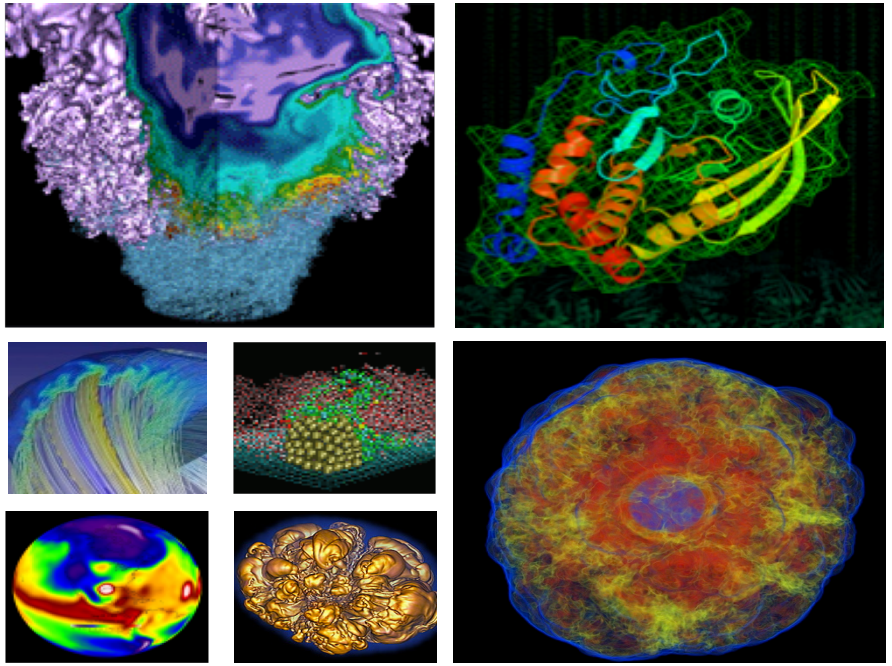
Containers at NERSC



Shane Canon
Lawrence Berkeley National Lab

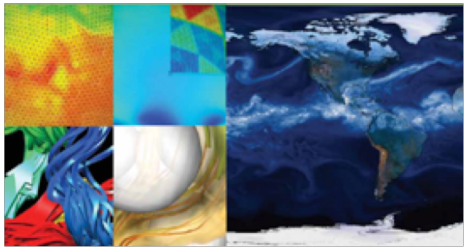
June 20, 2019

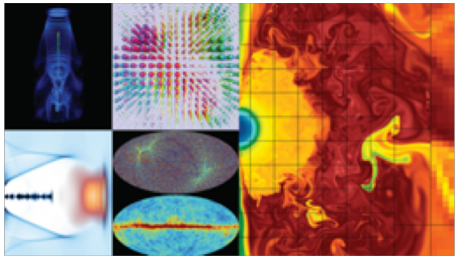
U.S. DEPARTMENT OF ENERGY | Office of Science

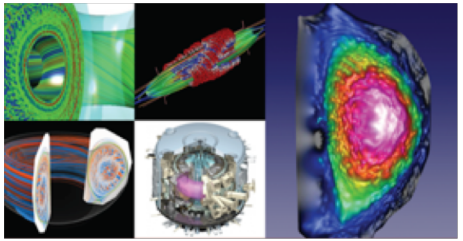
BERKELEY LAB

NERSC is the mission HPC computing center for the DOE-SC

- NERSC deploys advanced HPC and data systems for the broad Office of Science community
- NERSC staff provide advanced application and system performance expertise to users
- Approximately 7000 users and 750 projects
- Over 2000 publication resulting in NERSC resources per year
- Data Initiative: Pioneer new capabilities to enable scientists to make large-scale data-intensive science discoveries.







Supporting Data Intensive Systems and HPC

Data users and other emerging communities need the scale and unique capabilities from our HPC systems

But migrating to HPC systems can kill productivity

HPC can be Awkward

- **No local disk**
  - Breaks a lot of standard Linux work flows
- **Minimal OS**
  - Designed to accelerate parallel software
  - Many “expected” Linux tools are absent
  - Runs SUSE, and doesn’t upgrade often
- **Different file systems have different responses**
  - Sometimes unclear to users where is the best place to put their software and data
- **Many groups have turned to Shifter to over come these obstacles**

Long History with Custom Environments

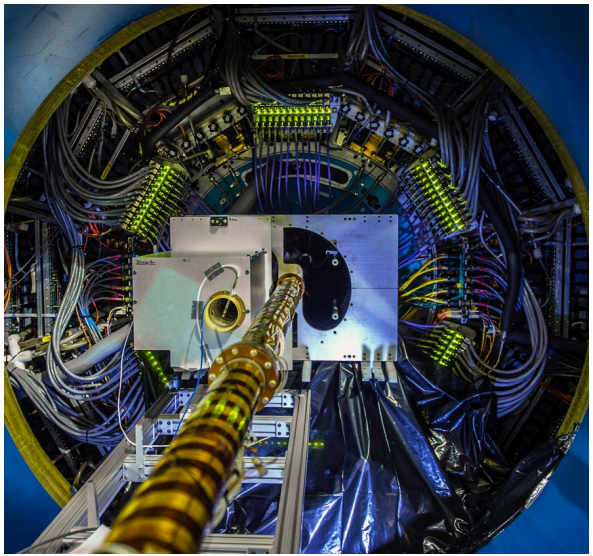
- **~2003: ChrootOS(CHOS) – System to enable projects to have customized environments on a shared cluster (PDSF). Integrated with login and batch and used a custom kernel module to provide a seamless experience.**
- **~2014: MyDock – Thin wrapper around Docker to allow users to securely run Docker on a shared cluster (Jesup/Carver).**
- **~2015: Shifter: "User Defined Environments"**

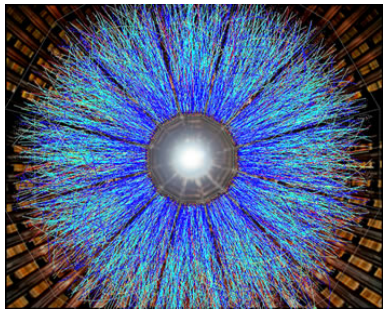
Containers and Science

- **Productivity**
  - Pick the OS that works best for your app and use the system package manager to install dependencies.
- **Reusability and Collaboration**
  - Share images across a project to avoid rebuilds and avoid mistakes
- **Reproducibility**
  - Everything you need to redo a scientific analysis can be in the image (apps, libraries, environment setup, scripts)
- **Portability**
  - Can easily run on different resources (of the same architecture)

Probing The Nucleus

- **STAR at Brookhaven, NY**
  - smashing nuclei into each other to understand their component parts
- **Data analysis and simulation**
- **Why Shifter?**
  - Difficult software dependencies (32-bit libraries)





How'd They Do It?

```
# Build STAR environment image from tarballs
FROM ringo/scientific:6.4
MAINTAINER Mustafa Mustafa <mmustafa@lbl.gov>

# RPMs
RUN yum -y install libxml2 tcsh libXpm.i686 libc.i686 libXext.i686 libXrender.i
686 libstdc++.i686 fontconfig.i686 zlib.i686 libgfortran.i686 libSM.i686 mysql-
libs.i686 gcc-c++ gcc-gfortran glibc-devel.i686 xorg-x11-xauth wget make libxml
2.so.2 gdb libXtst.{i686,x86_64} libXt.{i686,x86_64} glibc glibc-devel gcc-c++

# Dev Tools
RUN wget -O /etc/yum.repos.d/slc6-devtoolset.repo http://linuxsoft.cern.ch/cern
/devtoolset/slc6-devtoolset.repo && \
    yum -y install devtoolset-2-toolchain
COPY enable_scl /usr/local/star/group/templates/

# untar STAR OPT
COPY optstar.sl64_gcc482.tar.gz /opt/star/
COPY installstar /
RUN python installstar SL16c && \
     rm -f installstar && \
     rm -f optstar.sl64_gcc482.tar.gz

# untar ROOT
COPY rootdeb-5.34.30.sl64_gcc482.tar.gz /usr/local/star/
COPY installstar /
RUN python installstar SL16c && \
     rm -f installstar && \
     rm -f rootdeb-5.34.30.sl64_gcc482.tar.gz

# DB load balancer
COPY dbLoadBalancerLocalConfig_generic.xml /usr/local/star/packages/SL16d/StDb/
servers/

# production pipeline utility macros
COPY Hadd.C /usr/local/star/packages/SL16d/StRoot/macros/
COPY lMuDst.C /usr/local/star/packages/SL16d/StRoot/macros/
COPY checkProduction.C /usr/local/star//packages/SL16d/StRoot/macros/
```

Publically available SL6.4 image

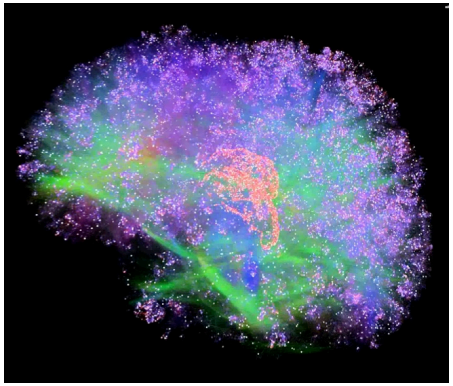
Custom STAR software:
Compiled on **another** system

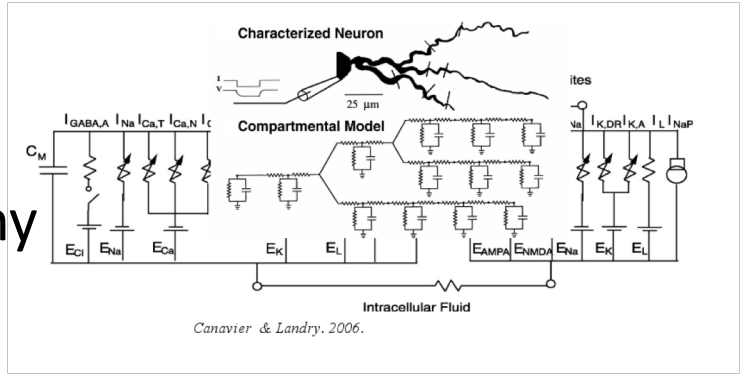
Leveraging Shifter for Easy Scalability

- **Shifter has capability to loop mount an xfs file**
  - Backed by Lustre, but all metadata actions are limited to a single node, so access is very fast
- **STAR needs to read from a ~100 MB MySQL database**
  - Running 32 individual jobs / node
- **DB on Lustre, query timed out after 30 minutes**
- **Copied DB to Shifter's xfs**
  - Initial copy ~5 minutes
  - DB Query was instantaneous
- **Used this functionality to quickly scale up without re-engineering their workflow**

Modeling the Mind

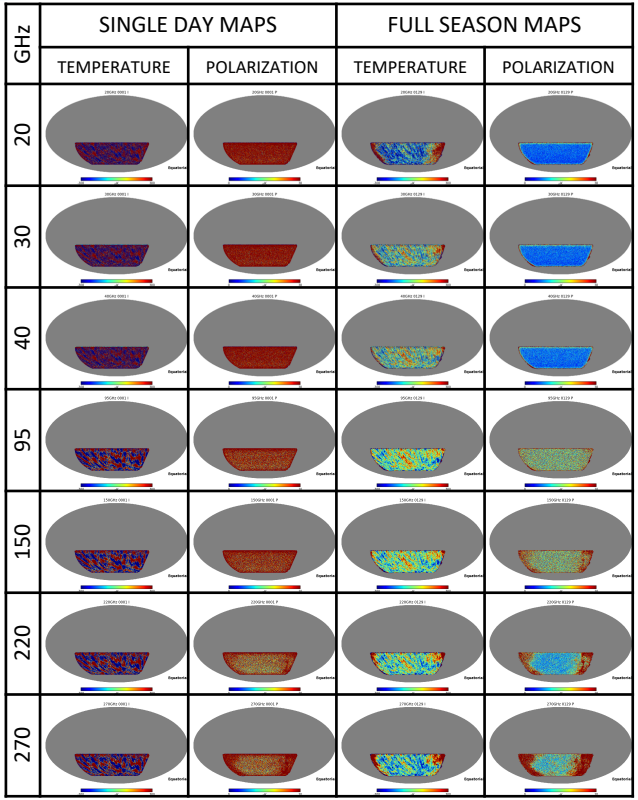
- **Neuron**
  - Simulation program to model neuron response to stimuli
- **Simulation:** 3000 points by 4000 time steps for 400 neurons
- **Why Shifter?**
  - Software developed in 1985, many older dependencies





CMB Simulation At Scale With TOAST (& Shifter)

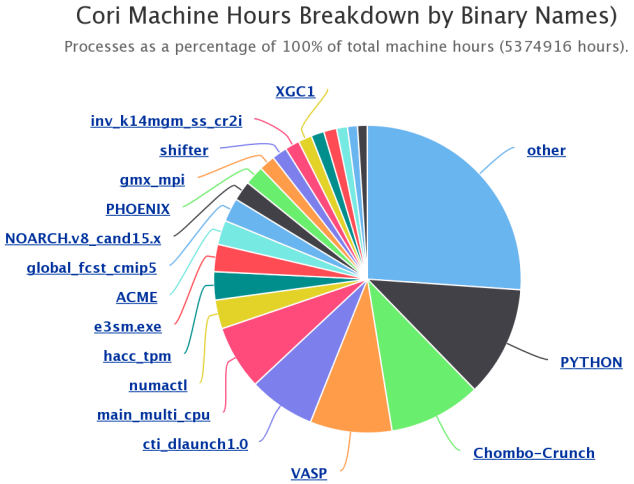
- **Simulate & map 50,000 detectors gathering 30X Planck mission data**
- **Using all of NERSC's Cori-2 supercomputer**
  - KNL optimization including Cray/Intel "dungeon session"
  - **Docker/Shifter to distribute python-wrapped code to 600,000 cores.**
- **Including realistic atmosphere simulation, instrument noise & sky signal.**
- **Re-using the expensive simulation in situ to allow multiple reduction pipelines.**
- **Meets Y1 stretch goals for LBNL LDRD project: Simulation & Analysis of CMB-S4 on Xeon Phi Systems.**
- **Animations at http://crd.lbl.gov/cmb-sims**

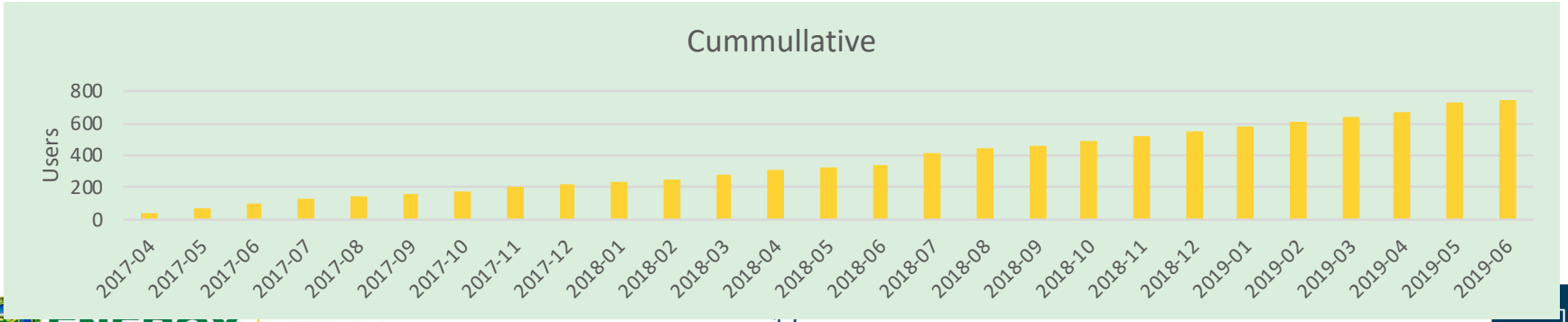


J. Borrill, R. Keskitalo & T. Kisner

Stats on Shifter Use at NERSC

- 4M Image Lookups
- 728 Unique Image tags
- 348 Unique Users
- Still a small fraction of NERSC overall use





Supporting "Edge Services" for Science

Spin* is a system to **build and deploy science gateways, workflow managers, databases, and other "edge services"** using Docker containers.

It is designed to be **flexible, scalable, and integrated tightly** with NERSC resources:

- Develop on your laptop; **deploy in minutes**.
- **Scale out** for performance.
- Access **HPC networks and file systems**.
- NERSC manages everything under the hood.

* Scalable Platform Infrastructure at NERSC

ECP SuperContainers Project

- **Led by Andrew Younge (Sandia)**
- **Focus on:**
  - Establishing Best Practices
  - Enabling Portability across Centers and Container Runtimes
  - Integrating with CI and Deployment Models
  - Training Scientists

CANOPIE-HPC Workshop

Home | Introduction | Call for Papers | Topics | Organizers | Program

CALL FOR PAPERS

Call for Papers

1st Workshop on Containers and New Orchestration Paradigms for Isolated Environments in HPC (CANOPIE HPC)

Held in conjunction with SC19: The International Conference on High Performance Computing, Networking, Storage, and Analysis, Nov 17th-22nd, 2019

Website: http://canopie-hpc.org

Submission Deadline: September 2nd, 2019

Acceptance Notice: October 2nd, 2019

Final papers due: October 7th, 2019

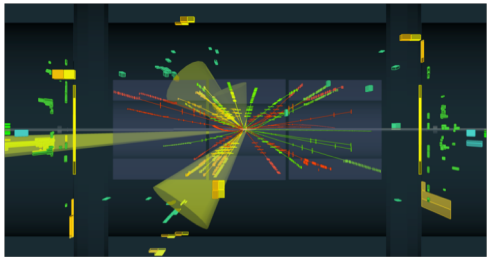
Questions

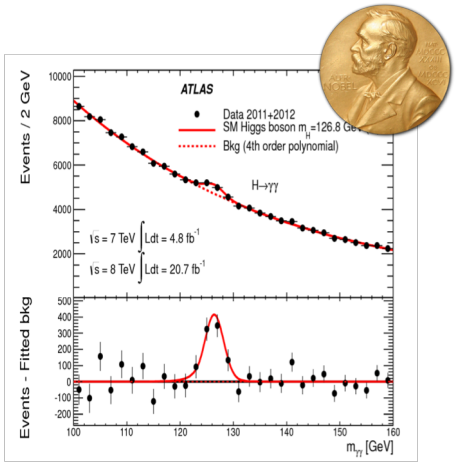
NERSC Resources

- **Cori Cray XC40**
  - Data-intensive (32-core Haswells, 128GB) partition
  - Compute-intensive (68-core KNLs, 90GB) partition
  - ~10k nodes, ~700k cores
  - High Speed Aires interconnect8 GB/s MPI bandwidth
- **High speed parallel file systems**
  - >10 PB project file system (GPFS)
  - >28 PB scratch file system (Lustre)
  - >1.5 PB Burst Buffer (flash)
- **Archive Storage**
  - 200 PB Tape Archive
- **Perlmutter – Cray Shasta (2020)**

Probing the Fundamentals of Matter

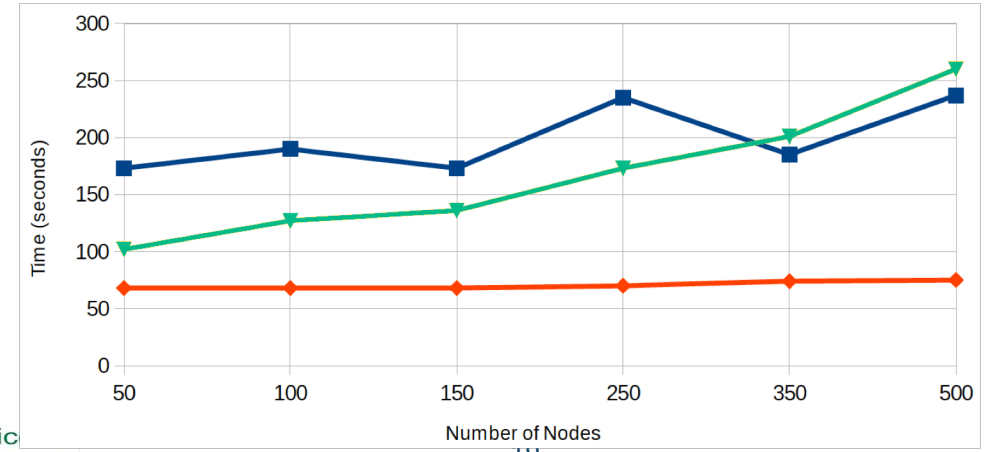
- **Large Hadron Collider (LHC)**
  - 300 trillion proton-proton collisions and 30 PBs of data per year.
- **Data analysis, simulation, multi-site data and computing pool**
- **Why Shifter?**
  - Complicated software stack: Needs FUSE and elevated permissions to run
  - Integrated framework for running with images at all computing sites



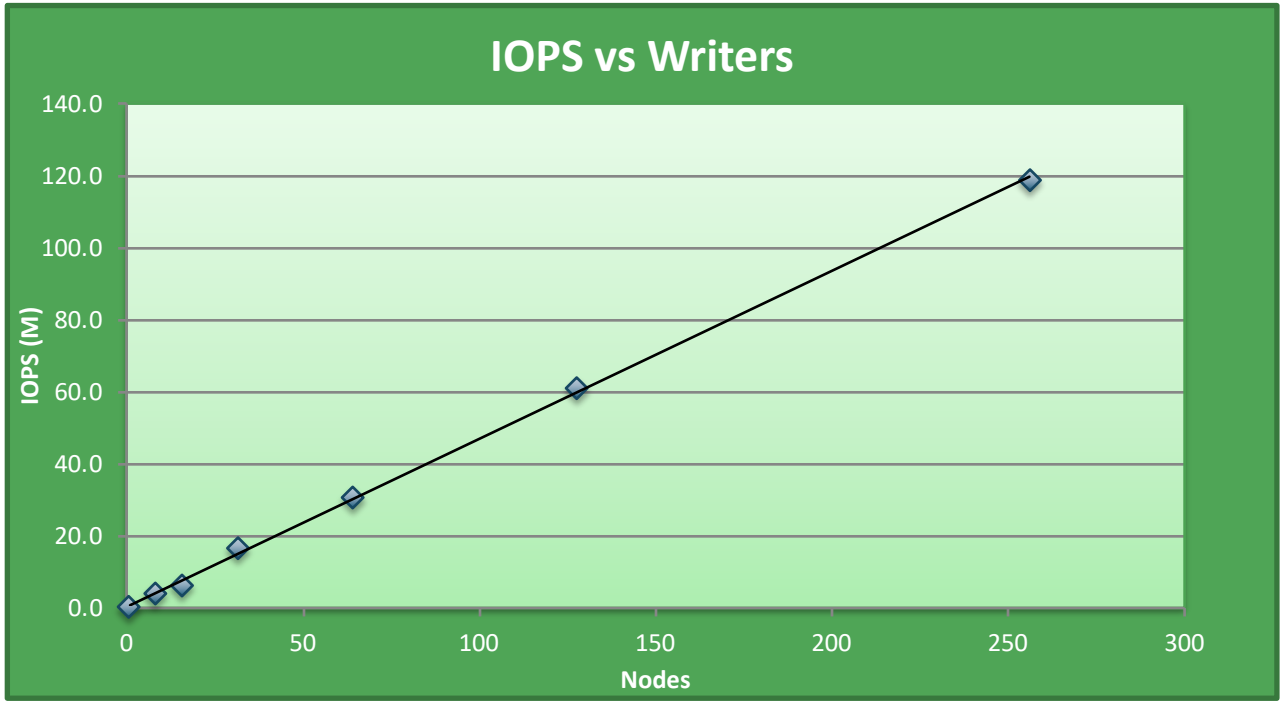


Creating a Monster

- The software stack is very big: 3.5 TB and 20M inodes
- Compress and deduplicate with squashfs: 350 GB
- Start up time shows excellent scaling out to 500 nodes (16,000 cores)



Per-Node Write Cache (IOPS)



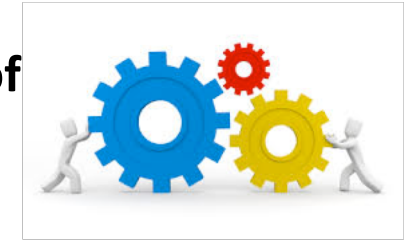
Results of an IOR File per-process, 2 tasks per node, 512B transfer size, 2GB write. 100x faster than Lustre at the same scale.

Why not just run Docker

- **Security: Docker currently uses an all or nothing security model. Users would effectively have system privileges**

```
> docker run -it -v /:/mnt --rm busybox
```

- **System Architecture: Docker assumes local disk**
- **Integration: Docker doesn't play nice with batch systems.**
- **System Requirements: Docker typically requires a very modern kernel**
- **Complexity: Running real Docker would add new layers of complexity**
- **Scale: Stock Docker would create a several scaling issues.**



Our Solution: Shifter

Design Goals and Principles:

- **Compatibility with Docker Images and Docker Registries**
- **Security – Enable site to configure and enforce policies**
- **Ability to run at scale on NERSC systems**
- **User independence: Require no administrator assistance to pull and use an image**
- **Full access to shared resource availability (e.g., file systems and network interfaces)**
- **Seamless user experience**
- **Open Source (https://github.com/nersc/shifter)**

Design Choices

- **Convert Docker images to a squashed format and loop mount the image**
  - Avoids adding additional metadata load on the PFS. Scales much better.
  - Conversion only needs to be done once and image is immediately visible across the system.
- **Centralized Image Management**
  - Prevents image corruption after mounting.
  - Prevents malicious user from crafting an image that could be used to attack the kernel.
- **Containers run as the user and images are mounted read-only**
- **Site based configuration**
  - Enable the site to limit what mount points the user can volume mount in the container.
  - Provides added security and further limits exposure.

Basic Idea

Users use Docker to create and build images

Once for each image:

1. Download the contents of an image
2. Unpack the contents
3. “Flatten” the image into a squashfs file
4. Copy that squash file to a global/cluster file system

At Runtime:

1. Mount the image with a loop back mount at run-time