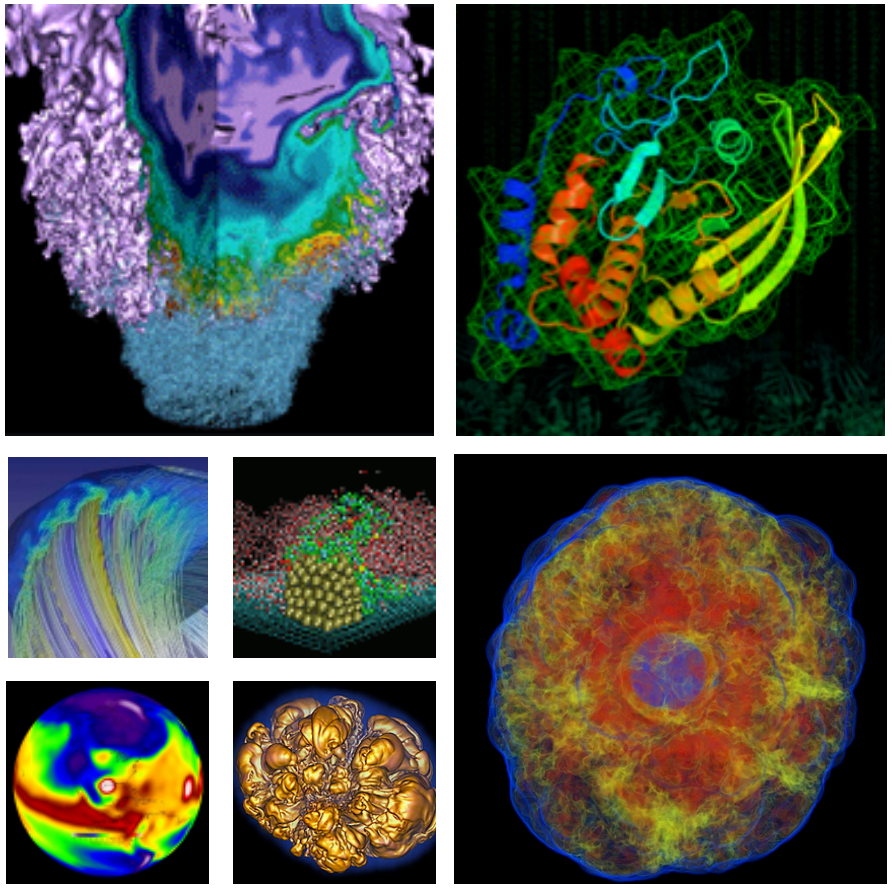
Native SLURM on the XC30



Doug Jacobsen, James Botts
NERSC Computational Systems Group

Slurm User Group Meeting
16 September 2015

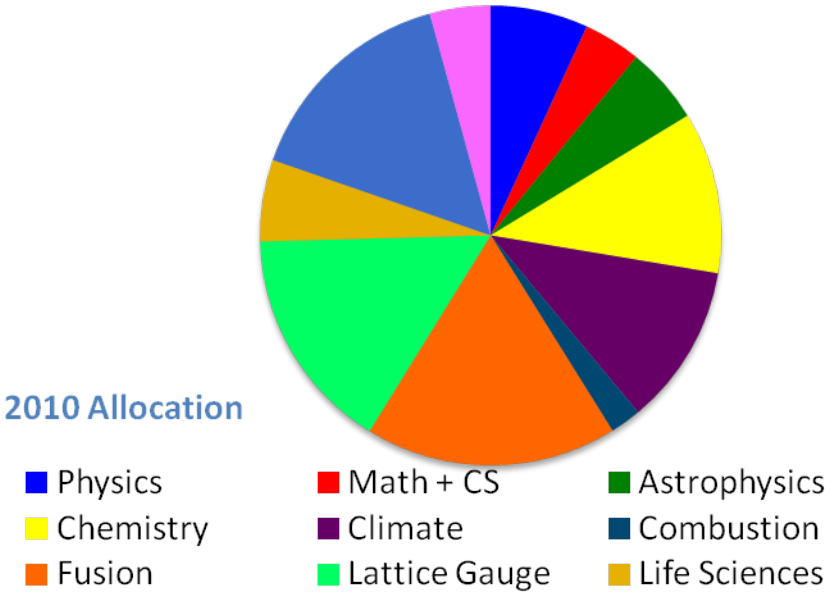
NERSC 40 YEARS at the FOREFRONT 1974-2014

U.S. DEPARTMENT OF ENERGY | Office of Science

BERKELEY LAB

Snapshot of NERSC

- Moving from the Oakland Scientific Facility to a new building at LBNL - CRT Facility
- NERSC is the primary computing facility for the US DOE Office of Science
- Division of LBNL
- over 5000 users
- over 400 projects
- 40th Anniversary in 2014



Systems at NERSC – SLURM Scale Tests

NERSC-7 Cray XC30
5576 Nodes
133728 cores
2.6 PFlops Theoretical

NERSC-6 Cray XE6
6384 Nodes
153216 cores
1.3 PFlops Theoretical

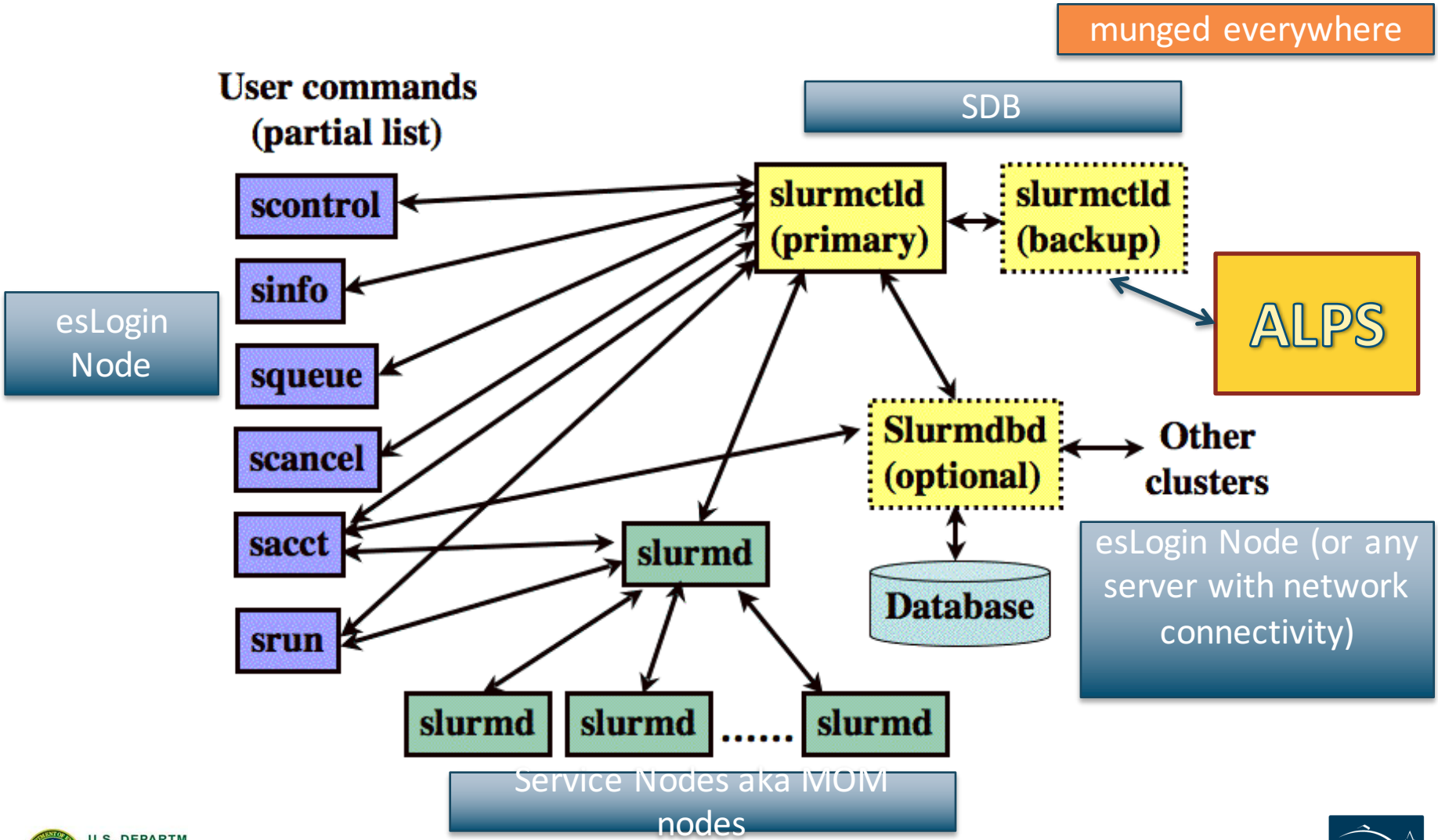
Carver
IBM iDataplex
1202 compute nodes
9984 cores
106.5 TFlops
Theoretical

Global Filesystem and HPSS Data Archive

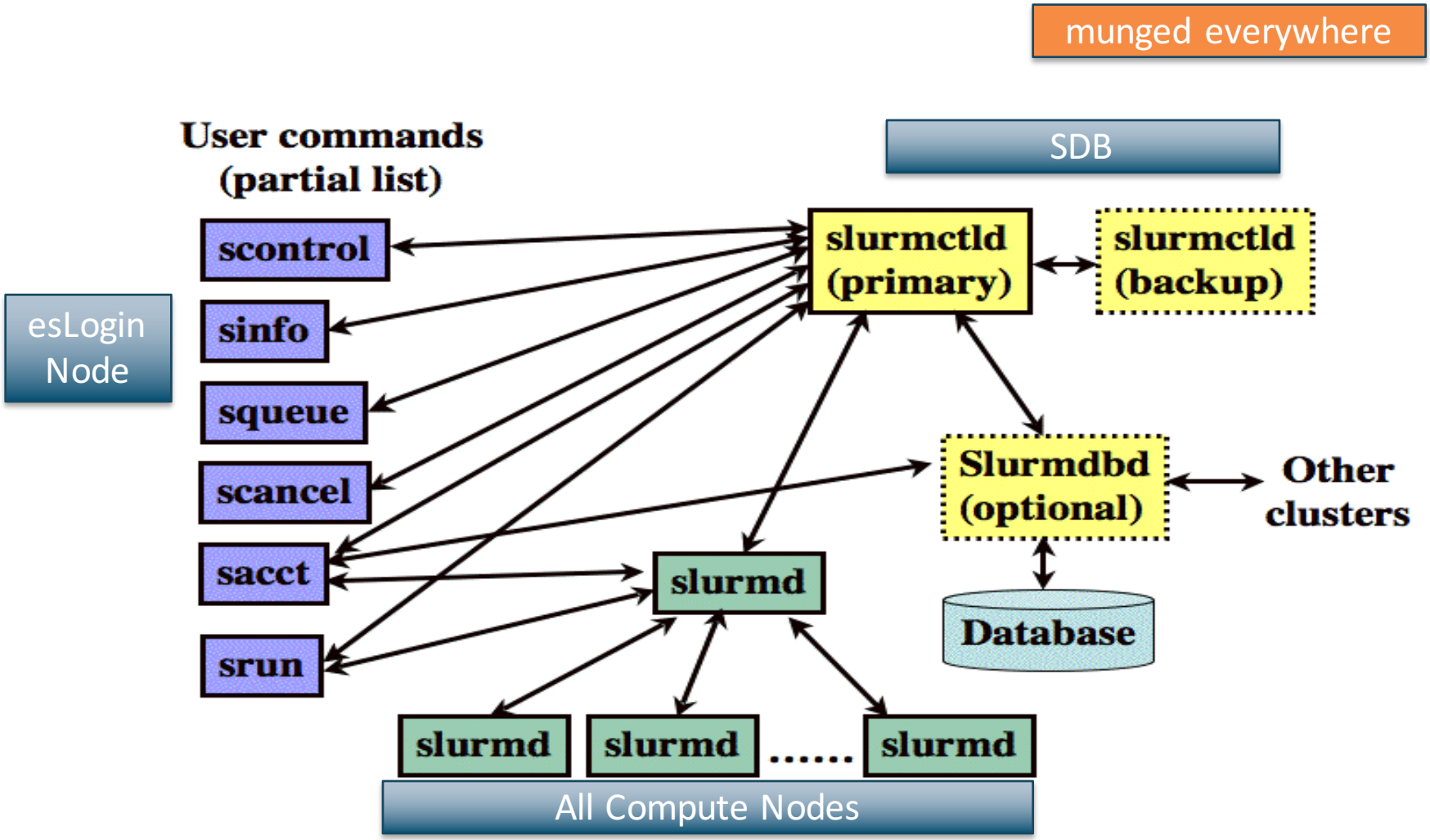
SLURM on a Cray

- **At scale tests on non-Cray traditional SL6 cluster had no surprises, easy configuration**
- **Porting batch configuration from Torque/Moab straightforward**
  - routing queues implemented in job_submit.lua
  - verification of user allocation through perl script called by job_submit.lua
- **First tests on Crays were using Hybrid Slurm on TDS – 10-20 compute nodes**
- **At scale tests were run on both the production XE6 and XC30 with Hybrid Slurm**

What runs where? Hybrid Slurm on a Cray



Native Slurm

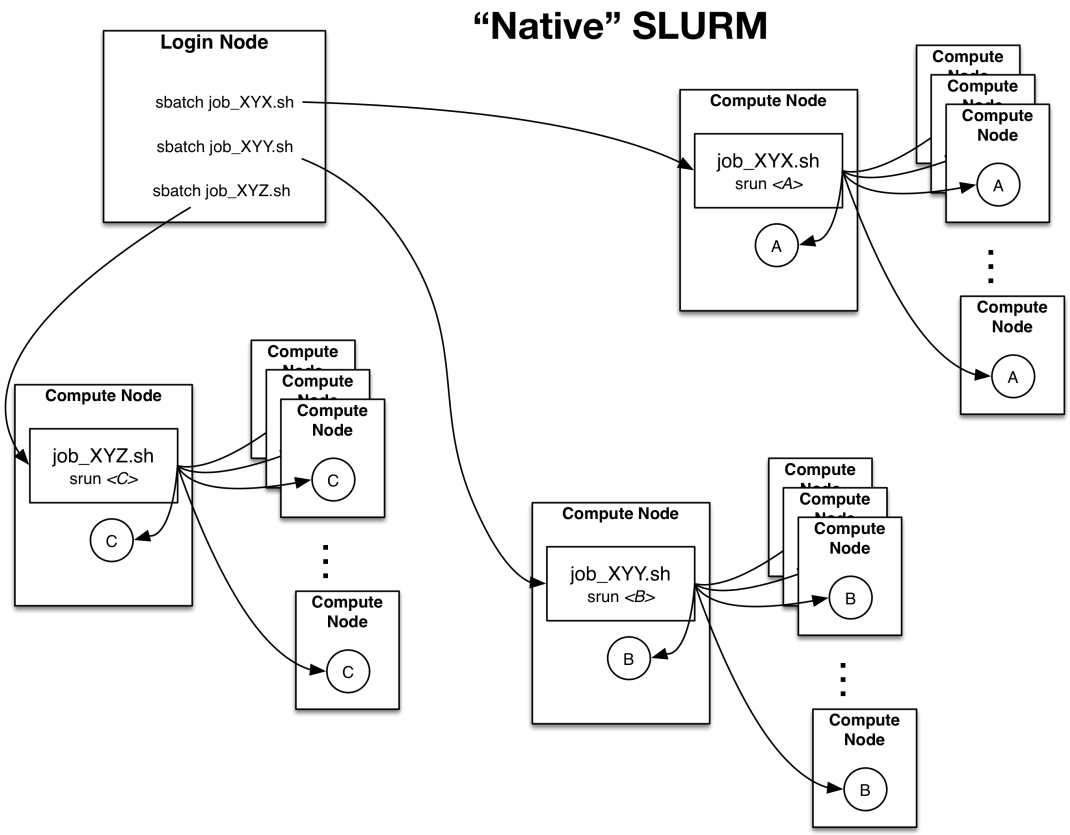


Native Slurm

- **Only works on Aries network (not XE6 with Gemini, e.g. hopper@NERSC)**
- **Requires CLE 5.2UP01 or later**
- **slurm does it all with alpscomm for low level interfaces for network management**
  - launches tasks
  - monitors node health
  - manages node state
- **cannot resize job**
- **no aprun – use srun**
- **No ALPS**
- **No RUR**
- **Supports MAMU (multiple user, multiple jobs) of up to four concurrent jobs on a node**
- **But can run as many single core jobs as desired on a node**
- **even fewer moving parts – recommended by Cray and SchedMD**
- **Hybrid Slurm deprecated with Slurm 15.08 release**
- **Customers running Native SLURM will be on the SLURM community feature roadmap**
- **uses the standard programming module from PE**
- **statically linked apps require relinking**

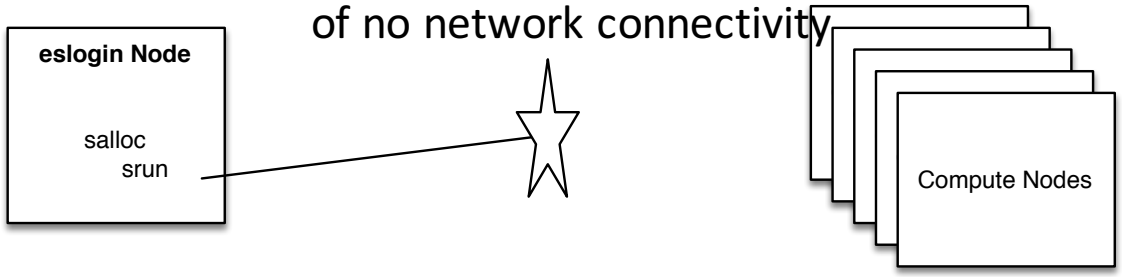
Native SLURM Architecture

- Job batch scripts run on compute nodes, not MOM nodes
- SLURM control daemon (not shown)
  - like moab/pbs_server/apbasil all-in-one
  - Runs on internal service node

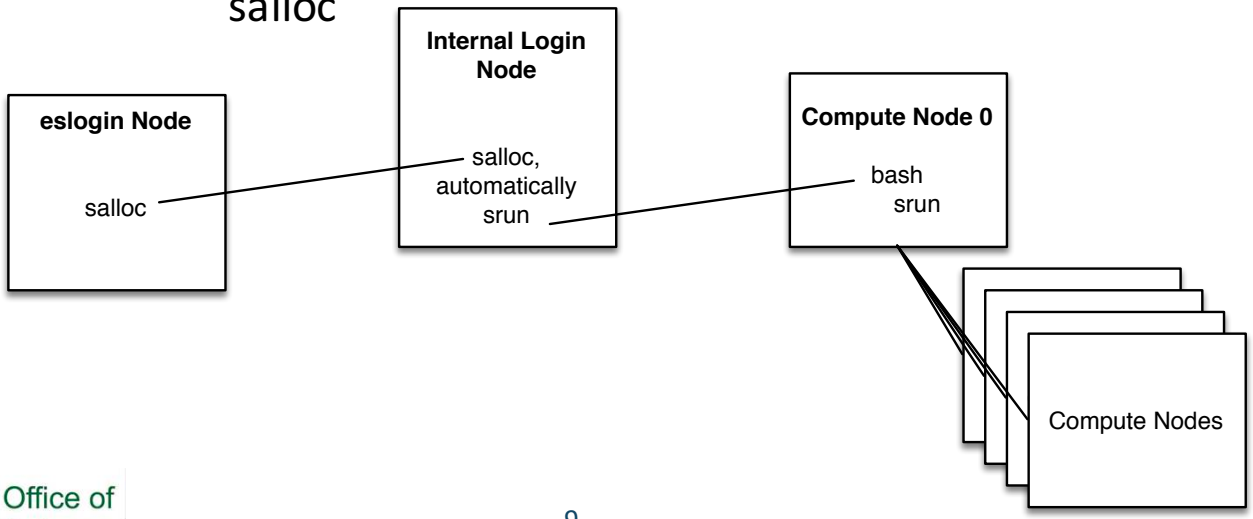


Native SLURM Architecture

Basic salloc fails because of no network connectivity



Use integrated wrapper to ssh to internal login node, transfer environment and run salloc



Aims of scale test

- **Determine if native SLURM is functional at full scale on NERSC edison-scale system**
- **Determine if native SLURM is usable at full scale on NERSC edison-scale system**
- **Validate that simulated NERSC workload functions efficiently**
  - Can achieve close to full system utilization for a sustained period of at least 2 hours
  - Schedule jobs with queue depth of 3000 jobs
    - running + pending entries in squeue >= 3000
  - "s" commands (sbatch, squeue, sinfo) responsive when system packed, ideally within a few seconds; >30s fail.
  - Job dispatch to "head" compute node occurs "quickly"
    - Time from slurmctrld job prolog start to batch script control start
  - srun dispatch to compute nodes occurs "quickly"
    - Median time from issuing srun to application start
  - "Quickly" – median time for dispatch should within some acceptable variance (perhaps 10%) of current or faster based on job scale

Switching to Native SLURM from ALPS

1. **Install slurm into shared root (had *almost* no effect on running system)**
   - Default slurm installation installed libpmi.so.2 that superseded cray libpmi for dynamically linked codes. After reporting, SchedMD disabled libpmi installation for cray systems.
2. **Modify compute node image to enable slurm on boot (cannot be done post-boot correctly)**
3. **Modify compute node config in shared root**
   1. nsswitch.conf, use ldap for passwd, group
   2. compute-dsl-services.conf, start munge
4. **Enable slurm, munge services in xtopview**
   1. Starts munge and slurmd on service nodes
5. **Reboot system**

Test timeline – 05.27.2015

- 0700 – Finish slurm 14.11.7 prep, reboot system
- 0830 – system up, discover config issue, determine faster to correct and reboot than manually correct compute nodes
- 0930 – system up, slurm online
- 1020 – functional test complete, start scale test
- 1112 – request help from schedmd, slurm commands become unresponsive once full utilization, no queue forming
- 1146 – receive advice to adjust config from schedmd; everything clears up
- 1500 – scale test period complete, begin targeted experimentation
- 1530 – switch slurmctld save state to GPFS, no problems
- 1600 – undo config changes, shutdown system
- 1615 – return system to Cray onsite for production boot
- 1800 – system available to users

Step 1: Basic Functionality Verification through Automation

python 3 + green test runner – about 200s to complete
basic health check – run serially (parallel execution is supported but ran into issues)

- **checks slurm version, daemons running (munge, slurmd, slurmctld)**
- **runs single jobs, mpi jobs**
- **dependencies**
- **submits to a reservation**
- **job arrays**
- **tests serial jobs**
- **tests jobs submitted to a node list**
- **gres**
- **scancel**
- **accounting**
- **CPU affinity**
- **Hold and Release Job**
- **wrapper scripts emulating torque – e.g. qsub, qstat, etc.**

edison scale test

- Job size selected randomly using between 1 – 3152 nodes
- Job size selection weighted by computed pdf based on NERSC workload sampling in June 2014
- 3 different MPI codes used: psnap, osu_alltoall, internal NERSC "A3"
- 1 serial code --- serial jobs failed to be submitted due to configuration issue in scale test script
- Job wall time request normally distributed around 2400s with 2000s std dev.
- Target execution time random using normal dist 1800s with 1333s std dev.
- Executable re-run as many times as is required to hit target execution time

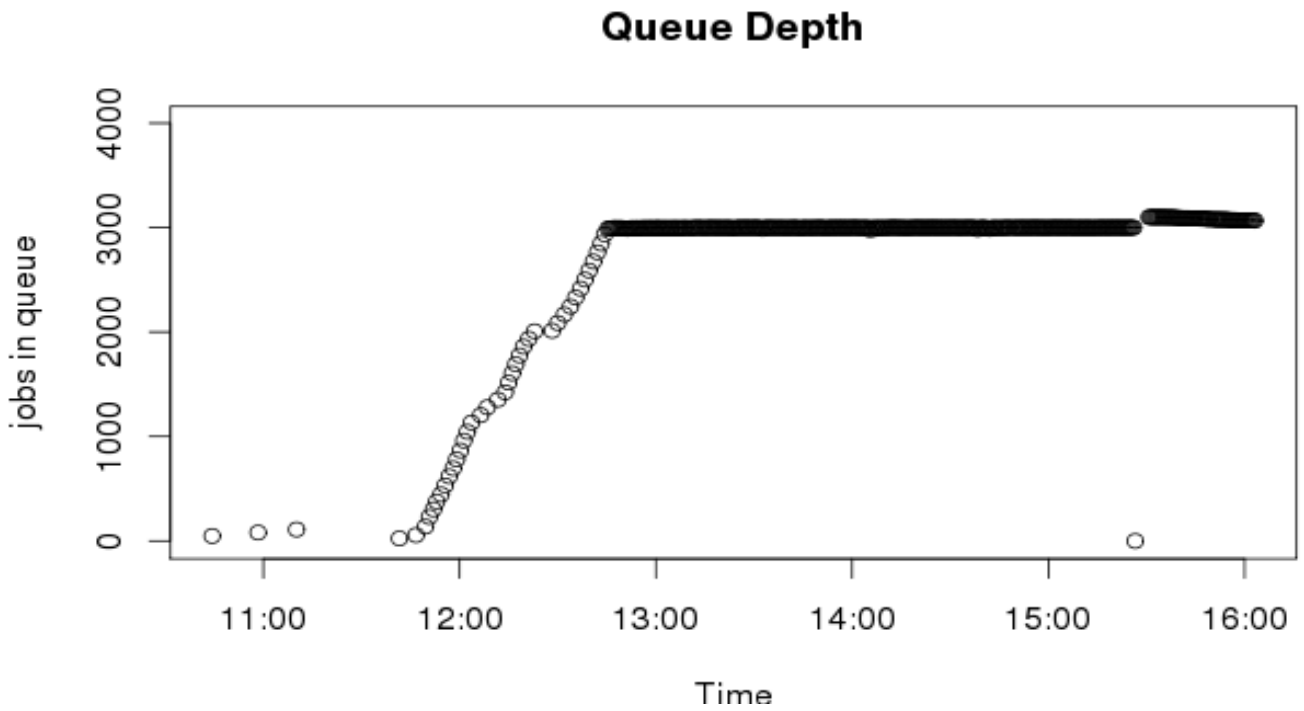
Details, details . . .

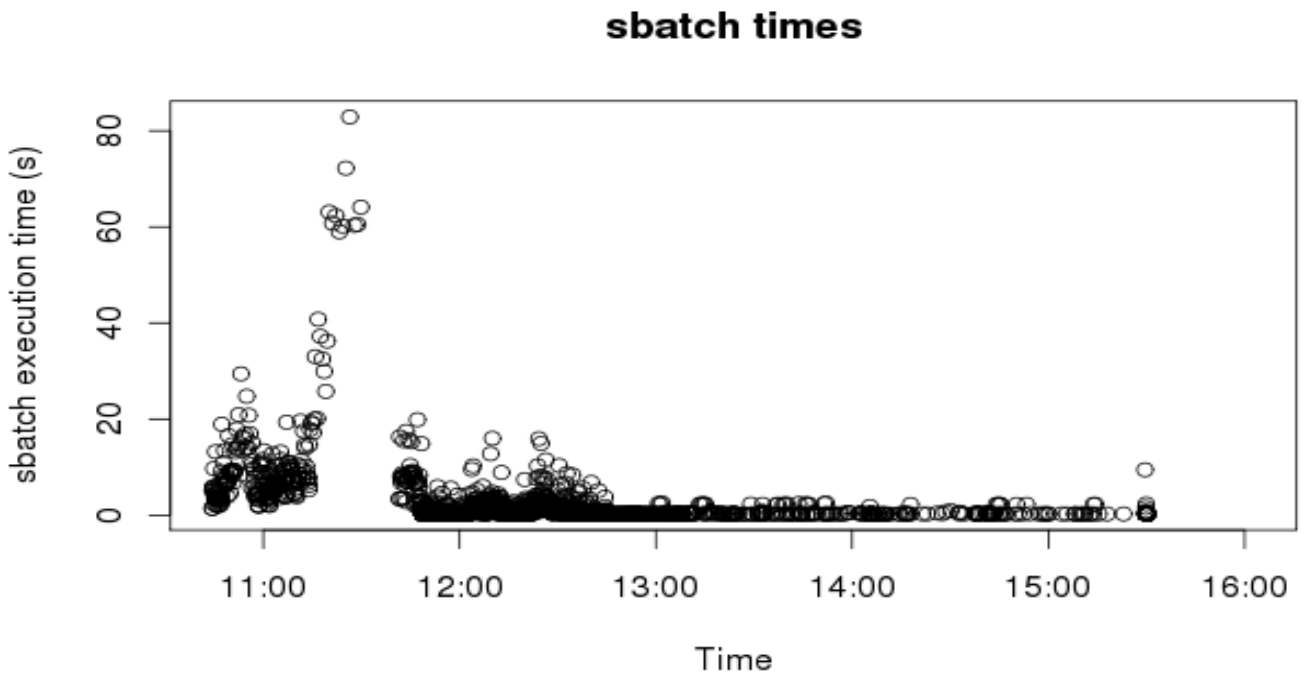
- Notified SchedMD 5 days previous that we were doing this in a bugzilla case (1692)
- Initial sluggishness at scale was reported in the ticket at 1112 – by 1126 had a response, by 1146 had the "magic bullet" – remove

DebugFlags=SelectType

from slurm.conf and do scontrol reconfigure

- After completion, discovered that jobs larger than 1024 nodes didn't run – RSIP exhaustion – patch given on the same day we reported the issue.
- aeld log went wild when had HA slurm config – fixed within the day (not part of Edison test – interesting detail)

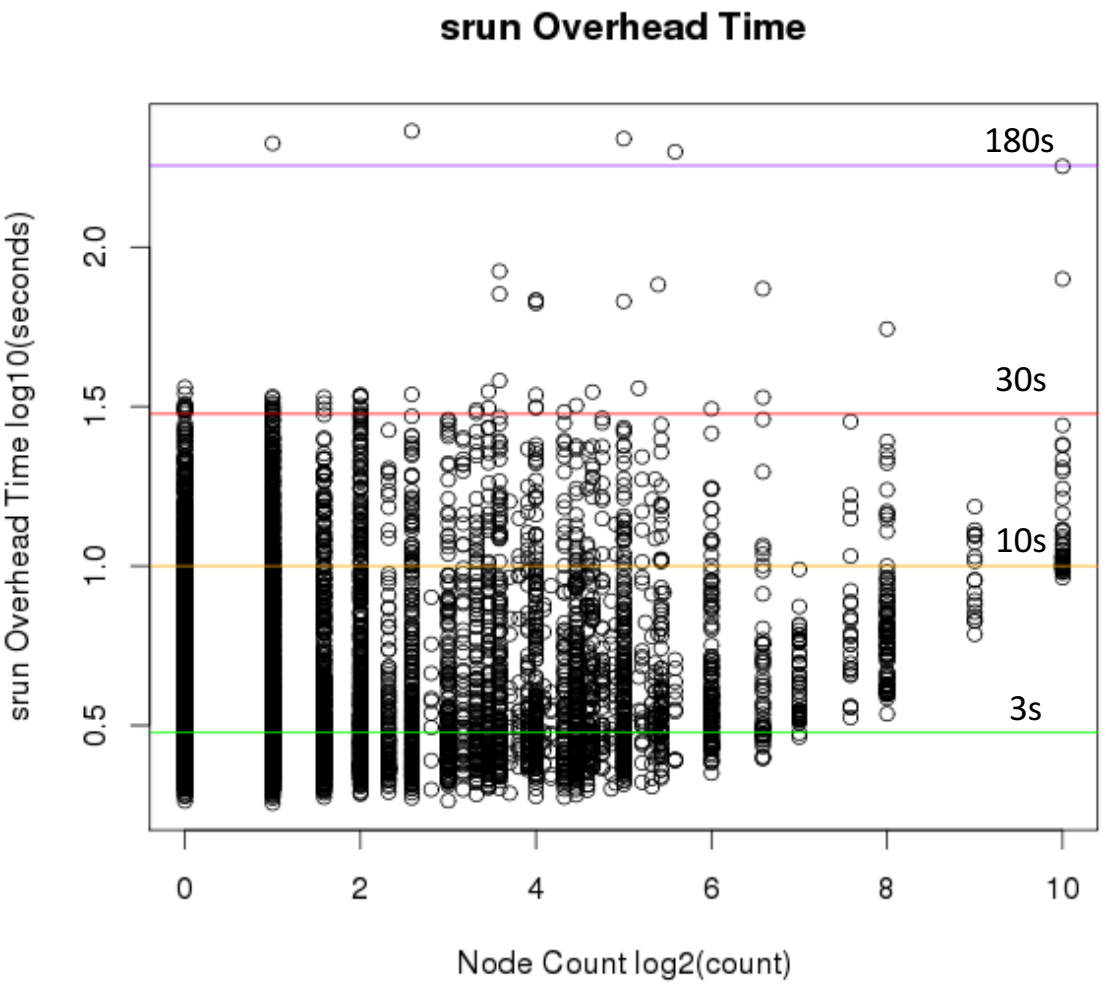

Queue Depth
jobs in queue
0
1000
2000
3000
4000
11:00
12:00
13:00
14:00
15:00
16:00
Time


sbatch times
sbatch execution time (s)
0
20
40
60
80
11:00
12:00
13:00
14:00
15:00
16:00
Time

srun Performance

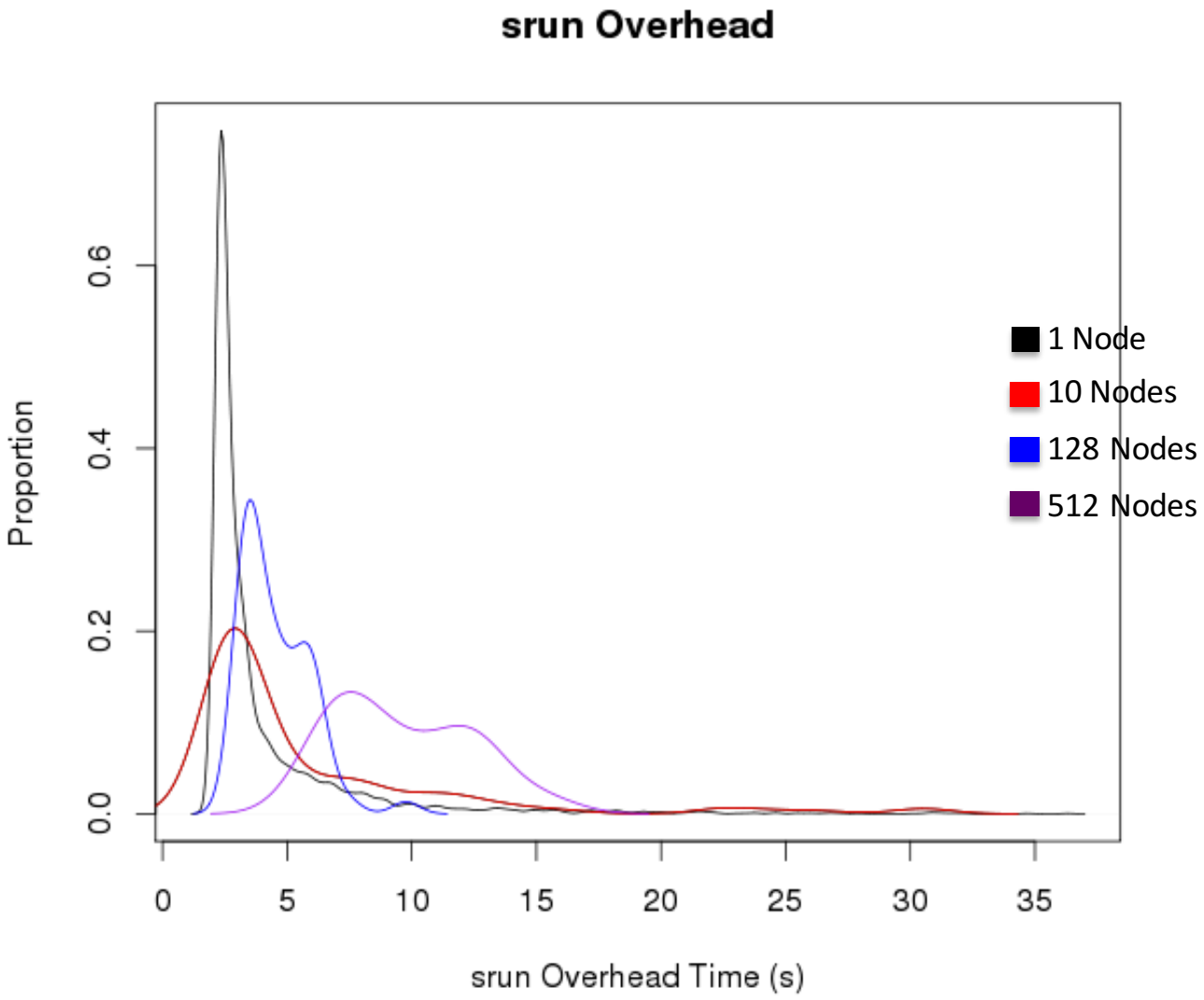
SLURM Performance

- Obtained timings for "srun overhead"
  - Time from when batch script executed srun until processes were running on the compute nodes
- Counted sruns executed during "good" portion of test 11:45 – 15:00
- 12,036 sruns in dataset
- Trend that higher node counts result in greater overhead (expected)



srun Performance for common job shapes

Data on this plot are taken from vertical groups from previous slide plot.



Edison scale test results

- **After correcting logging issue, SLURM scaled well enough to accept NERSC workload mimic and run in expected times**
- **Very large jobs failed to run properly (>1024 nodes)**
  - Similar to issue observed at another site
  - Caused by SLURM exhausting rsip ports (opening listening socket binding to all interfaces)
  - Patched, tested on alva using “multi slurmd” capability to run 1600 “nodes” on a single blade of alva (edison TDS)
    - aeld threw errors if we used more than a single blade

Moving towards Production

- **Open Questions**

  How to update SLURM or SLURM configuration on live system?

  – Updating SLURM or SLURM configuration on live system tricky due to DVS caching of shared root.

  – On test system drop fs caches, issue command to re-read slurm config.

  – Unclear if dropping filesystem caches would be advisable in production (assume not).

  – Can DVS_CACHE=off environment variable used correctly help? (Note: can't propagate to slurmstepd)

DVS caching behavior

- different DVS caching behavior on compute nodes and service nodes.
- make changes to configuration files and install software upgrades using xtopview on the boot node in the standard manner and find that
- due to caching, get unpredictable errors such as:

```
dmj@mom:~/psnap/native$ sbatch psnap.batch
safeopen(): refusing to open
`/etc/opt/slurm/plugstack.conf', which is a soft link
sbatch: error: spank: Failed to open
/etc/opt/slurm/plugstack.conf: No error
sbatch: error: Failed to initialize plugin stack
dmj@mom:~/psnap/native$
```

Moving towards Production

- **Open Questions**

  Is there a performance "impact" to running slurmd?

  – psnap indicates that overall system noise is comparable on edison (possibly lower, but hard to tell on a freshly rebooted system)

  – Reliance on nscd for LDAP on all compute nodes may have scaling issues for large srun jobs

  – Need to measure memory footprint delta to ALPS (low priority, considered negligible)

  – Have measured data for job dispatch vs. job size; unaware of similar data for torque/moab. Will analyze soon.

Thank you! (And we are hiring!)