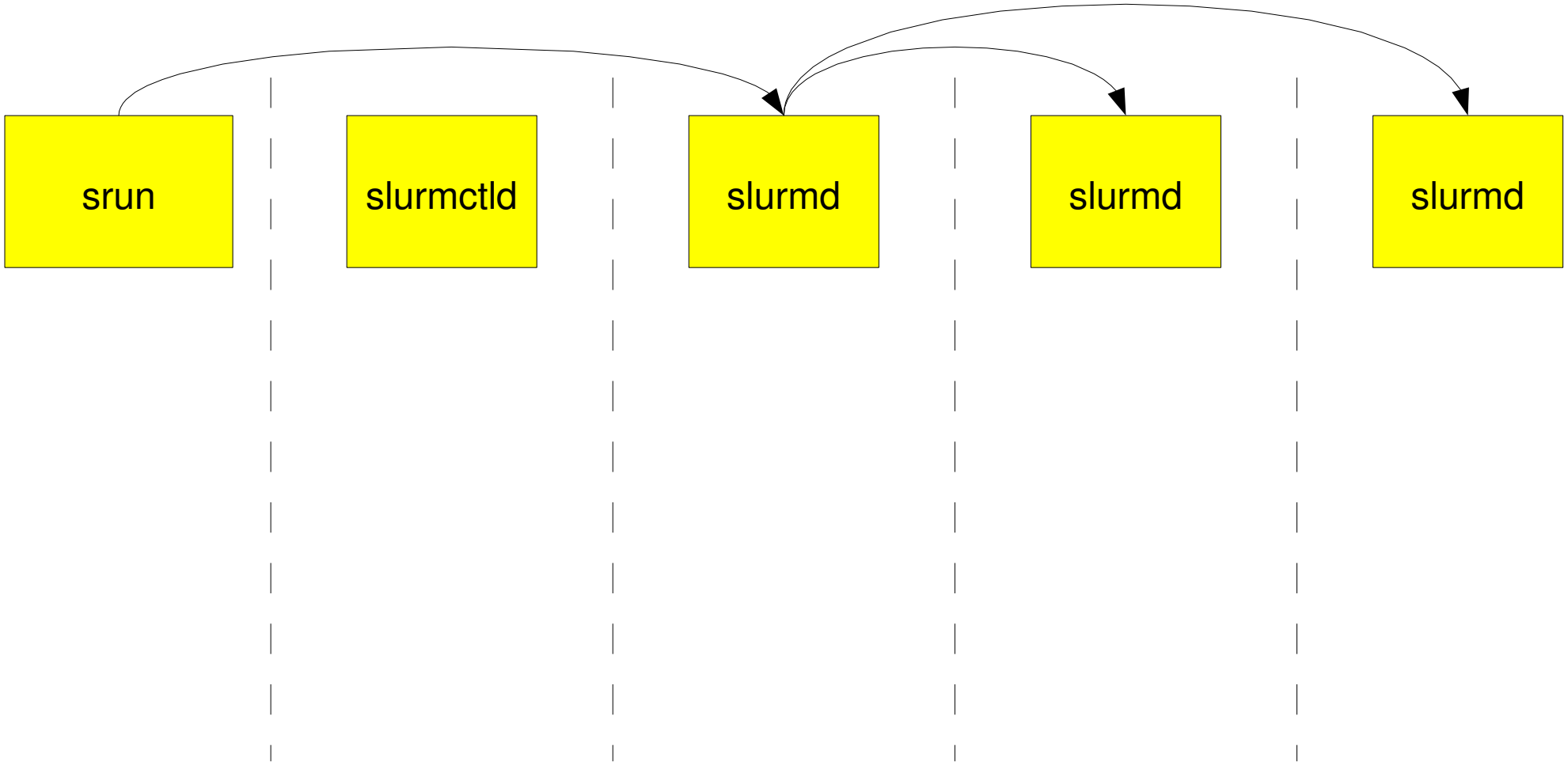


SLURM on Linux – launch



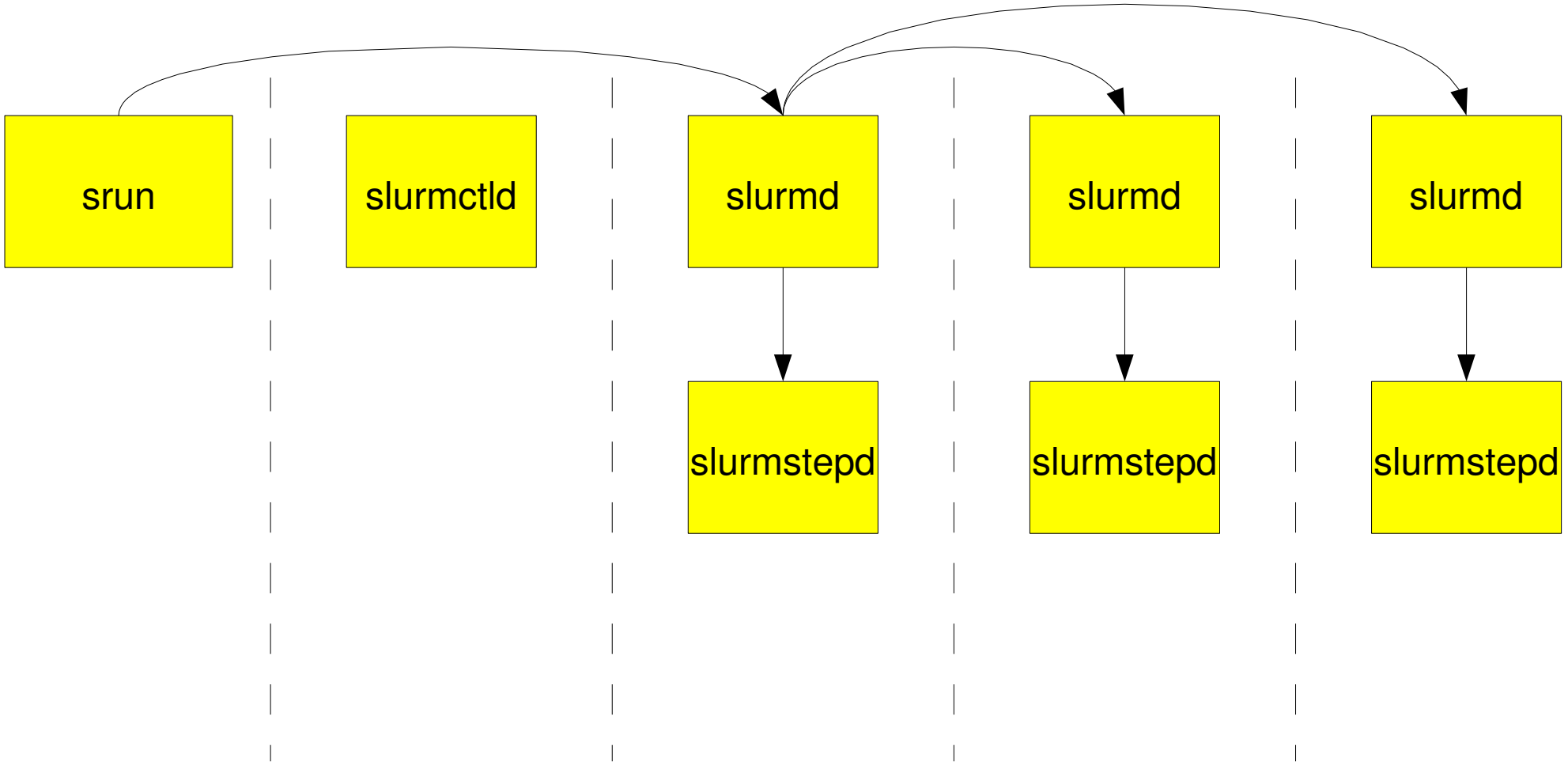
“srun -N3 -n6 -ppdebug mpiprog”

SLURM on Linux – launch



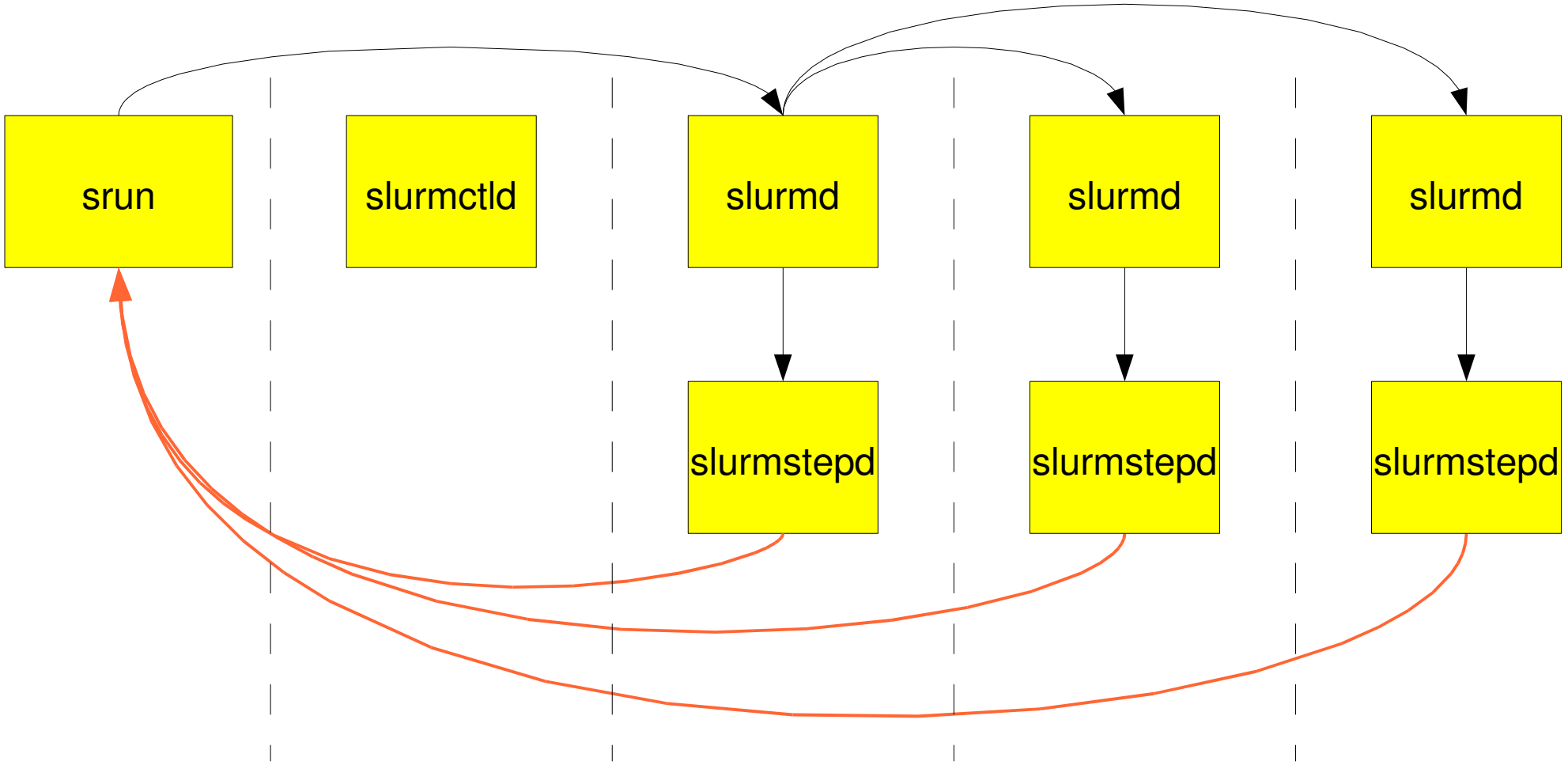
“srun -N3 -n6 -ppdebug mpiprog”

SLURM on Linux – launch



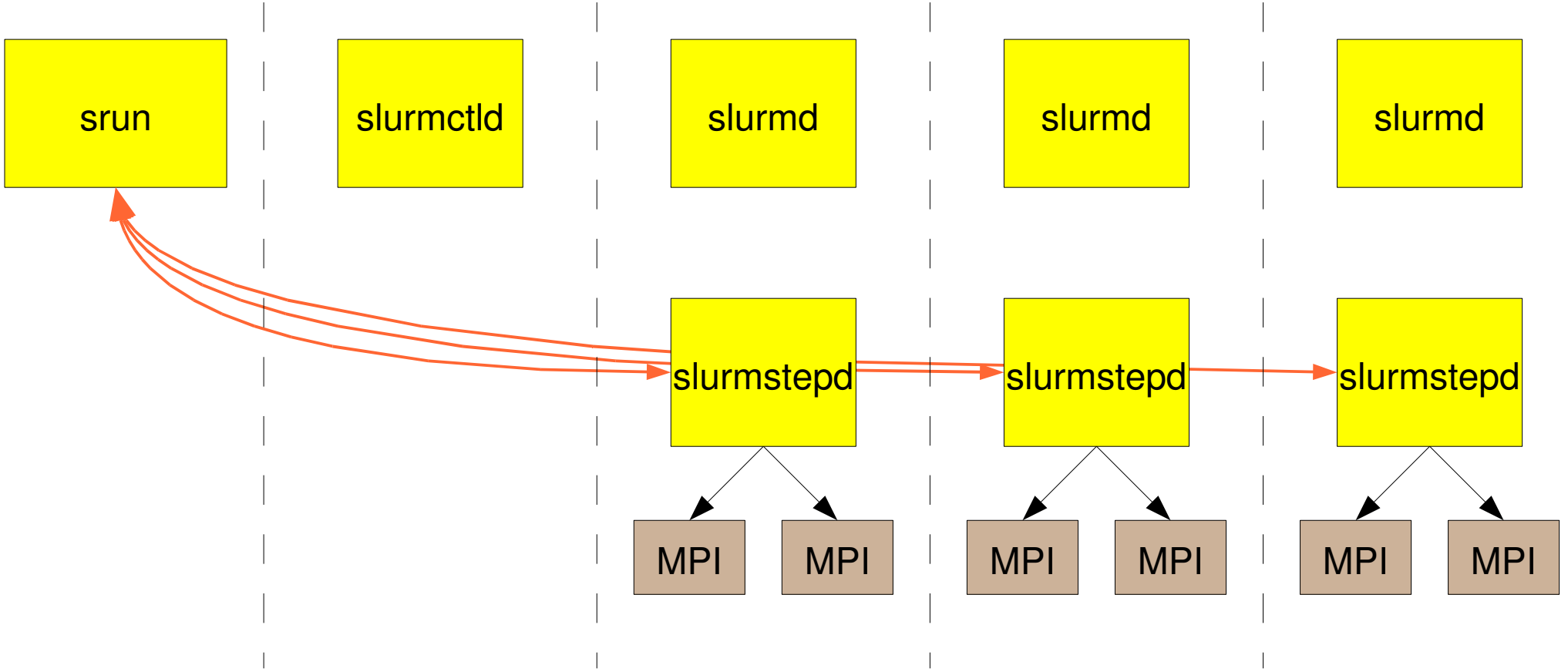
“srun -N3 -n6 -ppdebug mpiprog”

SLURM on Linux – launch



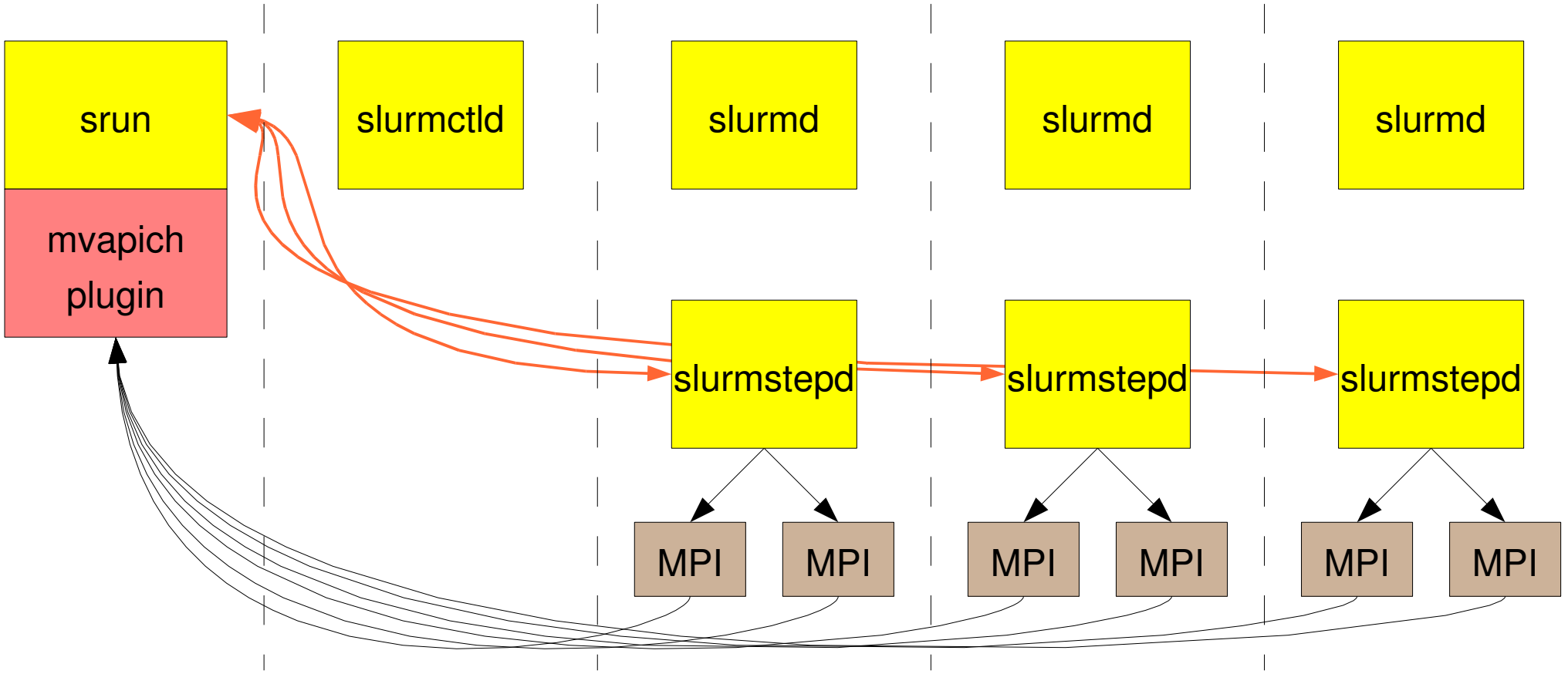
TCP streams for task standard IO (one per NODE)

SLURM on Linux – launch



"srun -N3 -n6 -ppdebug mpiprog"

SLURM on Linux – Infiniband launch

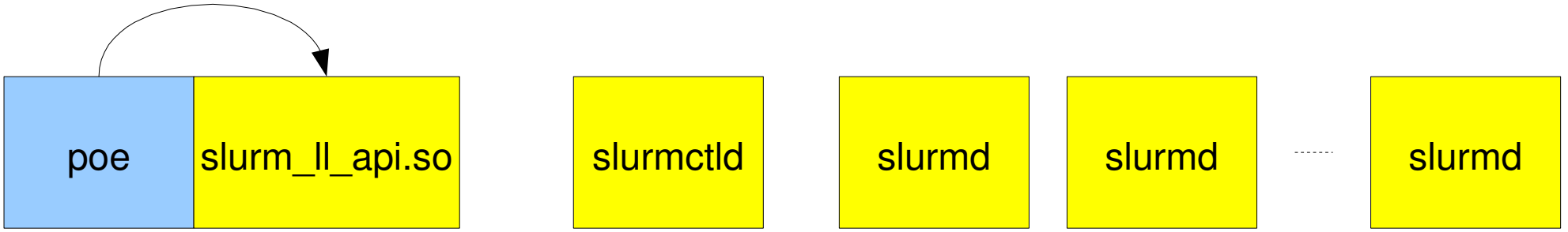


MPI_Init()

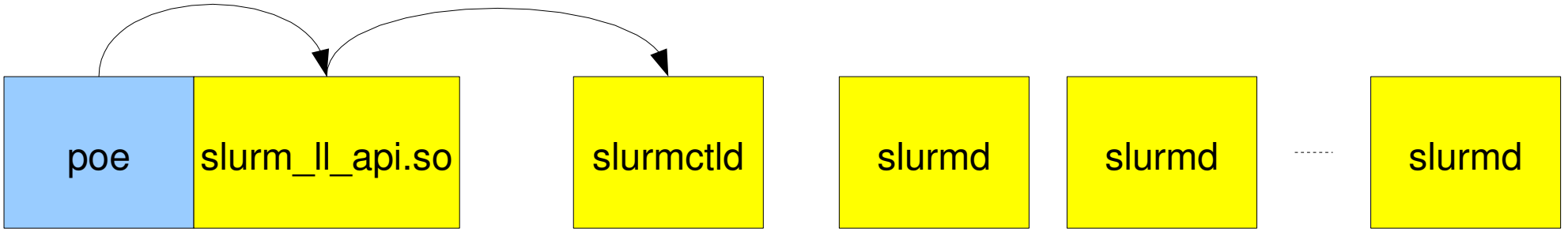
SLURM on AIX

- Must use IBM's “poe” command to launch MPI applications (SLURM replaces LoadLeveler)
- Limited number of switch windows per node + unreliable step completion

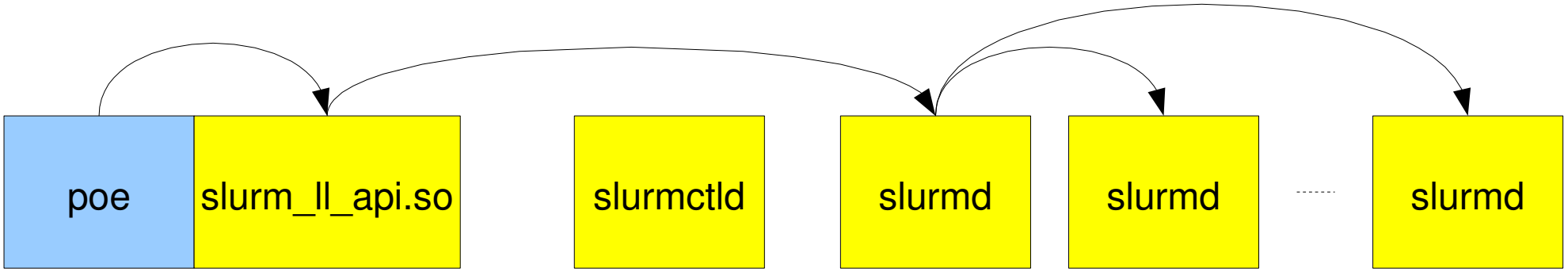
SLURM LoadLeveler API Library



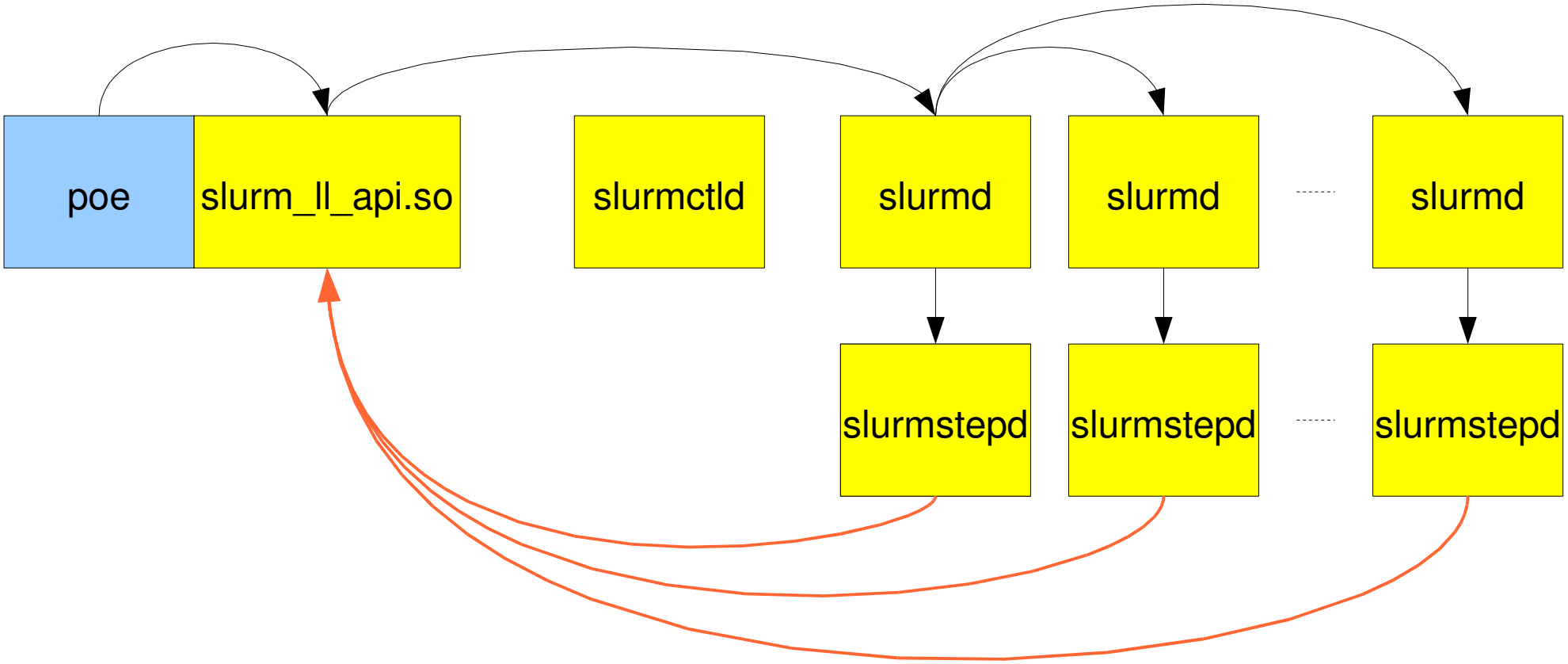
SLURM LoadLeveler API Library



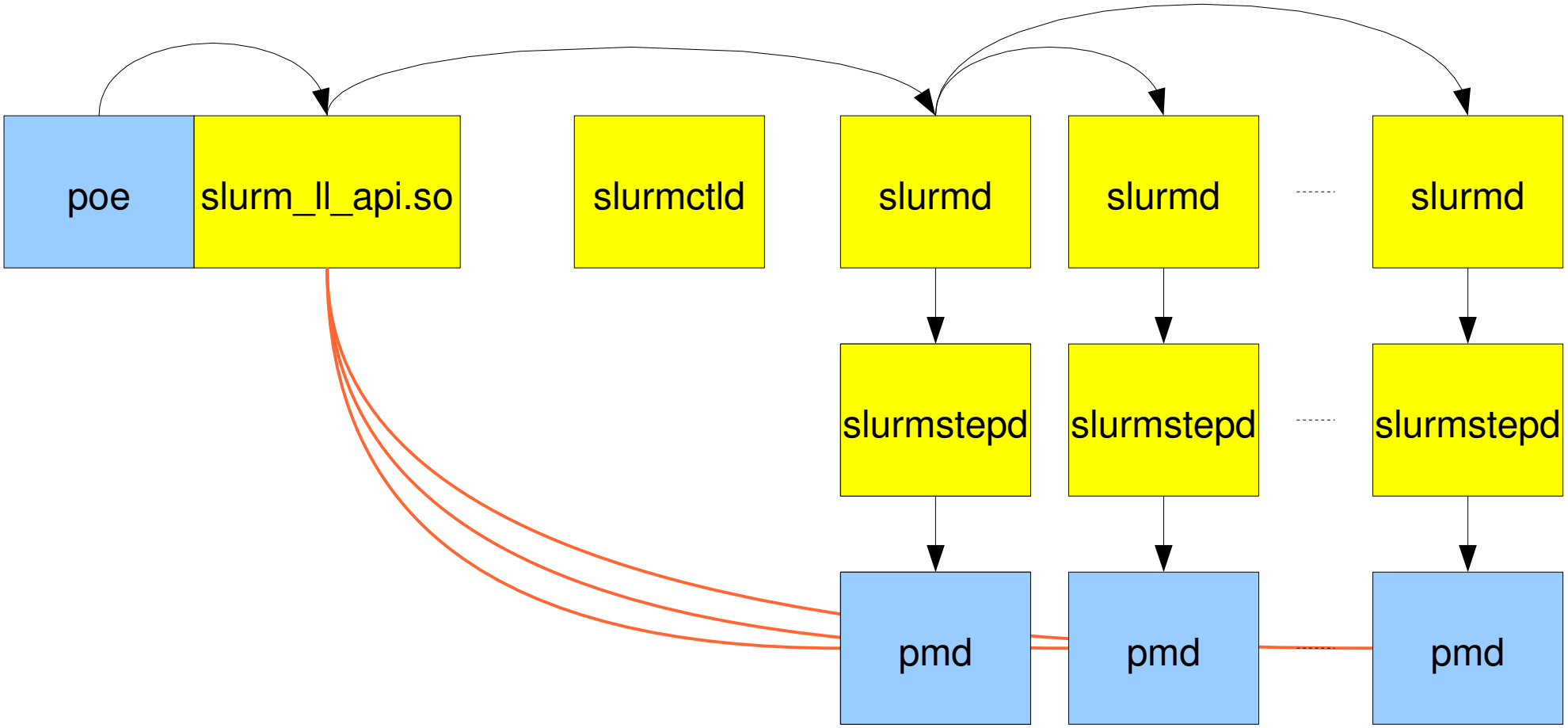
SLURM LoadLeveler API Library



SLURM LoadLeveler API Library

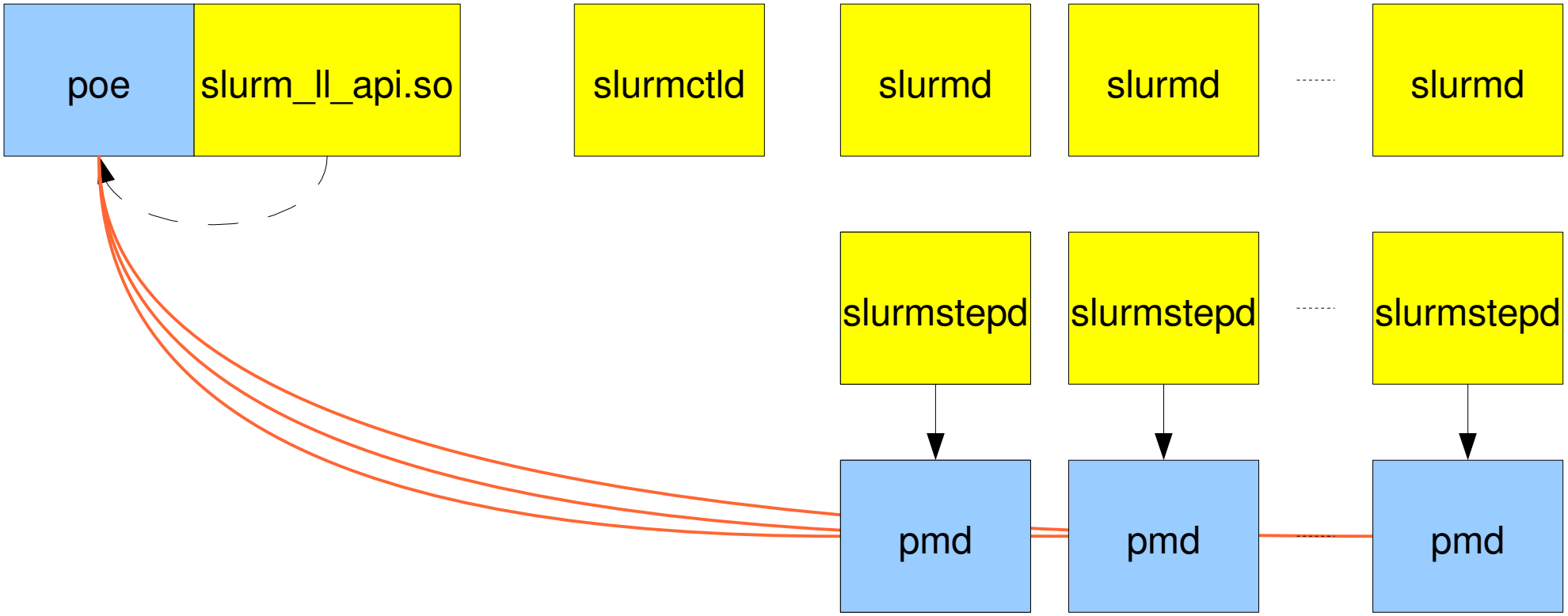


SLURM LoadLeveler API Library



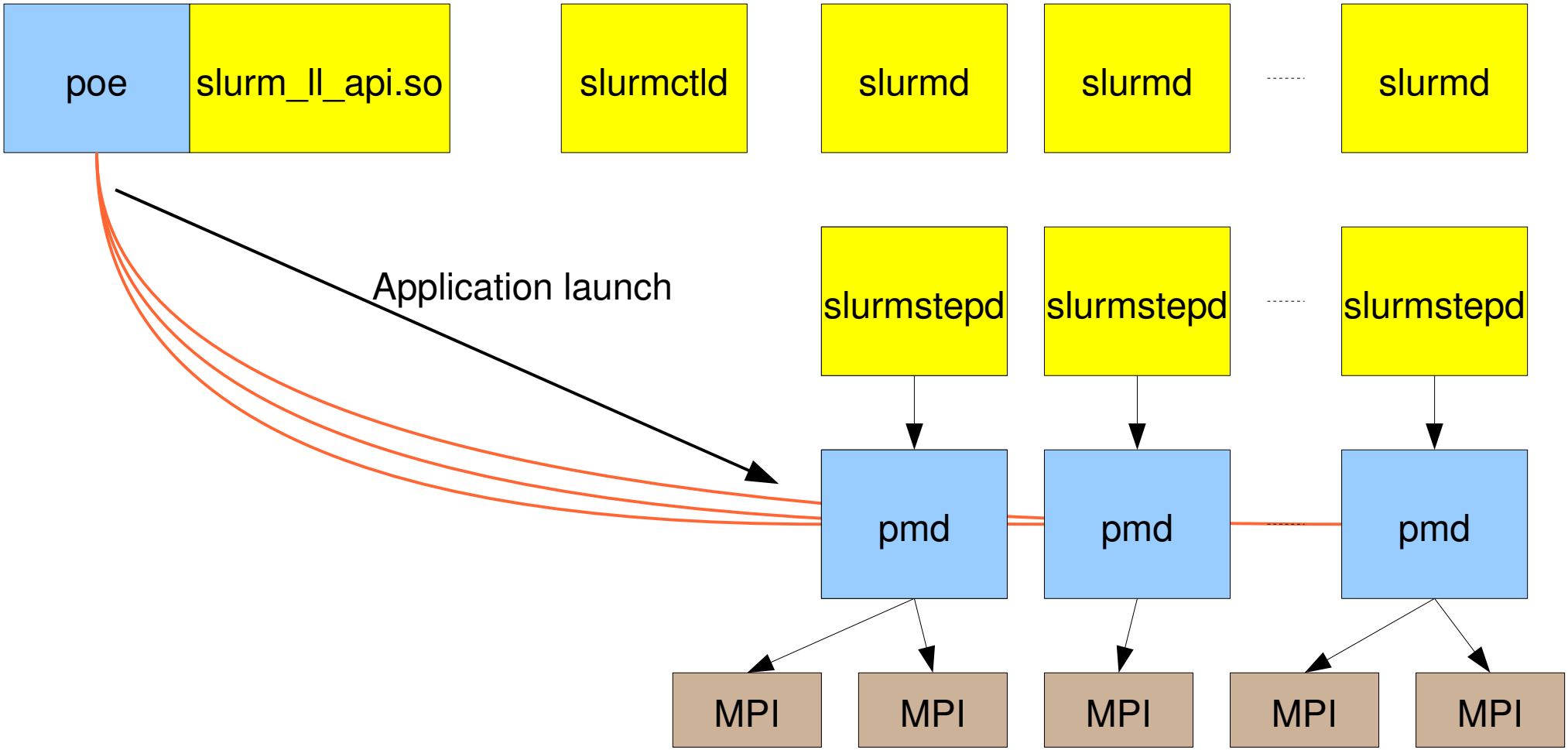
SLURM

SLURM LoadLeveler API Library



SLURM

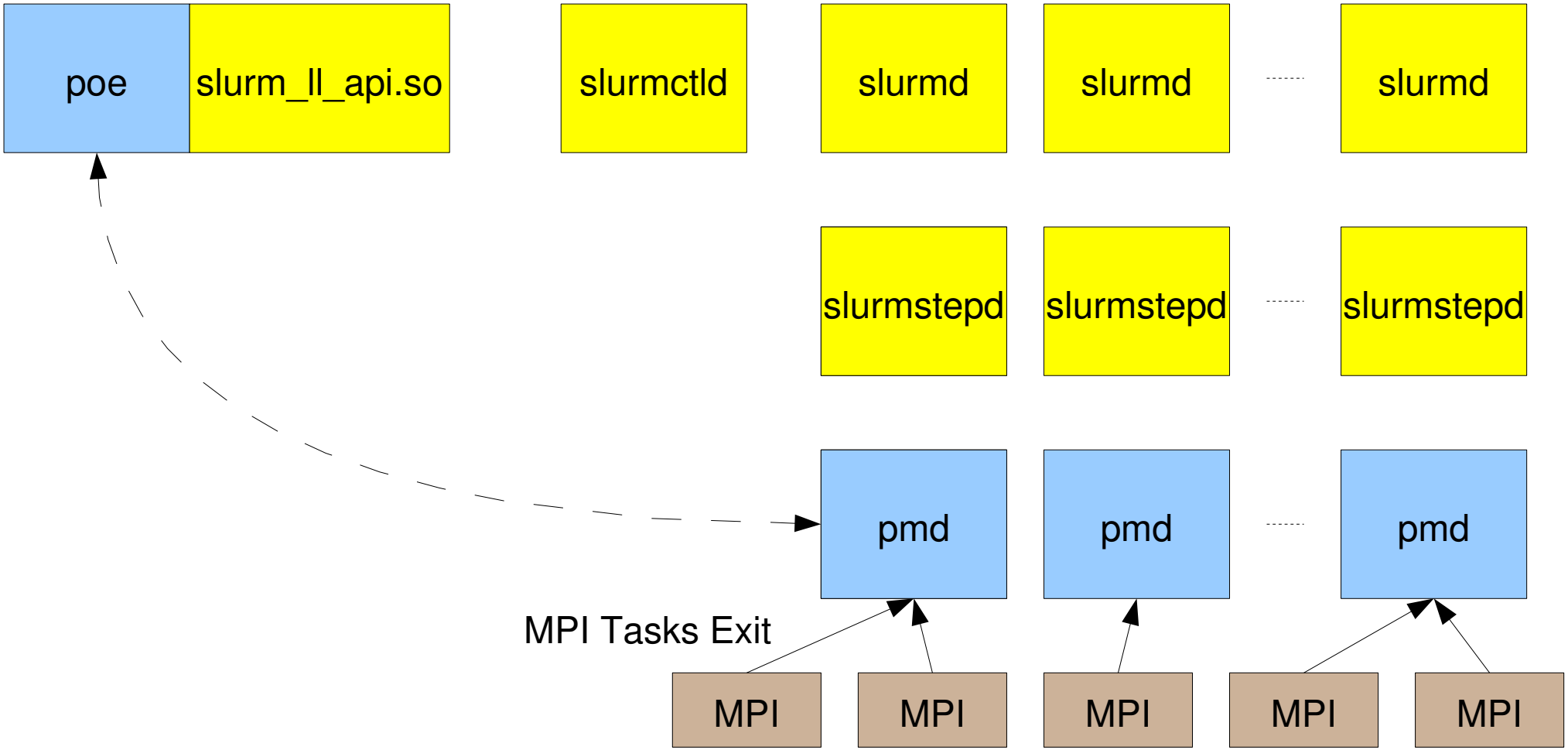
SLURM LoadLeveler API Library



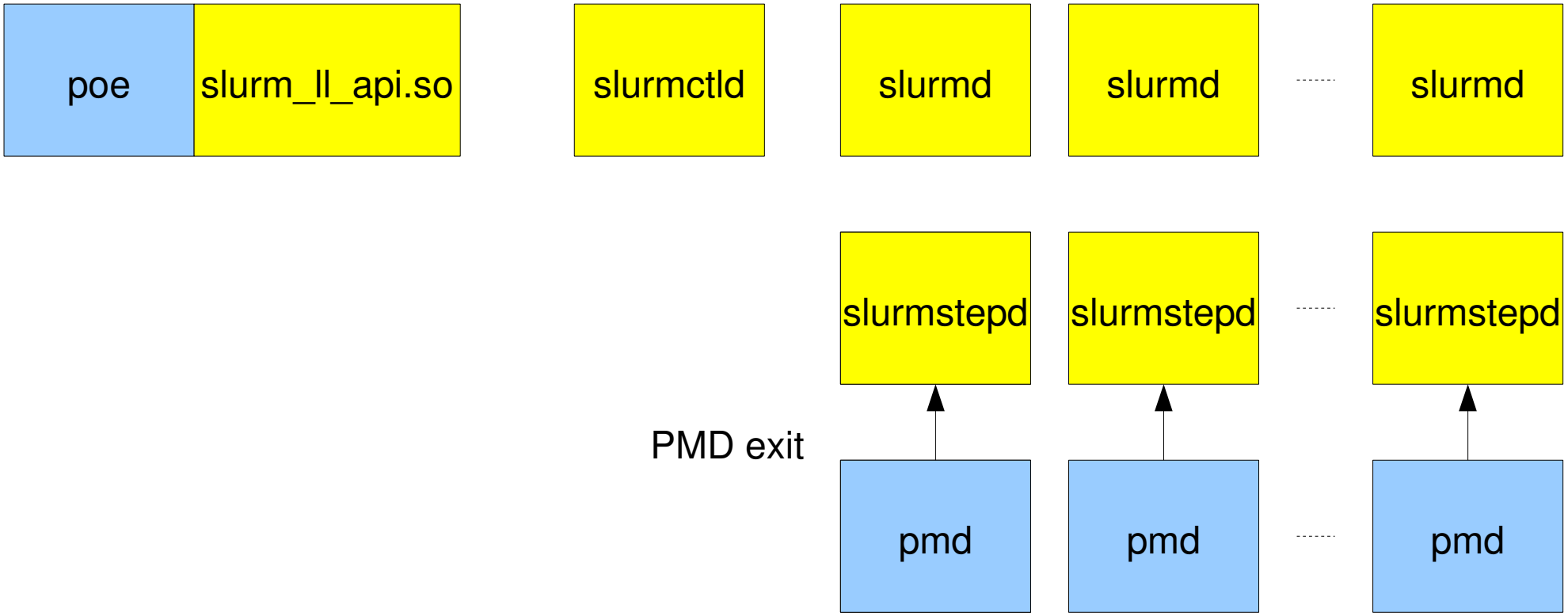
AIX - Limited Switch Windows

- 16 switch windows per adapter (32 per node)
- At 8 tasks per node, only two applications can be launched before windows are exhausted

Old SLURM Step Completion

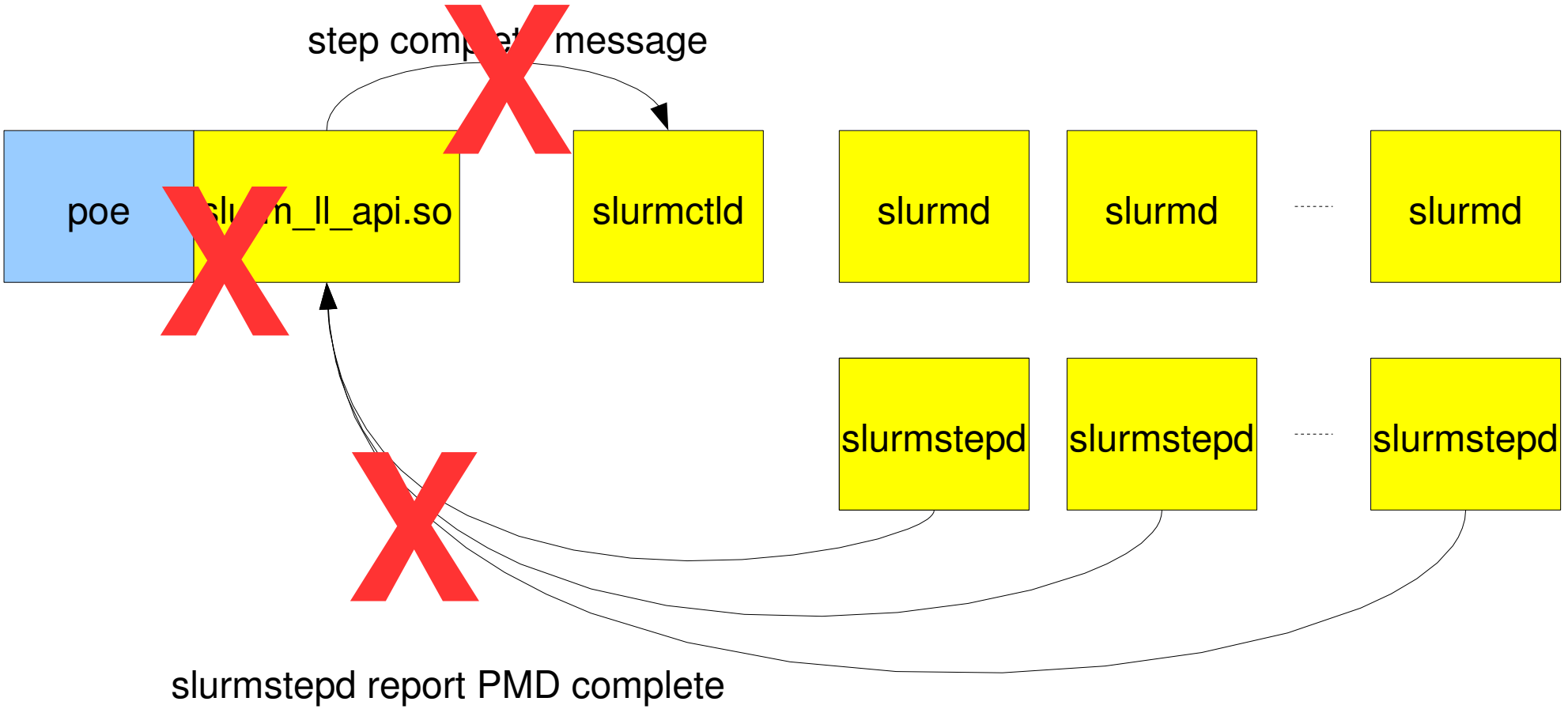


Old SLURM Step Completion

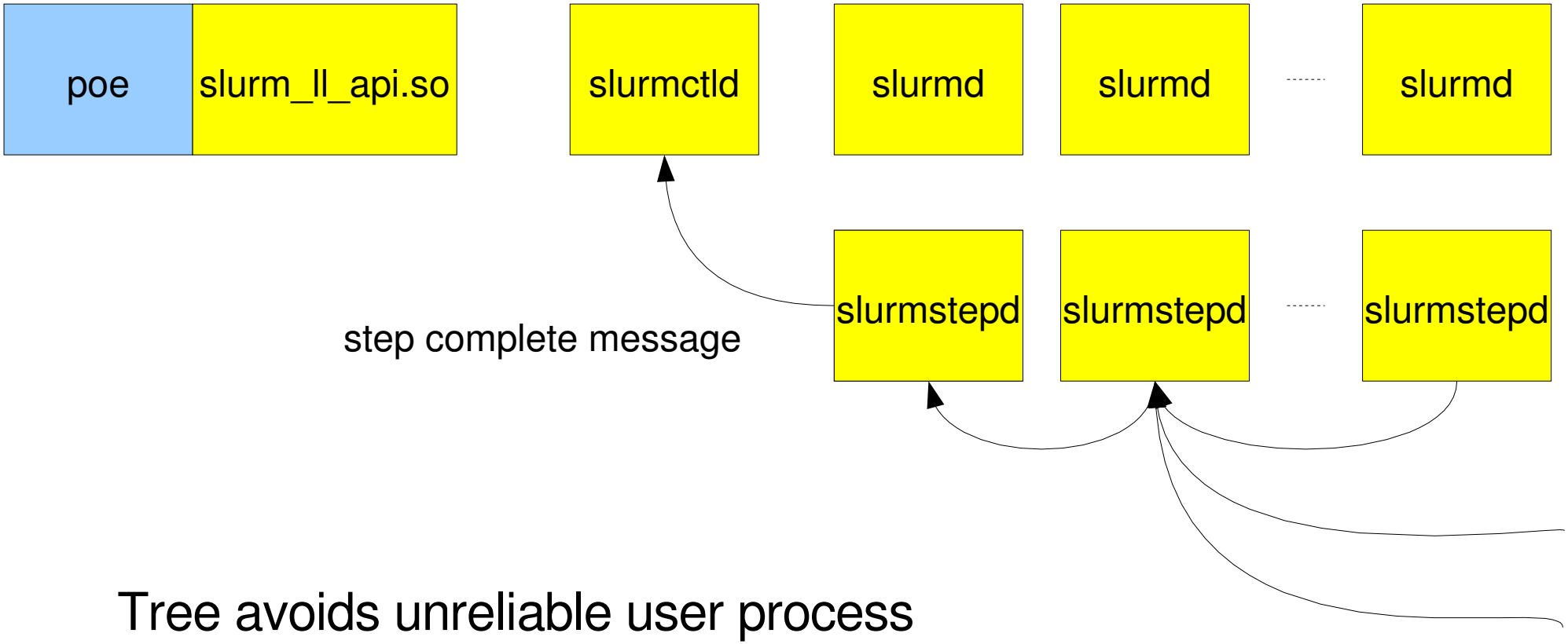


Old SLURM Step Completion

Possible problems if POE killed



New SLURM Step Completion

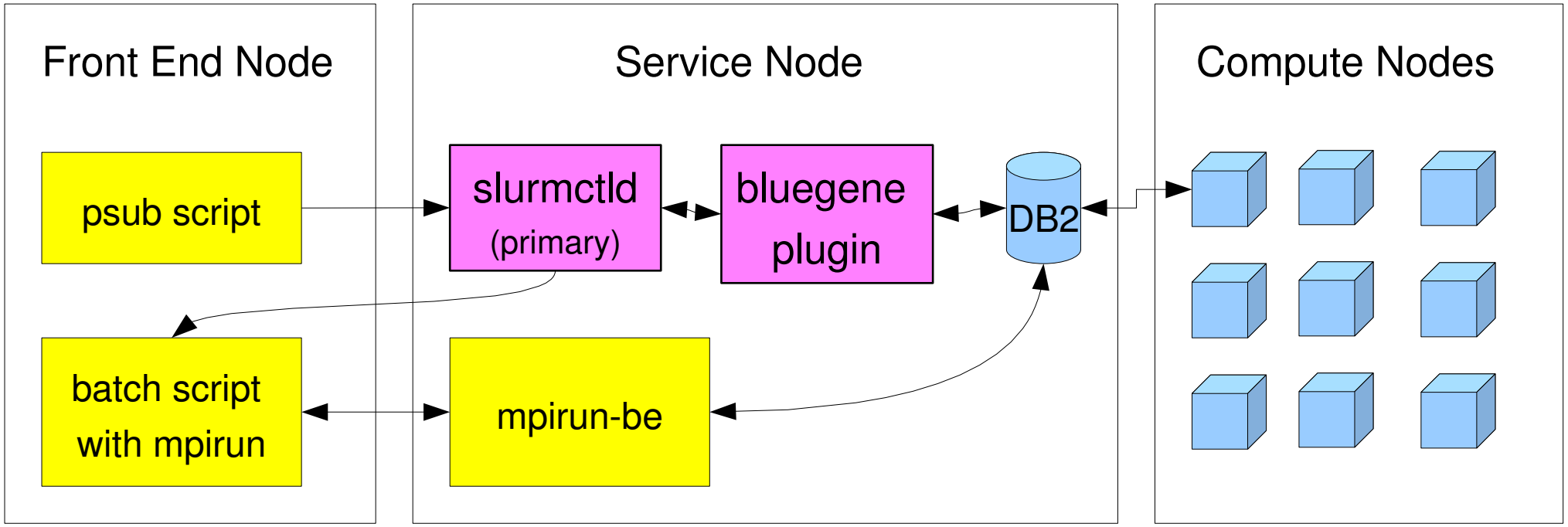


Tree avoids unreliable user process

Bluegene Differences

- Uses only one slurmd to represent many nodes
- Treats one midplane with many compute nodes (c-nodes) as one SLURM node.
- Creates “blocks” of bluegene midplanes (group of 512 c-nodes) or partial midplanes (32 or 128 c-nodes) to run jobs on
- SLURM “wires” together the midplanes to talk to each other
- Use IBM's “mpirun” command to launch MPI applications (SLURM replaces LoadLeveler)
- Monitoring the system through an API into a DB2 database to know the status of the various parts of the system

Bluegene Job Request Flow



New user commands

- salloc – Create an resource allocation, and run one command locally
- sbatch – Submit a batch script to the queue
- sattach – Attach to a running job step
- slaunch – Launch a parallel application (requires existing resource allocation)
- srun – Launch a parallel application, with or without and existing resource allocation

More new user commands

- sbcast – File broadcast using hierarchical slurmd communication
- strigger – Event trigger management
- sview – GTK GUI for users and admins

New Command Examples

```
salloc -N4 -ppdebug xterm

sbatch -N1000 -n8000 mybatchscript

sbatch -N4 <<EOF

#!/bin/sh

hostname

EOF

sattach 6234.15
```

Multi-Core Support

- Resources allocated by node, socket, core or thread
- Complete control is provided over how tasks are laid out on sockets, cores, and threads including binding tasks to specific resources to optimize performance
  - Explicitly set masks with --cpu_bind and --mem_bind OR
  - Automatically generate binding with simple directives
- HPLinpack speedup of 8.5%, LSDyna speedup of 10.5%
- Details at http://www.llnl.gov/linux/slurm/mc_support.html

Example:

srun -N4 -n32 -B4:2:1 --distribution=block:cyclic a.out

Task distribution across nodes : within nodes

Sockets per node : cores per CPU : threads per code

Future Enhancements

- Gang scheduling and automatic job preemption by job partition/queue (work by HP)
- Full PMI support for MPICH2 (work by Bull)
- Improved integration with Moab Scheduler (especially for BlueGene)
- Support for BlueGene/P
- Improved integration with OpenMPI
- Job accounting in proper database (not data file)
- Automatic job migration when node failures are predicted
- Support for jobs to change size