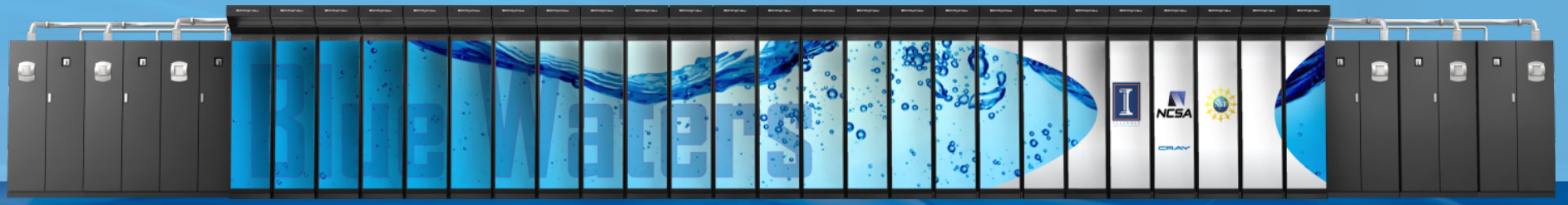
BLUE WATERS

SUSTAINED PETASCALE COMPUTING

Supporting Containers on Blue Waters

Greg Bauer, Hon Wai Leong, Roland Haas and other Blue Waters staff



NCSA NSF GREAT LAKES CONSORTIUM FOR PETASCALE COMPUTATION CRAY

Outline

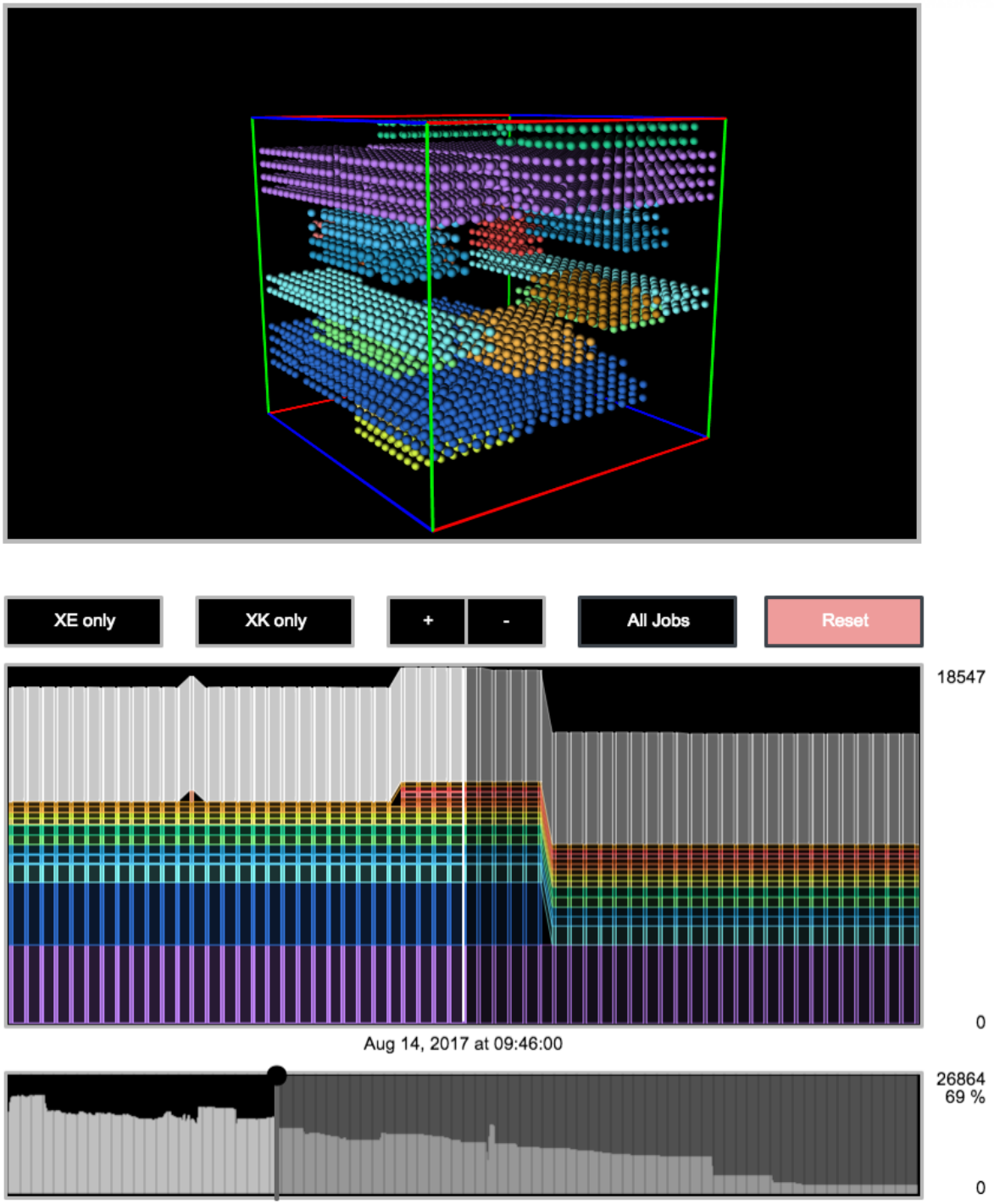
- Blue Waters: what is it, anyway?
- Do we need containers?
- Containers in HPC.
- Who is using containers on Blue Waters?

Blue Waters

- **Largest** academic **supercomputer**
  - **22,640 CPU** nodes with 2 x AMD Interlagos processors and 32 *cores* per node
  - **4,228 GPU** nodes with 1 x AMD Interlagos processor + 1 x NVIDIA K20x GPU.
  - 3D torus Cray Gemini interconnect with **9.6 GB/s** per node. Use topology aware scheduling.
  - **26 PB** of Lustre parallel file system storage with **1 TB/s**

https://bluewaters.ncsa.illinois.edu/blue-waters

- Largest jobs shown.
- Shows range of jobs in ribbon.
- Bottom shows jobs with future reservations.



Blue Waters continued

- No local disk on compute nodes.
- Most nodes have 64 GB (CPU) and 32 GB (GPU) RAM per node. 96 nodes with 2x memory.
  - IOP intensive workloads need to use /tmp (*tmpfs*) or /dev/shmem.
- No ssh access to or between compute nodes unless using a cluster compatibility mode (CCM).
- CCM has some scalability limits and memory consumption.

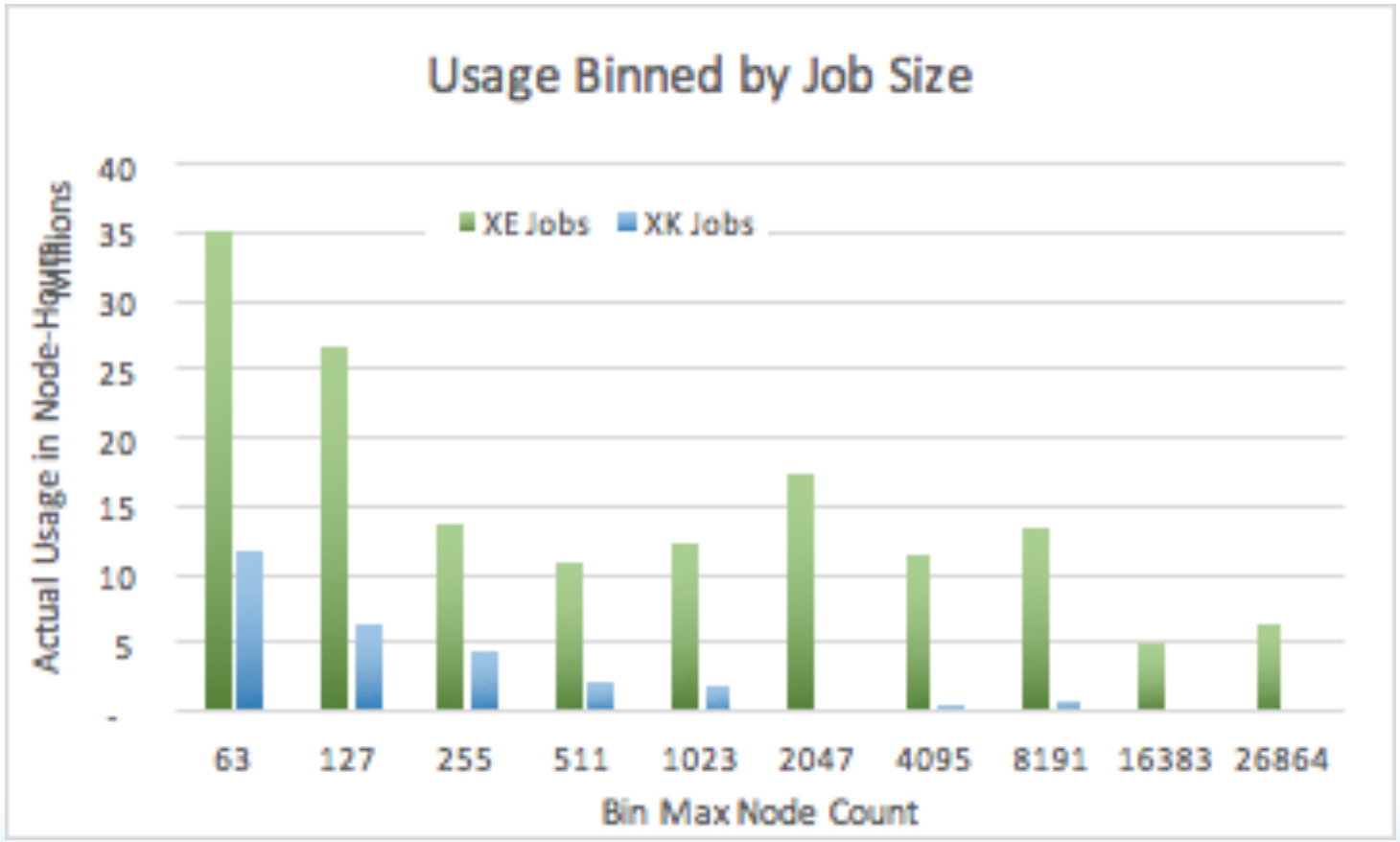
Blue Waters continued

- Blue Waters OS *frozen* at SLES11SP3
  - Linux kernel 3.0
  - GCC 4.3.4
  - glibc 2.11.3
  - python 2.6.8
- Software stack realized via Modules
  - GCC 4.4 - 6.3 along with Intel, PGI, Cray compilers.
  - python 2.6 - 3.5

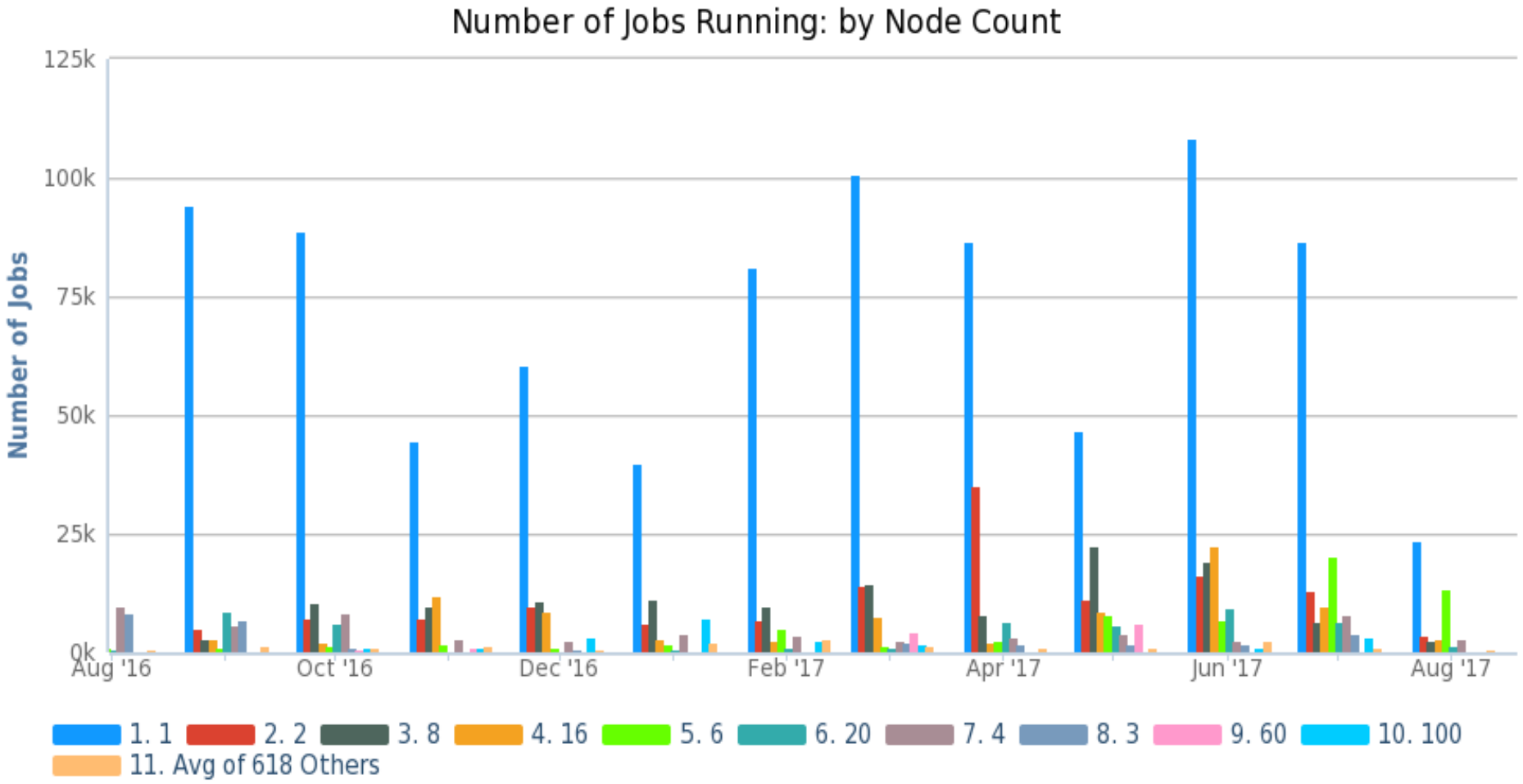
Why containers?

- Keep system current.
- Be able to accommodate new workloads that can help use Blue Waters efficiently.
- For example ArcticDEM workload
  - 300 TB of satellite imagery processed to produce elevation models.
  - 10s of millions of “jobs”.
  - Used swift workflow to bundle, launch and monitor.

Blue Waters usage by job node count



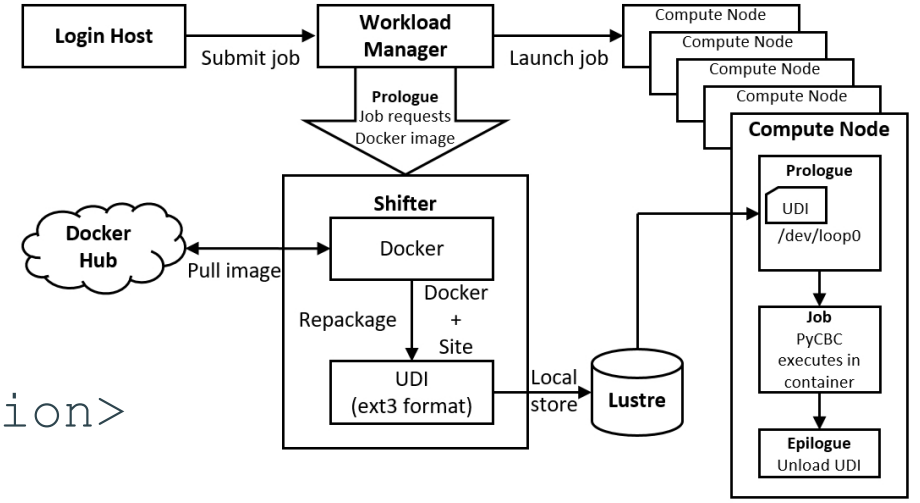
Blue Waters usage continued



Containers in HPC

- **Modules** do not solve all the problems…
  - **Old** or **complex** software stacks
- **Containers**
  - Use **any** software stack (**full control** over environment)
  - Develop on a laptop, deploy on a cluster
  - **Reproducible science**
  - **Shifter** (NERSC), **Singularity** (LBL)

Workflow of Shifter job



- Job submission requests for "shifter" resource.

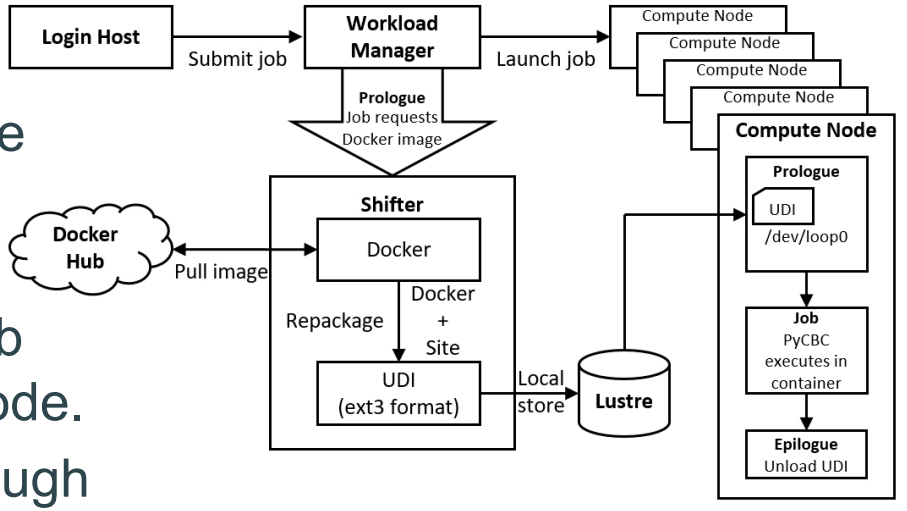
```
$ qsub -l nodes=4:ppn=32
  -l gres=shifter
  -v UDI=<dockerimg>:<version>
  shifter.job.sh
```

- Resource Manager detect "shifter" request, call Shifter prologue to pull docker image from Docker Hub, convert it to UDI and load it into compute nodes.
- Call from job script directly, without "gres=shifter" request.

```
$ aprun -b -- shifter --image=docker:osimage:version
  -- program args
```

- Internet access to Docker Hub.

Workflow of Shifter job



- Epilogue script to clean up compute nodes after job ends.
- Risk of improper cleanup due to timeout, and may affect the next job scheduled to the same compute node.
- Direct "SSH" to compute node through Shifter "port".

```
$ ssh -F ~/.shifter/config nid00002
```

- Shared storage space to store User-Defined Images (UDI) files.
- UDI files owned by 'root'. Could be exploited by users to download all kind of images from Docker Hub that fill up disk space.
- All UDI files accessible to all users, no access control.

Blue Waters continued

- Two-factor user authentication using RSA.
- Support Globus based methods like gsissh, globus-url-copy, myproxy.
- Prefer use of Globus Online for data transfer.
- Compute nodes can initiate connection to outside via RSIP/NAT. Finite number of ports out.

Users of containers on Blue Waters

- LSST/DES (previous talk)
  - HTCondor
- LIGO (next talk)
  - OSG
  - Pegasus and Condor
- ATLAS - OSG All Hands Meeting presentation
  - OSG
  - CVMFS – limited by lack of *fuse*

HPC uses of containers

- Support MPI model for communication between processes running on multiple containers
  - native MPI possible when using MPICH2 API
  - up to 800 node jobs (9000 in future release)
- Bundle code and dynamic libraries in image
  - reduce load on metadata server, speed up application startup time