LA-UR-12-24558
Approved for public release; distribution is unlimited.

Title:	DOE ASC L2 Review for Milestone 4466
Author(s):	Daniels, Marcus G
Intended for:	ASC L2 Review for Milestone 4466, 2012-09-06 (Albuquerque, New Mexico, United States)

DOE ASC L2 Review for Milestone 4466

Cielo Production Capability Readiness

A user perspective

Marcus Daniels <mdaniels@lanl.gov>

Me: A Developer on Eulerian Applications team

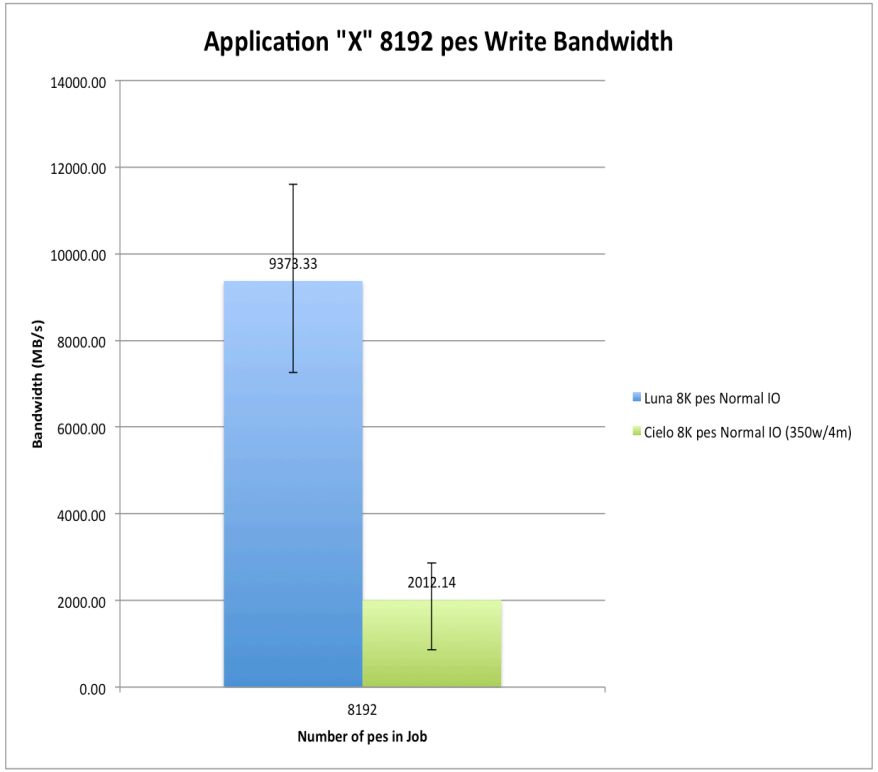
- **Increase the scalability of the application**
  - Facilitate models with more than 2 billion cells
  - Widespread code changes -- don't break working models
  - 3D asteroid problem with 2 billion cells
    - 32 TB of RAM
    - 3 TB checkpoints at this scale
  - Test mesh manipulation beyond the 2 billion cell limit
    - Adding one or more refinement levels to a model with 2 billion cells can exhaust the memory of a machine of the size of Roadrunner (51 TB)
    - Cielo has more room to move (297 TB)
- **Make large existing models run reliably on Cielo**
  - The *Melissa problem* – an example of the largest practical job to run on Cielo
    - 128 TB memory footprint
    - 24 TB checkpoints
      - Cycle times up to an hour -- very valuable steps and checkpoints

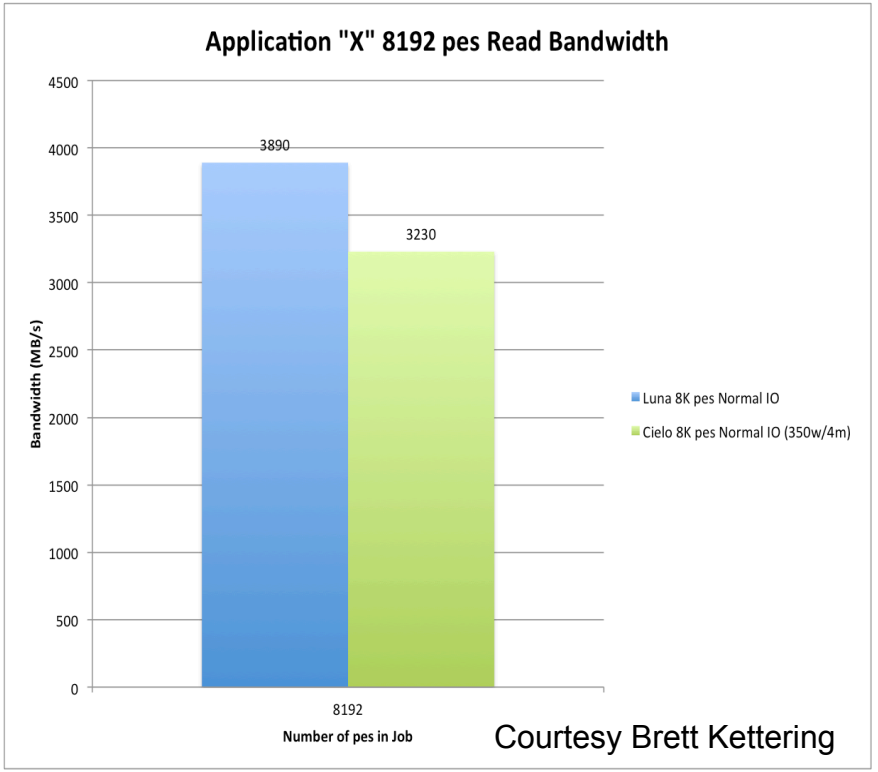
Production readiness -- thumbnail

- **Cielo is running several production codes per the most aggressive and agreed-upon usage model**
- **Main obstacle to getting over the bar was scratch I/O performance**
  - The Panasas scheme had unacceptable performance
  - Lustre is much better
- **Remaining issues**
  - Reliability of batch scheduling system
  - Lack of transparency of system makes debugging hard
- **Production readiness cannot be decoupled from developer usability**
  - Some system issues only occur at scale
  - Bug reproducers will often be real-world problems that must run at scale.

Panasas I/O rates (Asteroid Problem)

At this rate: reading and writing of a 24TB *Melissa* checkpoint: 1/3 of a 24 hour allocation





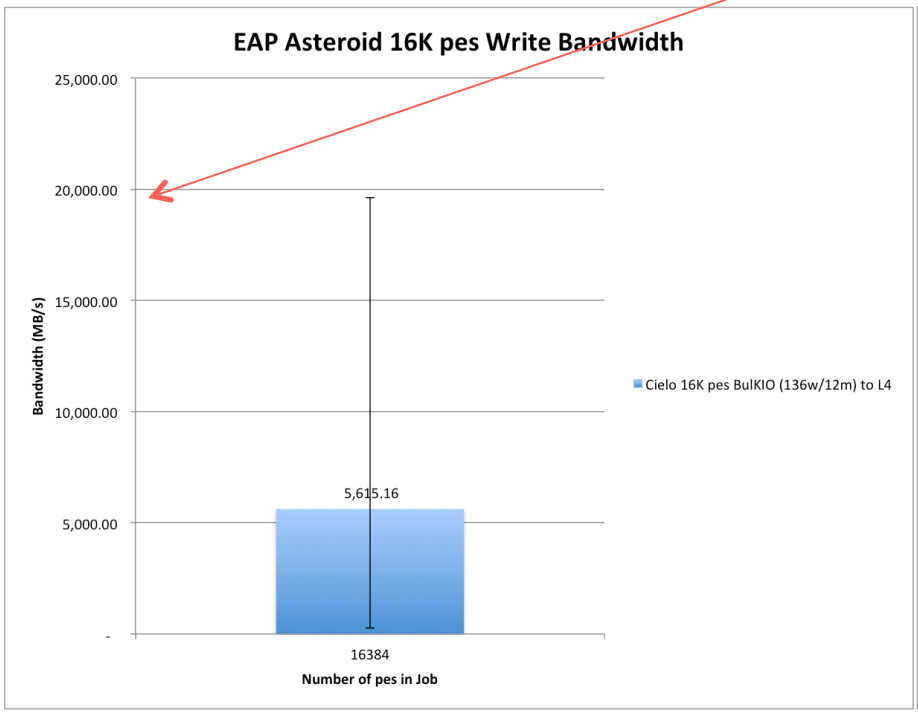
Courtesy Brett Kettering

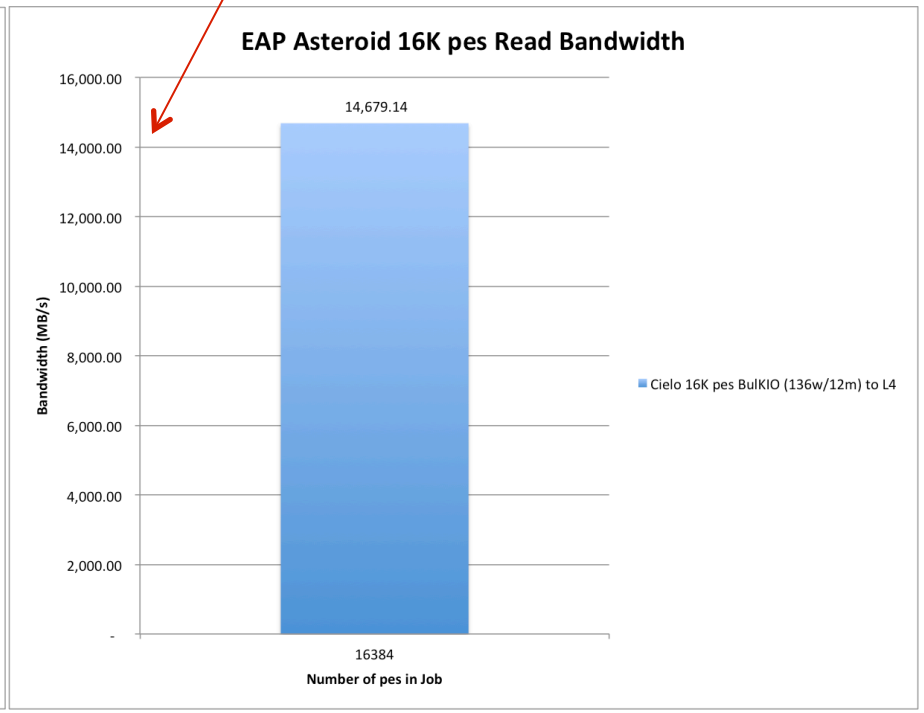
Panasas can also work when it is more tightly integrated with the client – as Luna results above show

Asteroid Problem using Lustre

Reading and writing a 24TB *Melissa* checkpoint: 1/24 of a full day allocation

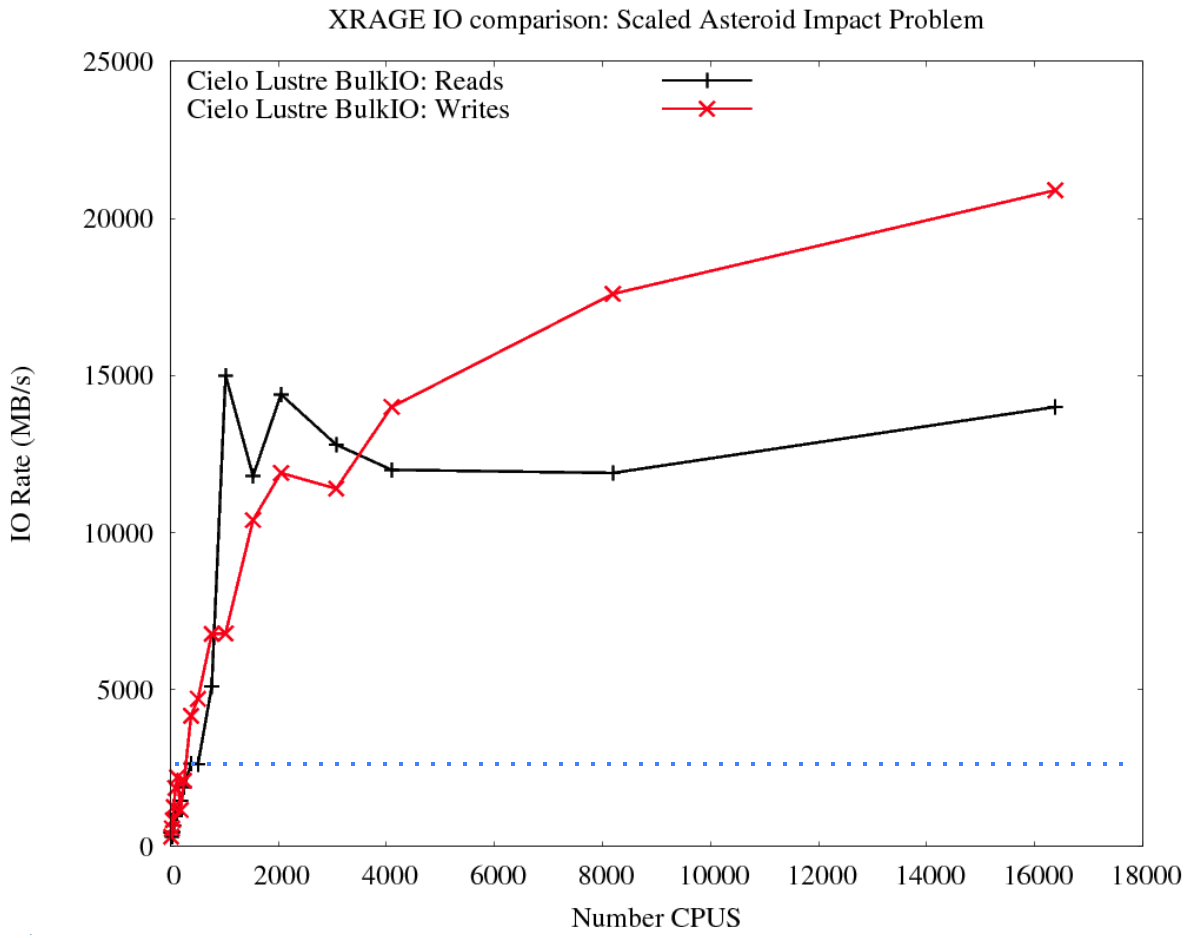
15 GB/sec is typical for a 64k PE *Melissa* run





Courtesy Brett Kettering

Asteroid Problem using Lustre



Courtesy Andy Nelson

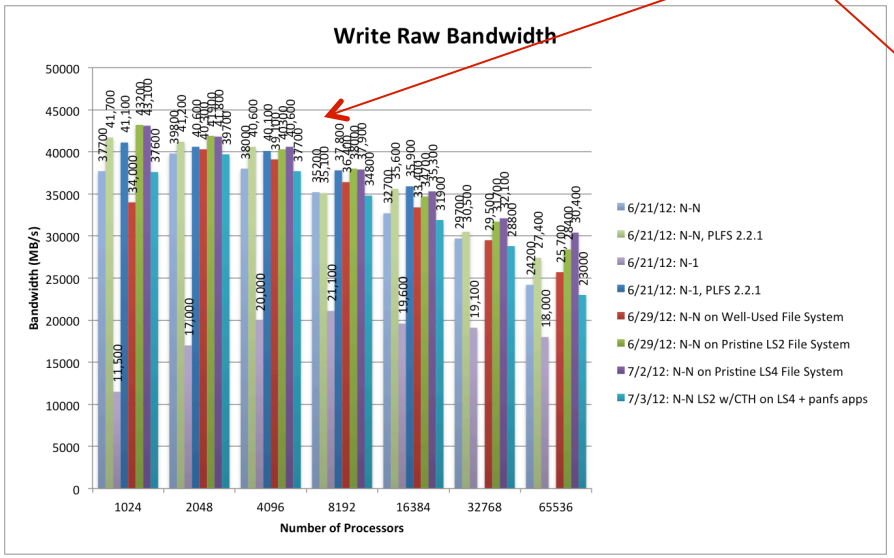
Black line can likely improve by letting readers have more PEs

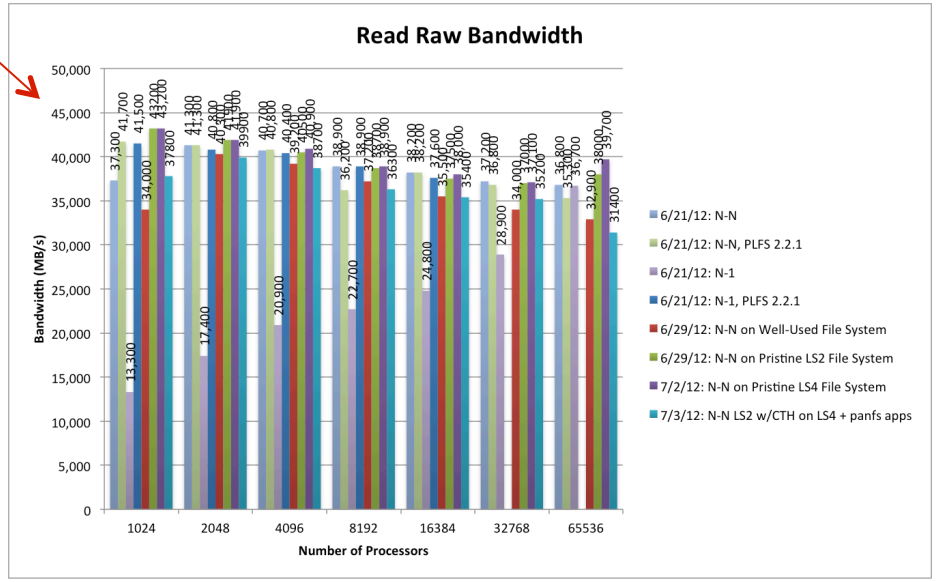
Both red and black lines will improve with larger filesystem

Blue dotted line is approximate legacy Panasas performance

Synthetic benchmarks

- **Application performance within a factor of 2 of synthetic benchmarks.**
- **Panasas was 10 times slower**
- **Existing Lustre filesystems represents ¼ of 160 GB/sec aggregate bandwidth of Cielo drive array hardware, or ~40 GB/sec.**





Courtesy Brett Kettering

The *Melissa* problem

- ½ hour checkpoint reads and writes (@ 24 TB)
  - Instead of 4 hour with Panasas
  - A positive qualitative change in productivity from Lustre
- Running beyond 4096 nodes (64k PEs) is not practical because Cielo is a shared resource
  - With some attention to space vs. time considerations in the code, this problem can reliably run within this resource limit.
- Some other obstacles..

The *Melissa* problem

- **Once the system is fast and stable, push a little harder!**
- **I/O is now asynchronous in the EAP code -- requests to send or receive data can stack up faster than they can be serviced. Symptom:**

[16] ERROR - nem_gni_dump_hugetlb_dir(): Listing of dir '/var/lib/hugetlbfs/global/pagesie-2097152'

[16] ERROR - nem gni_dump_hugetlb_dir(): Filename: '.'

[16] ERROR - nem_gni_dump_hugetlb_dir(): Filename: '..'

Rank 16 [Mon Jul 30 15:14:38 2012] [c0-0cc0s1n0] application called MPI_Abort(MPI_COMM_WORLD, 1)- process 16

MPICH2 ERROR Rank 16 (job id 1646747) [Mon Jul 30 15:14:38 2012] [c0-0c0s1n0] [nid00002] - MPIU_nem_gni_get_hugepages(): Unable to mmap 8388608 bytes for file /var/lib/hugetlbfs/global/pagesize-2097152/hugepagefile.MPICH.12.17201.kvs_1646747, err Cannot allocate memory

MPICH2 ERROR Rank 16 (job id 1646747) [Mon Jul 30 15:14:38 2012] [c0-0c0s1n0] [nid00002] - MPIU_nem_gni_get_hugepages(): large page stats: free 0 nr 88 nr_overcommit 16154 resv 0 surplus 88

*** SEND_DATA_FROMBUF: mpi_error on pe 16 from MPI_Isend = Other MPI error, error stack:

PMPI_Isend(148).....MPI_Isend(buf=0x2aab83a30000, count=12582912, MPI_BYTE, dest=15662, tag=4972161, MPI_COMM_WORLD, request=0x7fffffff8b7c) failed

MPID_nem_lmt_RndvSend(114)

MPID_nem_gni_lmt_initiate_lmt(766).....: failure occurred while attempting to send RTS packet

MPID_nem_gni_iStartContigMsg(1380).....:

MPID_nem_gni_iSendContig_start(1106)..:

MPID_nem_gni_smsg_cm_send_conn_req(577):

MPID_nem_gni_smsg_cm_progress_req(2134).:

MPID_nem_gni_smsg_mbox_alloc(361)......:

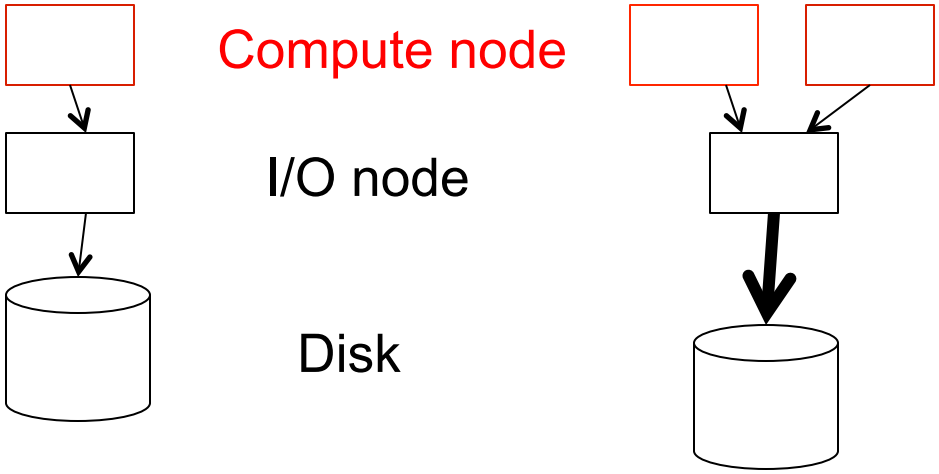
MPID_nem_gni_smsg_mbox_block_alloc(242): Out of memory

_pmiu_daemon(SIGCHLD): [NID 00002]] [c0-0c0s1n0] [Mon Jul 0 15:14:38 2012] PE RANK 1 exit signal Aborted

[NID 00002] 2012-07-30 15:14:38 Apid 1646848: initiated application termination

The *Melissa* problem

- **Relatively more data per IO PE than smaller problems**
- **Best performance by having a number of writers that matches the number of RAID components on the array minus some spares (136).**
- **The larger the number of compute PEs, the more pressure there is on the relatively fewer IO PEs.**



The *Melissa* problem

- Cray uses huge pages for MPI messages. The MPI messages in question are used for Bulk IO.
- Defensively throttle within the application.
- Default to normal heap when huge pages are exhausted
  MPICH_GNI_MALLOC_FALLBACK=enabled
- Use a bigger disk array with more IO PEs that does not get swamped
  - The one built from /scratch5 will be twice as big, and proportionately faster. Will it avoid the problem?
- Tune eager/rendezvous message size thresholds? Exactly how?

Lustre problems (apparently resolved..)

- **In the past have observed file I/O errors of this kind**

BADBOY called mype = 31696 cycle = 407: Unable to sync PIO file: Cx60j-dmp00407 Errmsg: File sync error in bulkioc_sync on pe # 31696: Message: Input/Output error

BADBOY called: mype = 13120 cycle = 407: Unable to sync PIO file: Cx60j-dmp00407 Errmsg: File sync error in bulkioc_sync on pe # 13120: Message: Input/Output error

BADBOY called: mype = 25504 cycle = 407: Unable to sync PIO file: Cx60j-dmp00407 Errmsg: File sync error in bulkioc_sync on pe #25504 Message: Input/Output error

- **It was clear that it was a system call failure from the code**

```
*err = fsync(fp->fd);
if (*err) {
  sprintf(buff, "File sync error on pe #%d. Message: %s\n", fp->mype, strerror(errno);
  elen = *len_errmsg;
  FC_FUNC(cio_ctof_str,CIO_CTOF_STR)(errmsg,buff,elen);
}
```

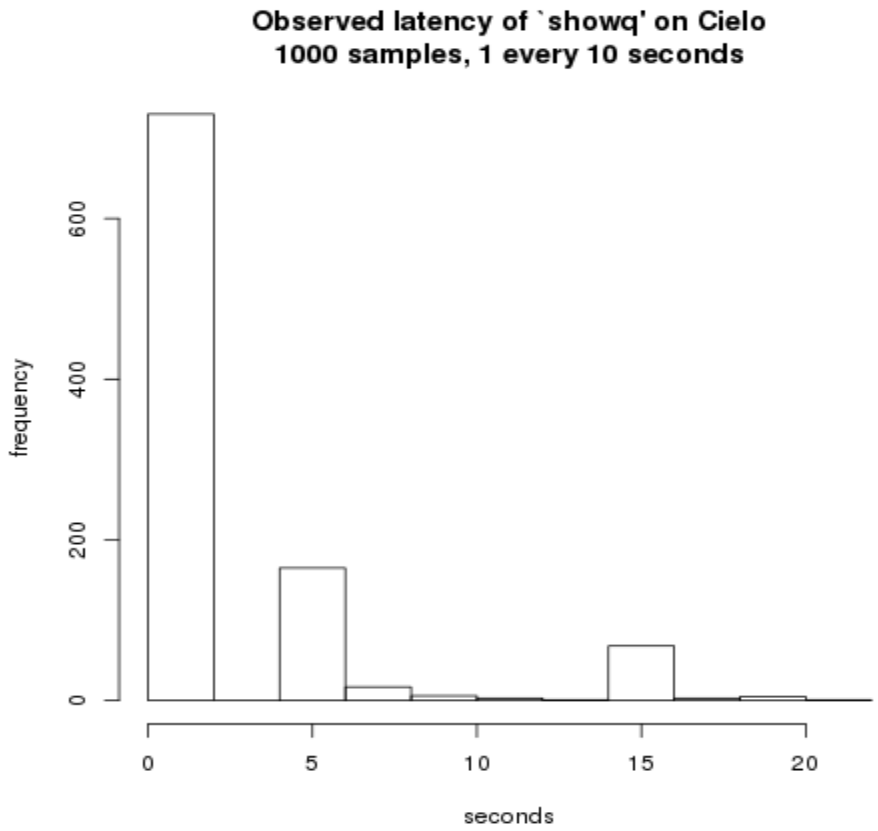
- **Cornell Wright installed and tested a patch**
  - Was there justification from Cray? Or just magic – or a shot in the dark?
  - Source code patches we can judge on their merit.

Wish list from a developer's perspective [Operational optimization]

- **TotalView at >16k PEs has hours of setup time.**
- **Cray's Abnormal Termination Processing writes *"ATP has been tested to 10,000 cores. Behavior at core counts greater than 10,000 is still being researched."***
  - Where does that leave us with 14 times that number of PEs?
- **Having to use tricks to run on a backend is an obstacle.**
  - ssh to allocated nodes – given the issues above.
- **Having to guess how proprietary mpich issue arises without access to the backend or Cray mpich source code makes things hard.**
  - For any other LANL system I can grep through the OpenMPI source code and rationalize what is happening, or call up LANL experts for help.
- **Need more available instrumentation for nodes and interconnect.**

Moab [Operational optimization]

- **20 second latencies just to see what is running?**



What Moab says it will do and what it does are two different things. [Operational optimization]

- **Jobs that bounce between eligible and blocked/deferred status.**
- **No backfill in this interval! System stalled except for prior jobs.**
- `showres` **said it was to start to run at 9pm. Still waiting at 6am.**

```
ReservationID  Type S Start End Duration N/P StartTime

484870  Job I  00:00:37 1:00:00:37 1:00:00:00 4098/65568 Wed Sep  5 06:37:31


1 reservation located
```

```
$ showq –u mdaniels
active jobs------------------------
JOBID                  USERNAME  STATE PROCS REMAINING   STARTTIME

0 active jobs           0 of 142224 processors in use by local jobs (0.00%)
                        4188 of 8903 nodes active  (47.04%)

eligible jobs----------------------
JOBID                  USERNAME  STATE PROCS REMAINING   STARTTIME
484869                   mdaniels   Idle 65536 1:00:00:00   Tue Sep   4 23:10:48
484870                   mdaniels   Idle 65568 1:00:00:00   Tue Sep   4 23:14:50

2 eligible jobs

blocked jobs-----------------------
JOBID                  USERNAME  STATE PROCS REMAINING   STARTTIME

0 blocked jobs

Total jobs:  2
```

Policy issues [Operational optimization]

- **Input from other users:**
  - Purging policy is too fast relative to HPSS rates
  - High value, very large files can be very slow to copy away
  - Only purge past a high threshold limit, like 80%.
  - Do not purge small files, which can be very valuable
    - input decks
    - program source code
- **System administrator on-call escalation**
  - Difficult to explain system issues to operators.
    - Difficult to get them to reliably reach an on-call person
  - It seems that operators could use some additional system administration skills (or latitude to apply what they know), or rote procedures for restarting Moab, etc.
  - Escalation is important for users than need interactive sessions, and are trying to avoid disrupting the system by not opting for DATs.

Summary

- Remarkable progress on I/O scratch performance
- Running production per usage model (50% of the system per job)
- Opportunities for ongoing evolutionary improvements to configuration, and system software stack
- Hardcore production users and developers care how the systems are managed. Keep them in the loop.