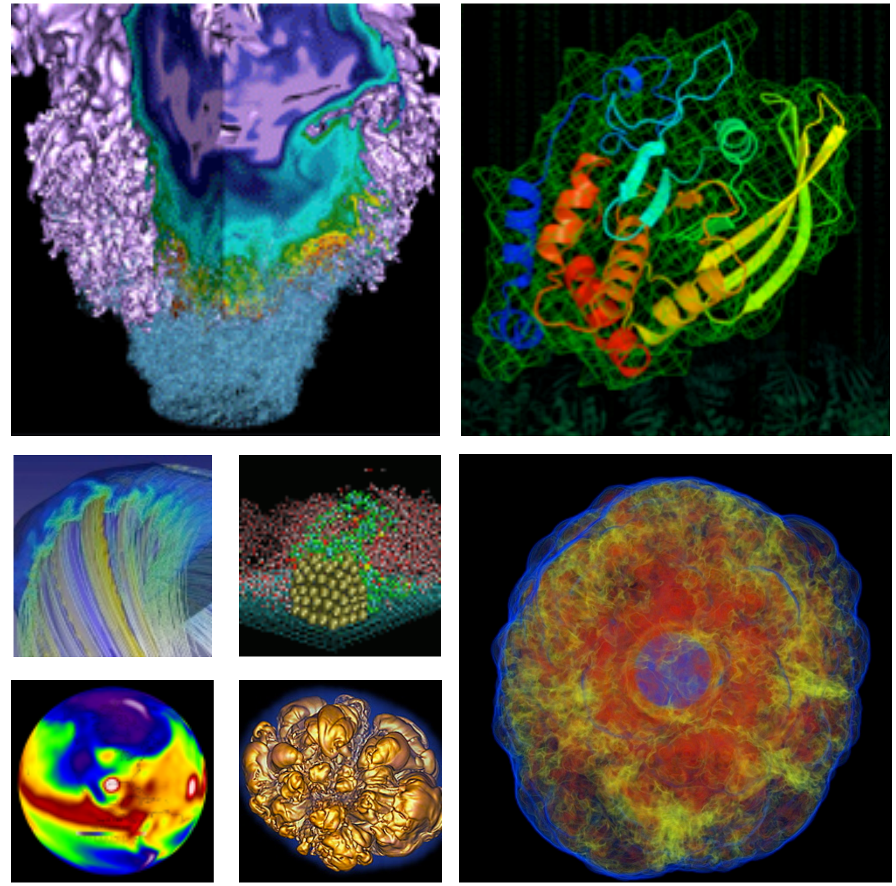
Contain This, Unleashing Docker for HPC



NERSC

Doug Jacobsen & Shane Canon

Cray User Group 2015 April 29, 2015

U.S. DEPARTMENT OF ENERGY | Office of Science

BERKELEY LAB
Lawrence Berkeley National Laboratory

Acknowledgements

- **Larry Pezzaglia (Former NERSC, UDI COE, CHOS)**
- **Scott Burrow (NERSC, Jesup/MyDock)**
- **Shreyas Cholia (NERSC, MyDock)**
- **Dani Conner (Cray, UDI COE)**
- **Martha Dumler (Cray, UDI COE)**
- **Dave Henseler (Cray, UDI COE)**
- **Dean Roe (Cray, UDI COE)**
- **Kitrick Sheets (Cray, UDI COE)**
- **Michael Barton (JGI, Example Docker Image)**

This work was supported by the Director, Office of Science, Office of Advanced Scientific Computing Research of the U.S. Department of Energy under Contract No. **DE-AC02-05CH11231**.

Convergence of Disruptive Technology

- **Increasing Role of Data**

- **Converging HPC and Data Platforms**

- **New Models of Application Development and Delivery**

docker

U.S. DEPARTMENT OF ENERGY | Office of Science

BERKELEY LAB

DOE Facilities Require Exascale Computing and Data

NERSC

Astronomy

Particle Physics

Chemistry and Materials

Genomics

Fusion

Petascale to Exascale

- **Petabyte data sets today, many growing exponentially**
- **Processing requirements grow super-linearly**
- **Need to move entire DOE workload to Exascale**

Popular features of a data intensive system and supporting them on Cori

Data Intensive Workload Need	Cori Solution
Local Disk	NVRAM 'burst buffer'
Large memory nodes	128 GB/node on Haswell; Option to purchase fat (1TB) login node
Massive serial jobs	NERSC serial queue prototype on Edison; MAMU
Complex workflows	***User Defined Images*** CCM mode Large Capacity of interactive resources
Communicate with databases from compute nodes	Advanced Compute Gateway Node
Stream Data from observational facilities	Advanced Compute Gateway Node
Easy to customize environment	***User Defined Images***
Policy Flexibility	Improvements coming with Cori: Rolling upgrades, CCM, MAMU, above COEs would also contribute

User-Defined Images

- **User-Defined Images (UDI): A software framework which enables users to accompany applications with portable, customized OS environments**
  - e.g., include ubuntu base system with Application built for ubuntu (or debian, centos, etc)
- **A UDI framework would:**
  - Enable the HPC Platforms to run more applications
  - Increase flexibility for users
  - Facilitate reproducible results
  - Provide rich, portable environments without bloating the base system

Use Cases

- **Large high energy physics collaborations (e.g., ATLAS and STAR) requiring validated software stacks**
  - Some collaborations will not accept results from non-validated stacks
  - Simultaneously satisfying compatibility constraints for multiple projects is difficult
  - Solution: create images with certified stacks
- **Bioinformatics and cosmology applications with many third-party dependencies**
  - Installing and maintaining these dependencies is difficult
  - Solution: create images with dependencies
- **Seamless transition from desktop to supercomputer**
  - Users desire consistent environments
  - Solution: create an image and transfer it among machines

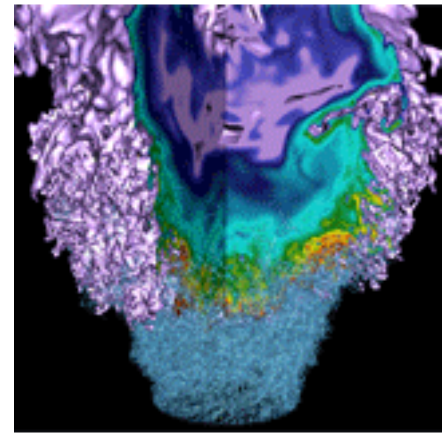
Value Proposition for UDI for Cori

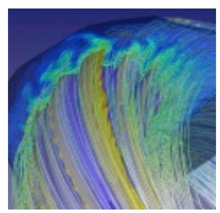
- **Expanded application support**
  - Many applications currently relegated to commodity clusters could run on the Cray through UDI
  - This will help position Cori as a viable platform for data-intensive
- **Easier application porting**
  - The need for applications to be “ported” to Cori will be reduced
  - More applications can work “out of the box”
- **Better Reproducibility**
  - Easier to re-instantiate an older environment for reproducing results

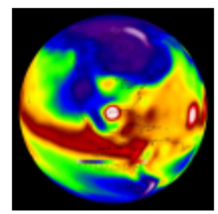
UDI Design Goals for Cori

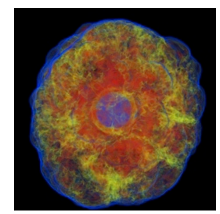
- **User independence: Require no administrator assistance to launch an application inside an image**
- **Shared resource availability (e.g., PFS/DVS mounts and network interfaces)**
- **Leverages or integrates with public image repos (i.e. DockerHub)**
- **Seamless user experience**
- **Robust and secure implementation**

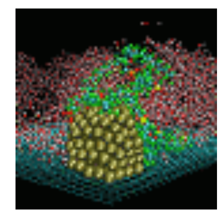
Background











Implementation Options

Multiple UDI implementations are available

- **CHOS: a custom solution developed at NERSC**
  - 8+ years' use on NERSC's production clusters
  - Demonstrates feasibility, low overhead, and seamless user experience
  - Lacks wide adoption
- **VMs**
  - Need to manage hypervisors
  - Overhead of VM
  - Hard/Impossible to integrate with file systems and interconnect
- **Docker: a solution implemented through Linux namespaces and cgroups**
  - Widespread community momentum and adoption
  - All kernel requirements have been mainlined
  - Potential cluster/HPC deployment issues
- **Hybrid**
  - Customized run time environment that integrates with CLE
  - Able to draw from and integrate with the Docker image ecosystem

Performance/Resource Consumption

- **Multiple Studies comparing Containers/Docker to VMs**
- **Containers typically approach bare-metal performance**
- **Containers launch much faster (10x-1000x)**
- **Containers require less memory since it typically launches a single or limited number of processes**

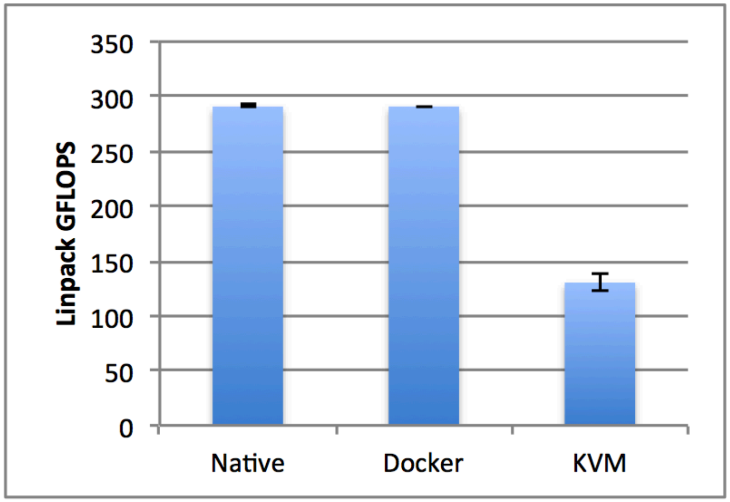
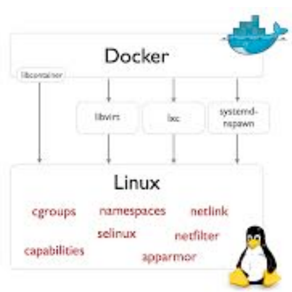


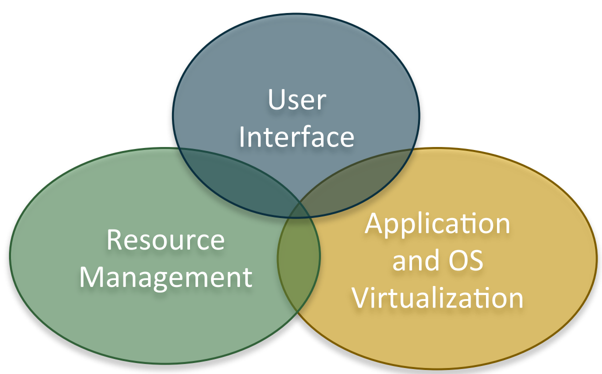
Figure 1. Linpack performance on two sockets (16 cores). Each data point is the arithmetic mean obtained from ten runs. Error bars indicate the standard deviation obtained over all runs.

Source: IBM Research Report (RC25482)

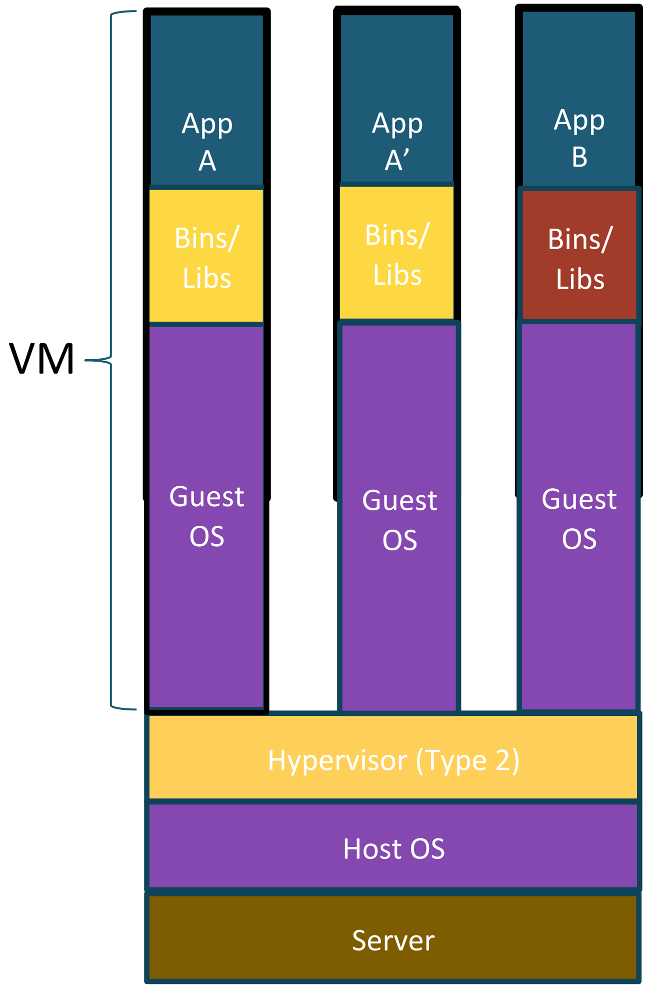
Docker at a High Level

- **Process Container: Uses Linux kernel features (cgroups and namespace) to create semi-isolated “containers”**
- **Image Management: Version control style image management front-end and image building interface**
- **Ecosystem: Can push/pull images from a community-oriented image hub (i.e. DockerHub)**

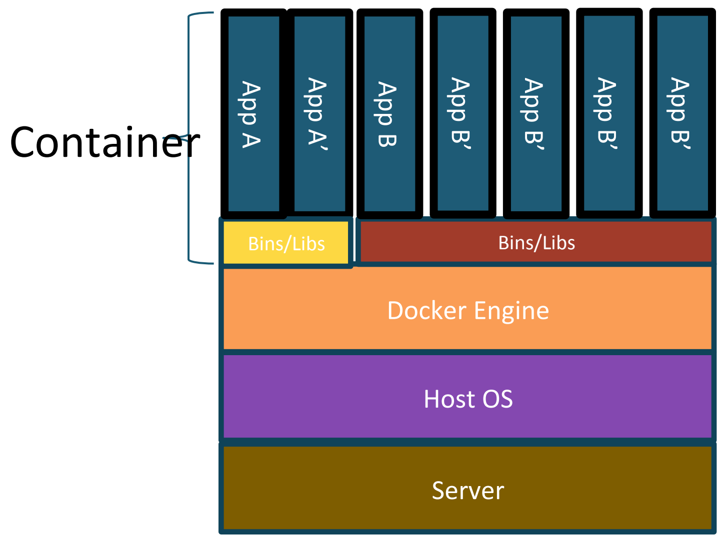




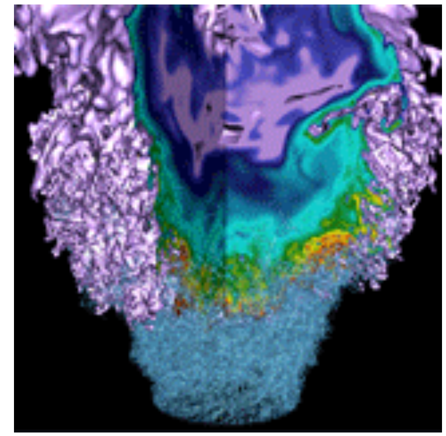
Containers vs. VMs

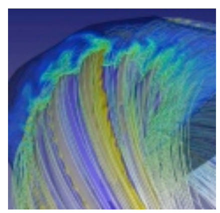


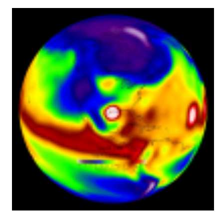
Containers are isolated,
but share kernel and, where
appropriate, bins/libraries

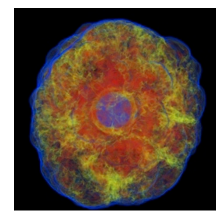


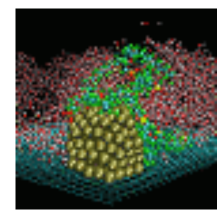
Docker@NERSC











“MyDock”

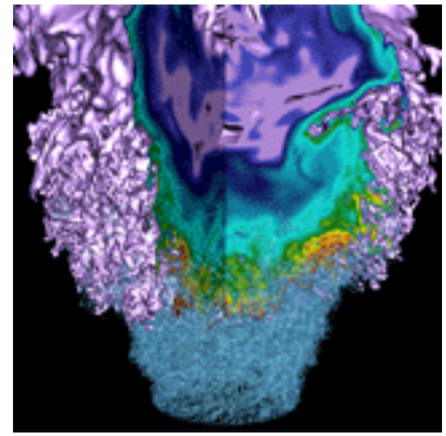
- **Goal: Allow Users to run their own images using Docker with access to shared resources (i.e. file systems, network)**
- **Challenges:**
  - Security Risk: Escalated privileges (suid, privileged ports, devices)
- **Approach: MyDock Wrapper**
  - Limit users to running as themselves
  - Only pass-through/expose directories they own
  - Disable “privileged” mode options
  - Mount everything with nosuid, nodev

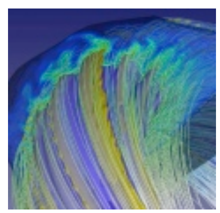
Use of MyDock by NERSC User's

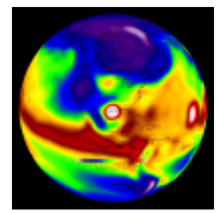
Dark Energy Survey

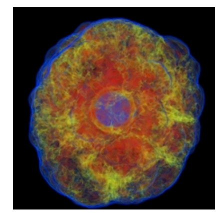
- **Image analysis from Sky Surveys**
- **Part of an SC14 Demo**
- **Used global scratch to store data**
- **Container launches a condor worker that talks to a global scheduler at U. of Illinois**
- **Used reserved nodes on Jesup Testbed Cluster**

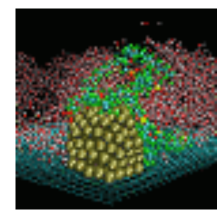
User Defined Images on the Cray Platforms











Docker on the Cray?

- **Docker assumes local disk**
  - aufs, btrfs demand shared-memory access to local disk

Approach

- Leverage Docker image and integrate with DockerHub
- Adopt alternate approach for instantiating the environment on the compute node (i.e. don't use the Docker daemon)
- Plus: Easier to integrate and support
- Minus: Can't easily leverage other parts of the Docker ecosystem (i.e. orchestration)

Image Location/Format Options

Image Location Options

- **GPFS** – No local clients, so overhead of DVS
- **Lustre** – Local clients, large scale
- **Burst Buffer** – No widely available yet
- **Local Disk** – Not available on Crays
- **iSCSI/SRP mounts** – Not readily available, worth exploring

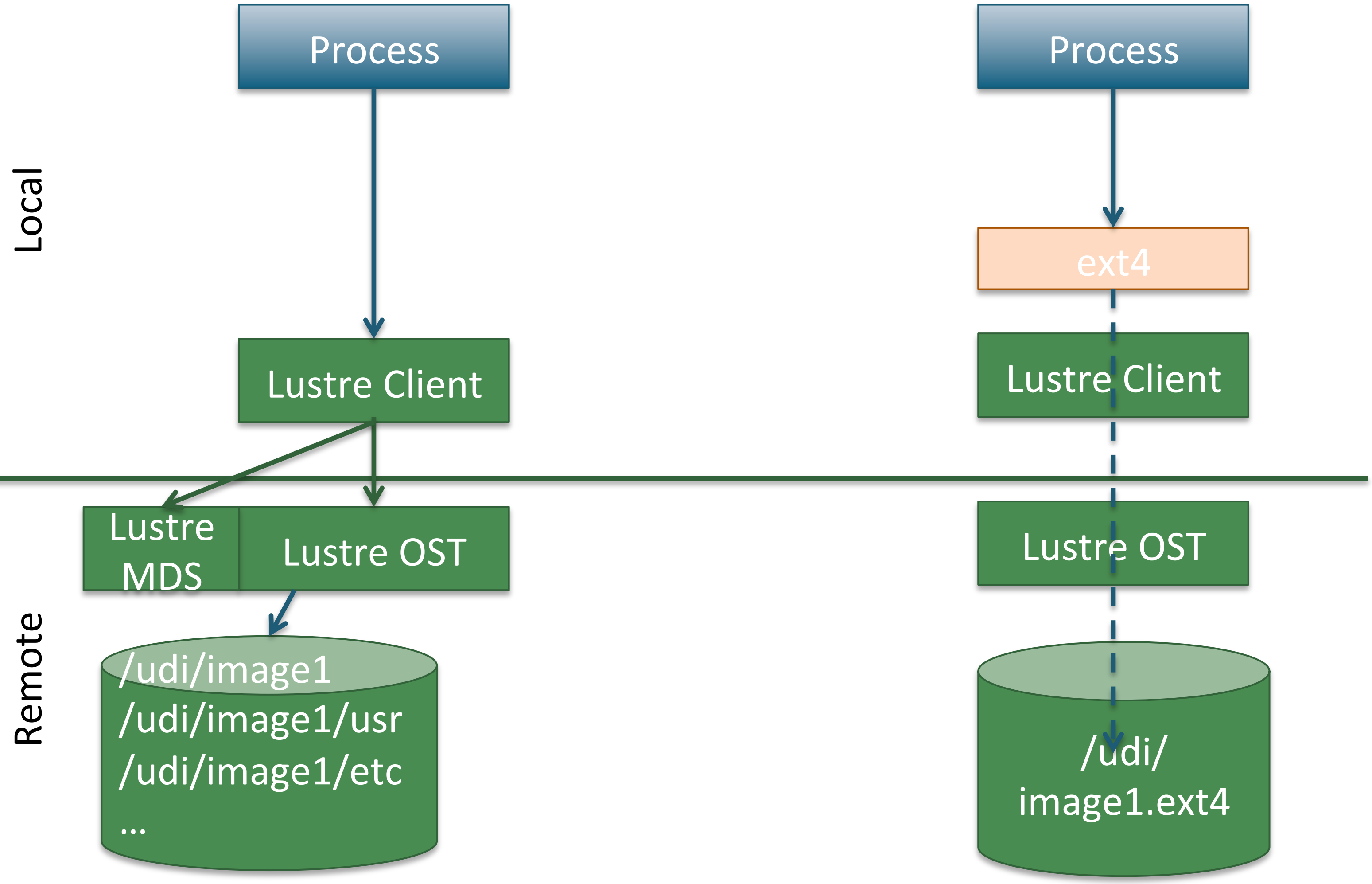
Image Format Options

- **Unpacked Trees**
  - Simple to implement
  - Metadata performance depends on metadata performance of the underlying system (i.e. Lustre or GPFS)
- **Loopback File Systems**
  - Moderate complexity
  - Keeps file system operations local

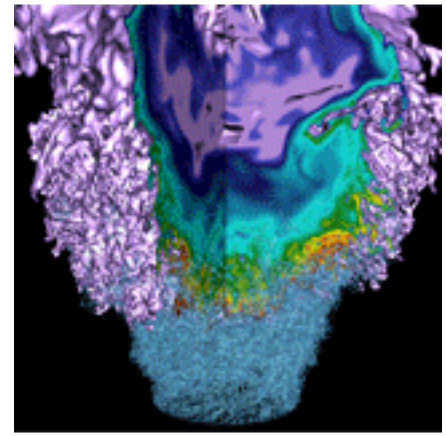
Considerations

- Scalable
- Manageable
- Metadata performance
- Bandwidth Consumption

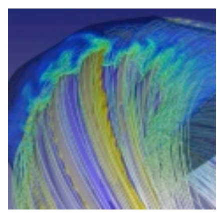
Layout Options – Stack View

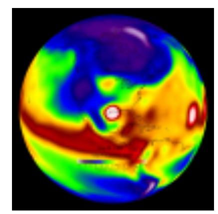
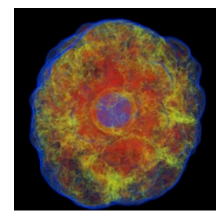
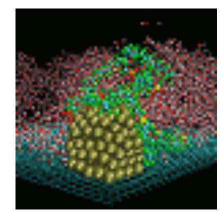


Prototype Implementation



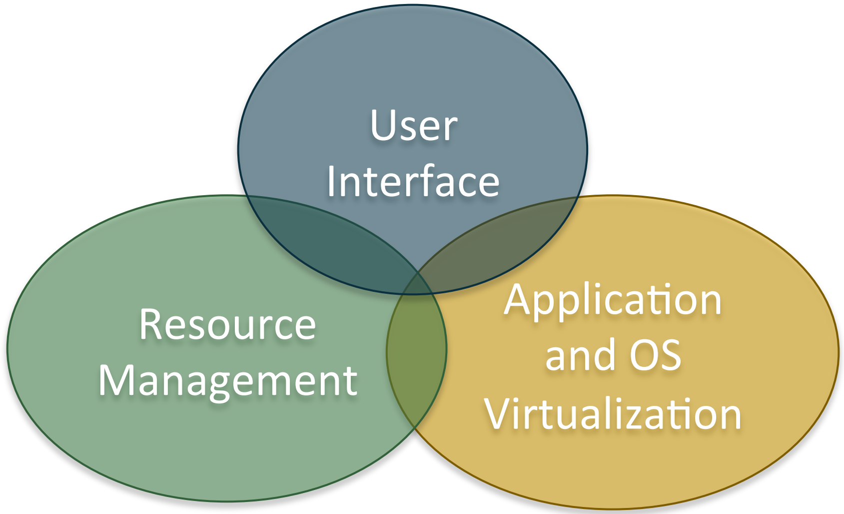
NERSC



Prototype Implementation: “Shifter”

- **Supports**
  - Docker Images
  - CHOS Images
  - Will be able to support other image types (e.g., qcow2, vmware, etc)
- **Basic Idea**
  - Convert native image format to common format
  - Construct chroot tree on compute nodes using common format image
  - Directly use linux VFS namespaces to support



Prototype Implementation: "Shifter"

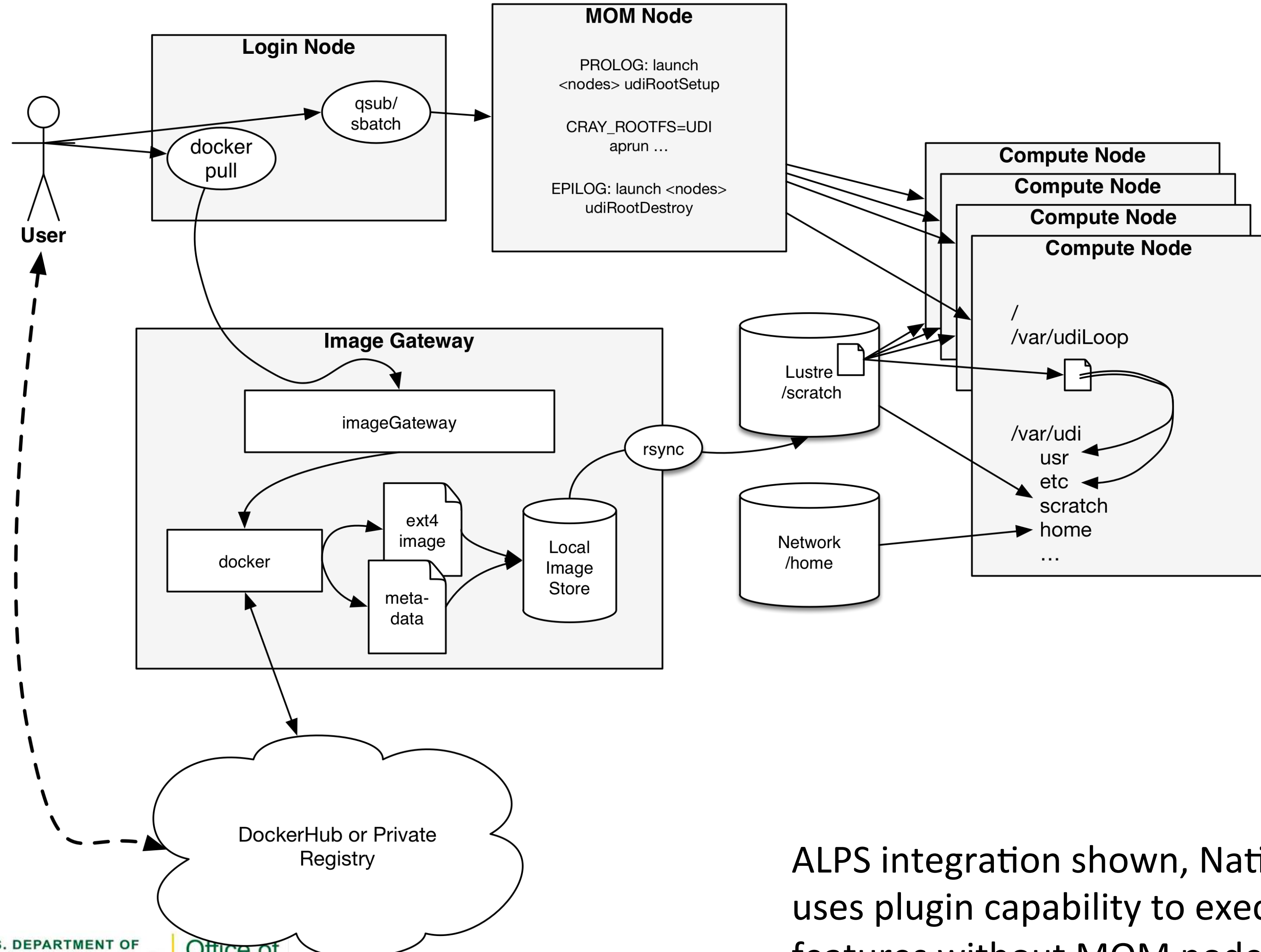
- **Command line interface to emulate some docker functionality**
  - List images
  - Pull image from DockerHub / private registry
  - Login to DockerHub
- **Central gateway service**
  - Manage images
  - Transfer images to computational resource
  - Convert images several sources to common ext4 sparse file
- **udiRoot**
  - Sets up image on compute node
  - CCM-like implementation to support internode communication
- **Workload Manager Integration**
  - Pull image at job submission time if it doesn't already exist
  - Implement user interface for user-specified volume mapping
  - Launch udiRoot on all compute nodes at job start
  - Deconstruct udiRoot on all compute nodes at job end
  - WLM provides resource management (e.g., cgroups)

User Interface

Application and OS Virtualization

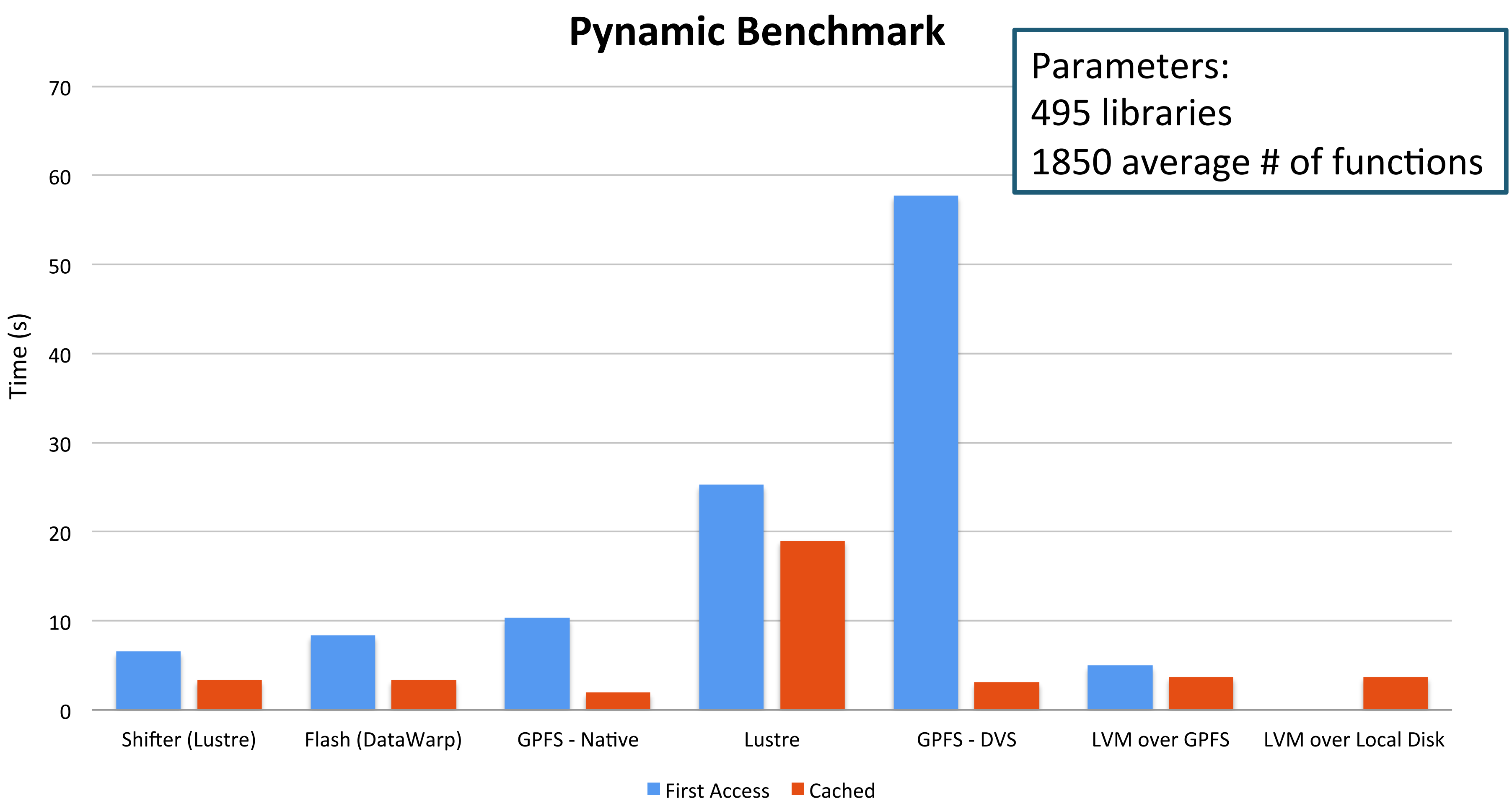
Resource Management

Prototype Implementation: “Shifter”

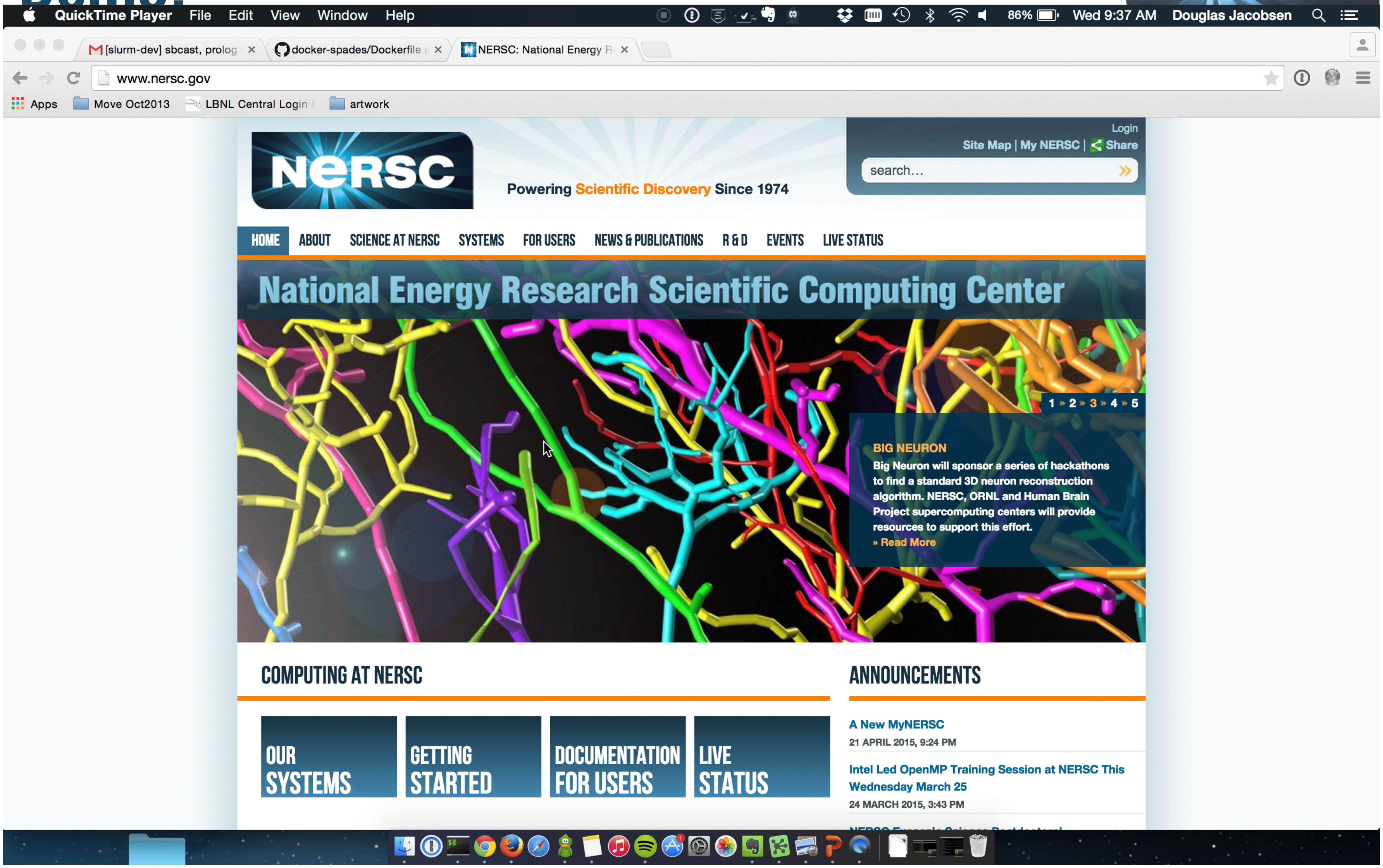


ALPS integration shown, Native Slurm uses plugin capability to execute features without MOM node

Pynamic Benchmark Results

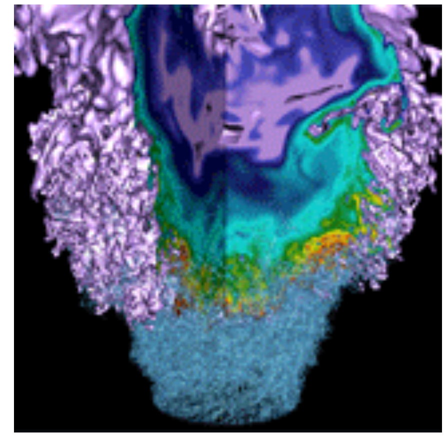


Demo!

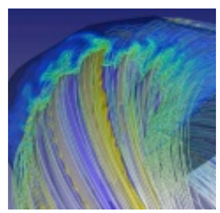


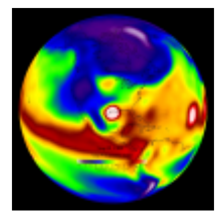
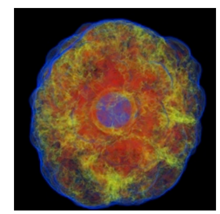
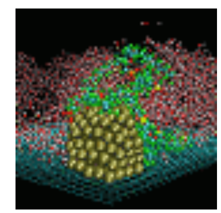
QuickTime Player File Edit View Window Help
86% Wed 9:37 AM Douglas Jacobsen
[slurm-dev] sbcast, prolog
docker-spades/Dockerfile
NERSC: National Energy R
www.nersc.gov
Apps
Move Oct2013
LBNL Central Login
artwork
Login
Site Map | My NERSC | Share
search...
NERSC
Powering Scientific Discovery Since 1974
HOME ABOUT SCIENCE AT NERSC SYSTEMS FOR USERS NEWS & PUBLICATIONS R & D EVENTS LIVE STATUS
National Energy Research Scientific Computing Center
1 » 2 » 3 » 4 » 5
BIG NEURON
Big Neuron will sponsor a series of hackathons to find a standard 3D neuron reconstruction algorithm. NERSC, ORNL and Human Brain Project supercomputing centers will provide resources to support this effort.
» Read More
COMPUTING AT NERSC
OUR SYSTEMS
GETTING STARTED
DOCUMENTATION FOR USERS
LIVE STATUS
ANNOUNCEMENTS
A New MyNERSC
21 APRIL 2015, 9:24 PM
Intel Led OpenMP Training Session at NERSC This Wednesday March 25
24 MARCH 2015, 3:43 PM

Future Work and Discussion



NERSC



Future Work

- **Integrate with other batch modes including serial queues**
- **Expand support for other image types (i.e. qcow2)**
- **Explore replacing image gateway with more standard image repo (i.e. glance from OpenStack)**
- **Create a base image for running MPI applications in a NERSC private docker registry**
  - either using Cray Libraries or RDMA compatibility

Conclusions

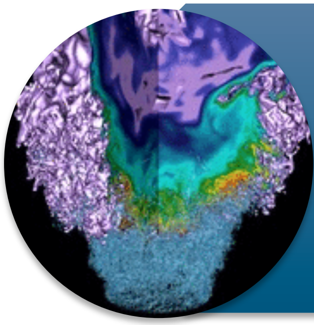
- **UDI will enable Data Intensive Workloads to get started much more quickly on cori**
  – Don't change your pipeline for the computation resource – shift the computational resource to your pipeline!
- **"Shifter" prototype implementation demonstrates that UDI is both possible and performant**
- **Shifter approach to UDI may provide additional benefits for shared-library-heavy codes**

National Energy Research Scientific Computing Center

Why Developers Care

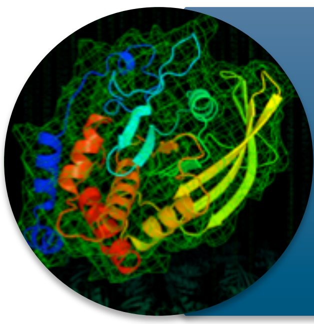
- **Build once…run anywhere**
  - A clean, safe, hygienic and portable runtime environment for your app.
  - No worries about missing dependencies, packages and other pain points during subsequent deployments.
  - Take a snapshot of a running container and restore it again when required.
  - Run each app in its own isolated container, so you can run various versions of libraries and other dependencies for each app without worrying
  - Automate testing, integration, packaging…anything you can script
  - Reduce/eliminate concerns about compatibility on different platforms, either your own or your customers.
  - Cheap, zero-penalty containers to deploy services? A VM without the overhead of a VM? Instant replay and reset of image snapshots? That's the power of Docker

More Science Requires More Computing



Science at Scale

Petascale to Exascale Simulations



Science through Volume

Thousands to Millions of Simulations

Science in Data

Petabytes to Exabytes of Data

What Data Users Like about Clusters

- Flexible Queue Policies (longer wall times, more queued jobs, serial queues)
- Access to faster single cores and larger memory
- Easier support for 3rd Party Tools
- Ease of Use and Flexibility