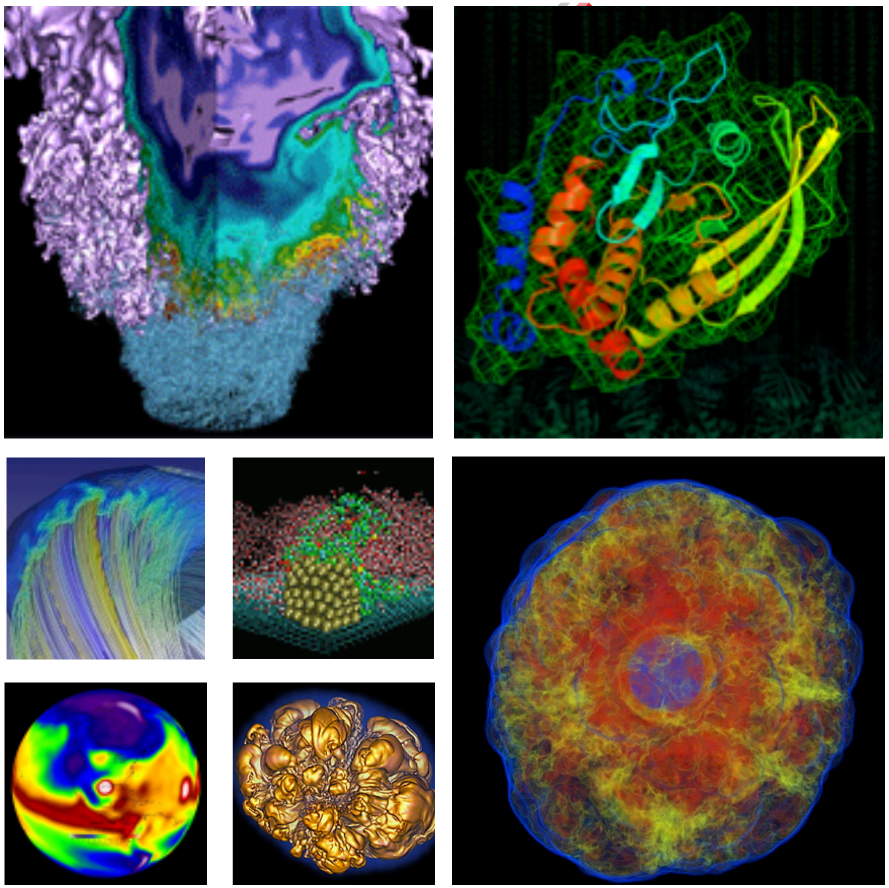
Interactive Session: Docker and Shifter



NERSC

CSCS

CRAY THE SUPERCOMPUTER COMPANY

Shane Canon & Doug Jacobsen (NERSC/LBNL)
Sadaf Alam, Lucas Benedicic, Nick Cardo, &
Miguel Gila (CSCS)
Don Balhs (Cray)

Cray User Group 2016 May 12, 2016

U.S. DEPARTMENT OF ENERGY | Office of Science

Agenda

- Background
- Use Cases
- Security
- Questions and Discussion

Motivation

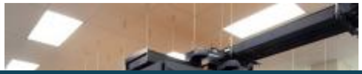
- Data Intensive and other emerging workloads need access to HPC scales.
- Many of these workloads involve complex software stacks with layers of dependencies.
- Tools like Docker have been developed that provide a new model for packaging, shipping and running software.
- Users are starting to ask for this capability in HPC centers and shared resource providers.

CSCS
Centro Svizzero di Calcolo Scientifico
Swiss National Supercomputing Centre

Converging Data Intensive Systems and HPC

Compute Intensive

Data Intensive



Carver

Why Convergence?

- Scale: Cori will have the scale needed to tackle current and emerging data challenges
- Coupling: Increasing Need to Couple Simulation and Analysis
- Capabilities: Access to the Burst Buffer
- Exascale: Helps place data intensive communities on exascale path

Docker Basic's

Build

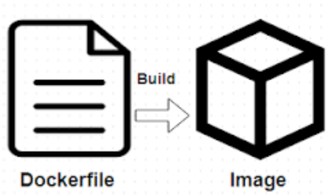
Ship

Run

- Build images that captures applications requirements.
- Manually commit or use a recipe file.

- Push an image to DockerHub, a hosted registry, or a private Docker Registry.
- Share Images

- Use Docker Engine to pull images down and execute a container from the image.



Why not just run Docker

- System Architecture: Docker assumes local disk
- Security: Docker currently uses an all or nothing security model. Users would effectively have system privileges

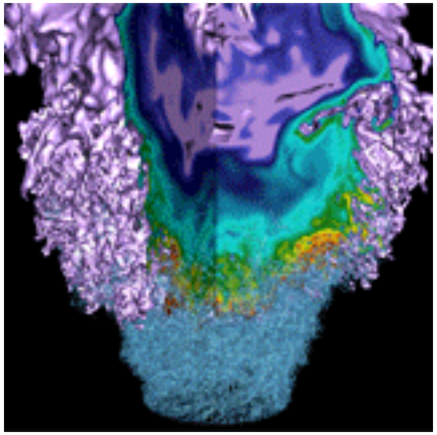
- Integration: Docker doesn't play nice with batch systems.
- System Requirements: Docker typically requires very modern kernel

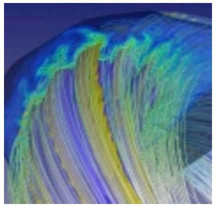
- Complexity: Running real Docker would add new layers of complexity

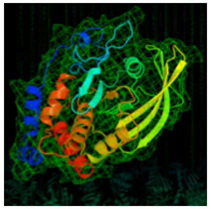
Solution: Shifter

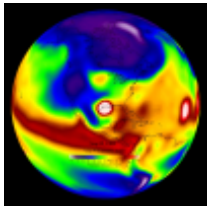
- **Partnership with Cray to design a solution to run containers on an HPC platform.**
- **Design Goals:**
  - User independence: Require no administrator assistance to launch an application inside an image
  - Shared resource availability (e.g., PFS/DVS mounts and network interfaces)
  - Leverages or integrates with public image repos (i.e. DockerHub)
  - Seamless user experience
  - Robust and secure implementation

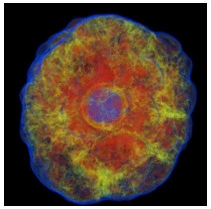
Implementation

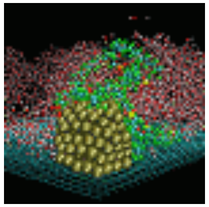






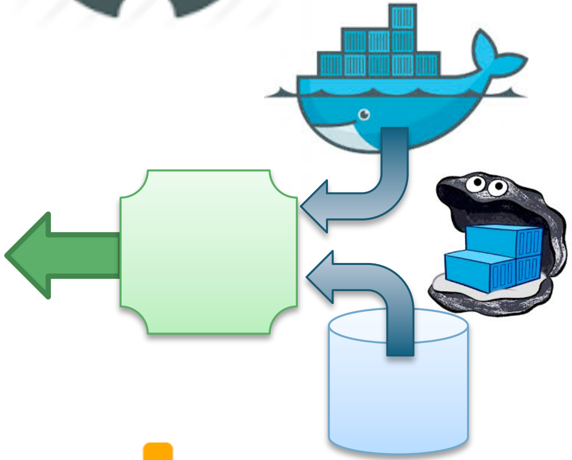






Shifter Components

- **Shifter Image Gateway**
  - Imports and converts images from DockerHub and Private Registries
- **Shifter Runtime**
  - Instantiates images securely on compute resources
- **Work Load Manager Integration**
  - Integrates Shifter with WLM

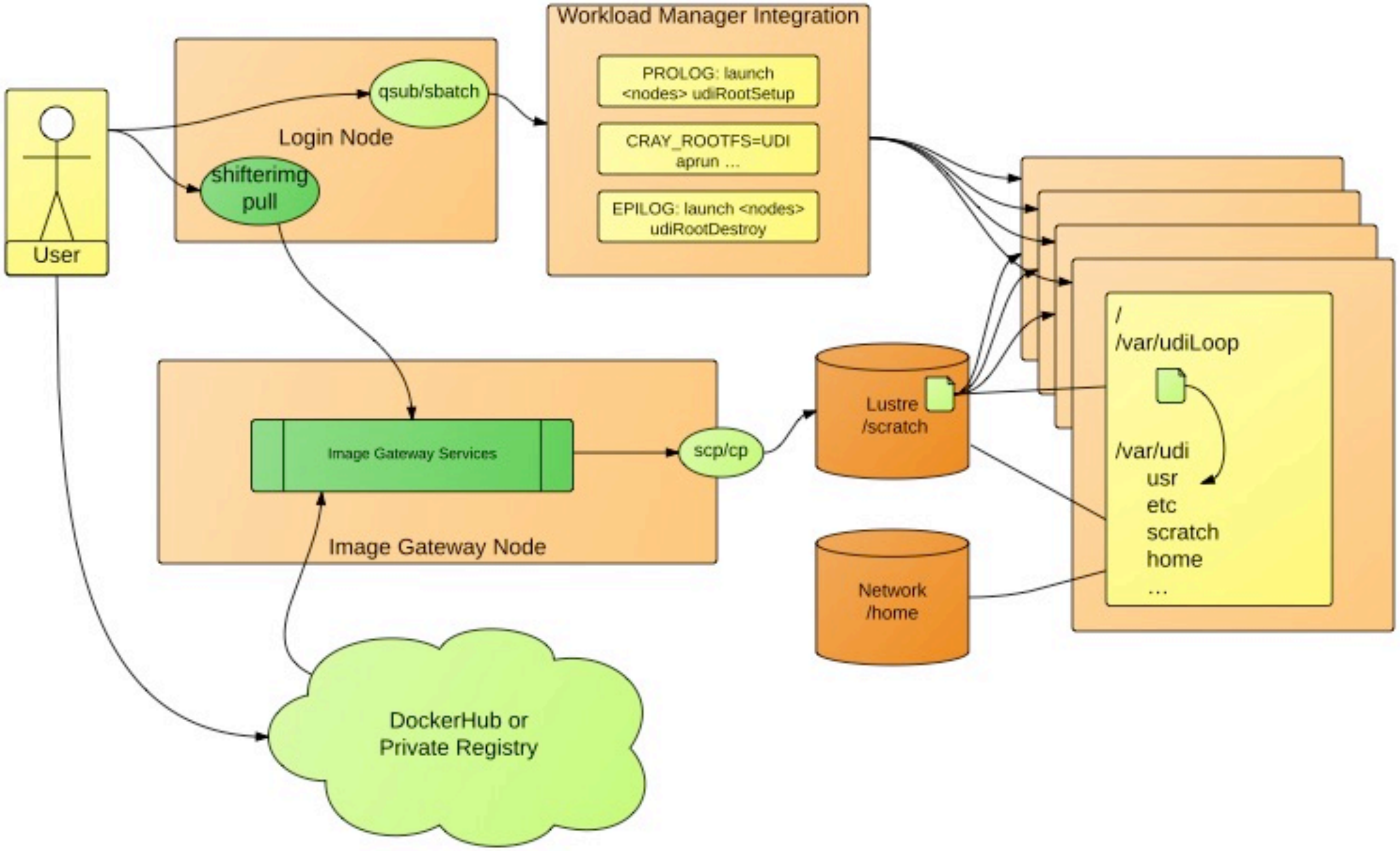


slurm workload manager

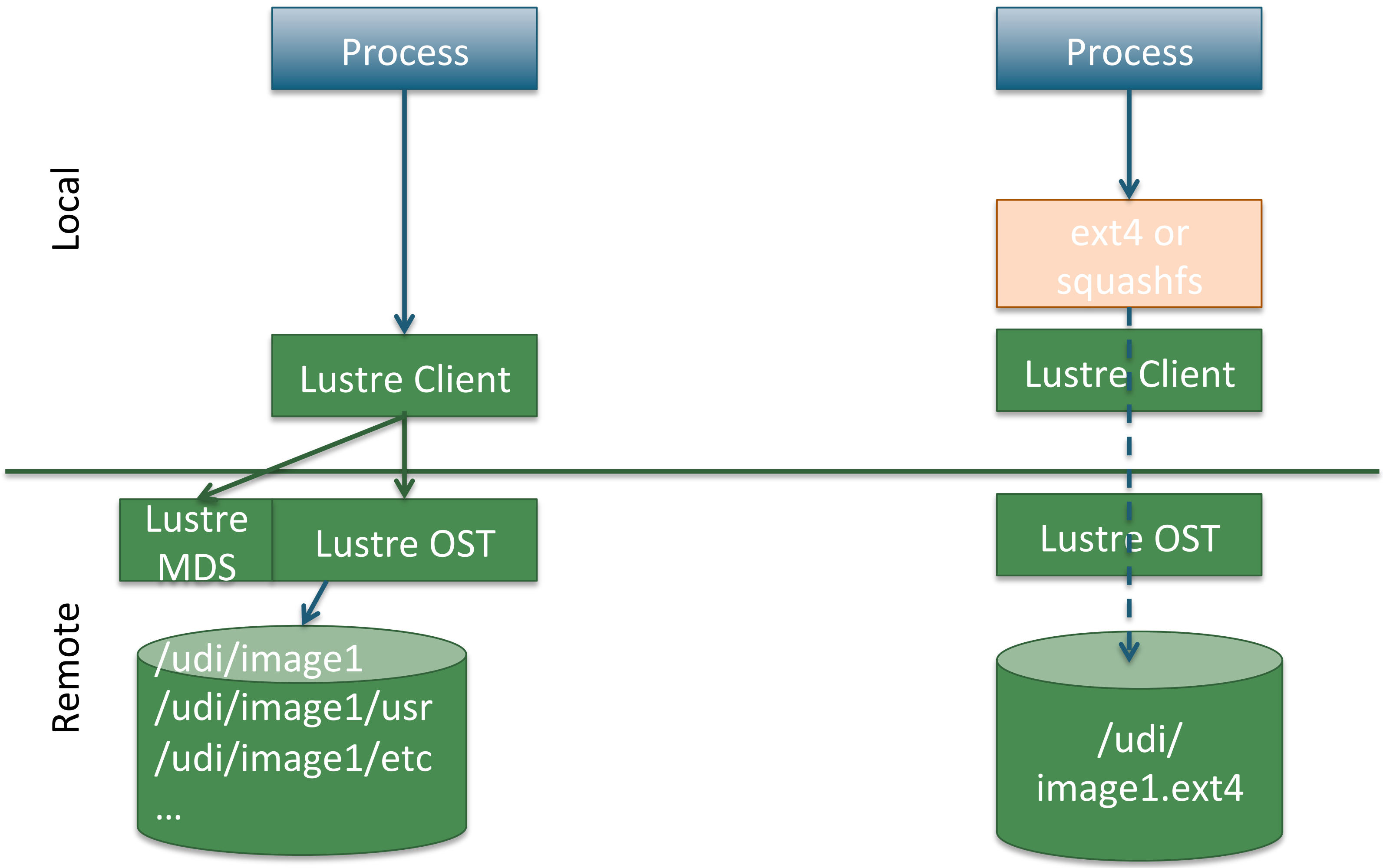
grid engine

Adaptive COMPUTING

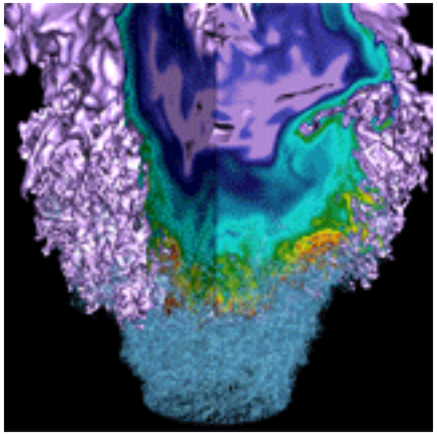
Shifter Architecture and Flow

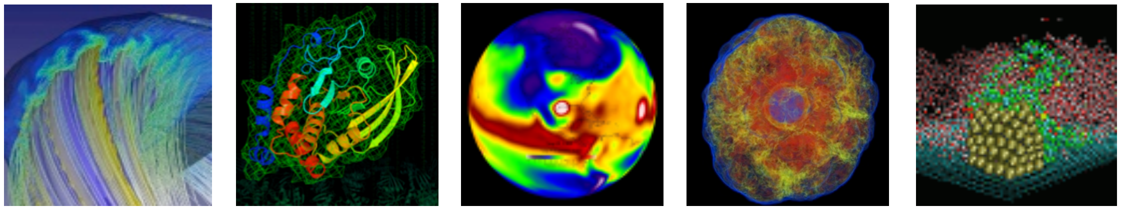


File System flow – Traditional vs Shifter



Use Cases and Shifter in Action





Create an image with Docker

Dockerfile

```
FROM ubuntu:14.04
MAINTAINER Shane Canon scanon@lbl.gov
# Update packages and install dependencies
RUN apt-update -y && \
   apt-get install -y build-essential

# Copy in the application
ADD . /myapp
# Build it
RUN cd /myapp && \
    make && make install
```

```
> docker build –t scanon/myapp:1.1 .
> docker push scanon/myapp:1.1
```

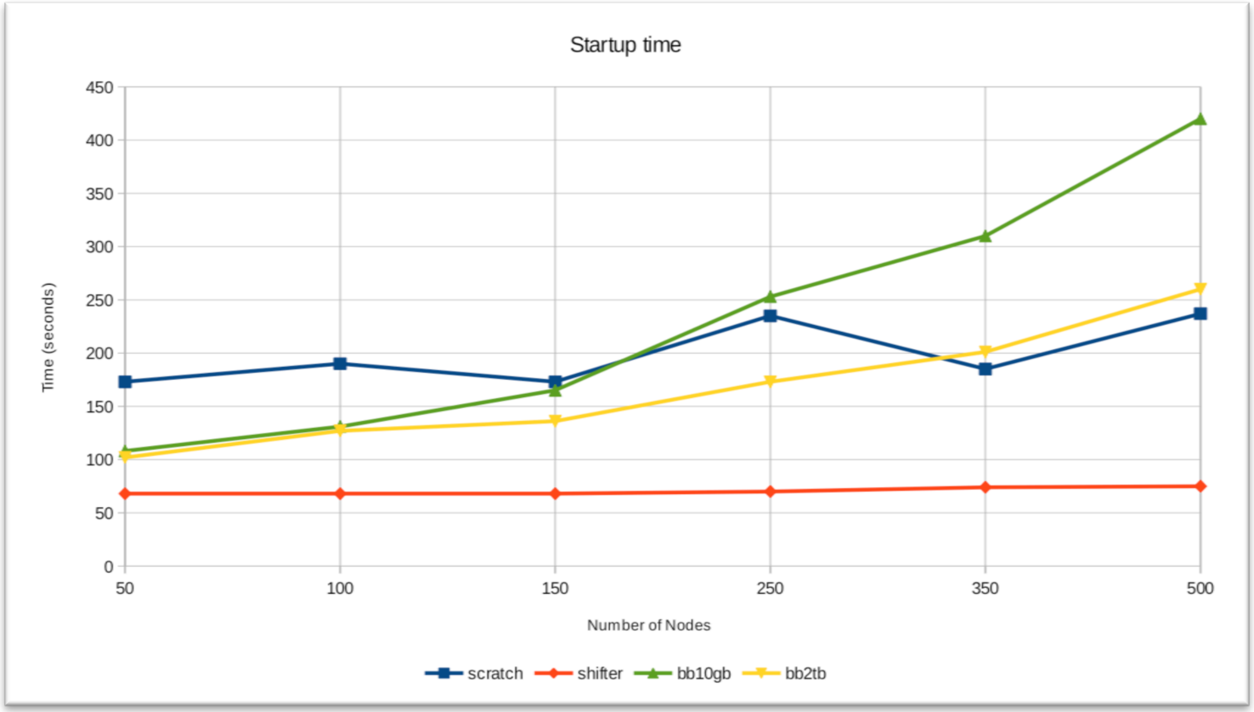
Use the Image with Shifter

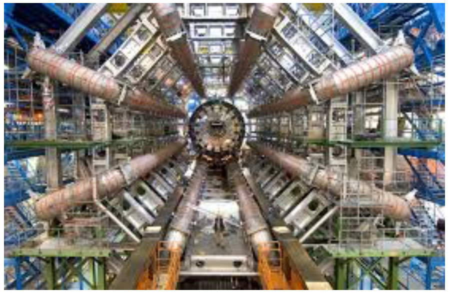
```
#!/bin/bash
#SBATCH -N 16 -t 20
#SBATCH --image=docker:ubuntu:14.04
#SBATCH --volume=/global/cscratch1/sd/canon/
backingFile:/mnt:perNodeCache=size=100G

module load shifter
export TMPDIR=/mnt
srun -n 16 shifter /myapp/app
```

```
> sbatch ./job.sl
```

Shifter and Atlas





- ATLAS software built and maintained by the international collaboration.
- Makes heavy use of “CVMFS” a software distribution system.
- Complete ATLAS CVMFS distro is O(TB) in size.
- Shifter provides linear startup times and requires no additional integration to run on the Cray systems.

Spark

- "Big Data" high productivity analytics Framework
- Designed around commodity clusters (Ethernet network and local disk)
- Shifter image: lgerhardt/spark-1.6.0
- Uses per-Node write cache for spills and other temporary per-node file caches.
- Tested up to full scale of Cori Phase 1 (1600 nodes) with multiple Spark applications.

GPUs

The Docker catch with GPUs and Shifter

- **Docker containers are both hardware-agnostic and platform-agnostic *by design*.**
- **This is not the case when using GPUs since:**
  - it is using specialized hardware, and
  - it requires the installation of the NVIDIA kernel driver
- **With Shifter**
  - The required character devices are automatically exposed when starting the container on the target machine (/dev/nvidiaX)
  - With the pre-mount hooks provided by Shifter we expose the driver files to the container image, and alter the runtime library search configuration

Success stories: GPU Stream

- **GPU Stream benchmark within a Shifter container shows native performance!**

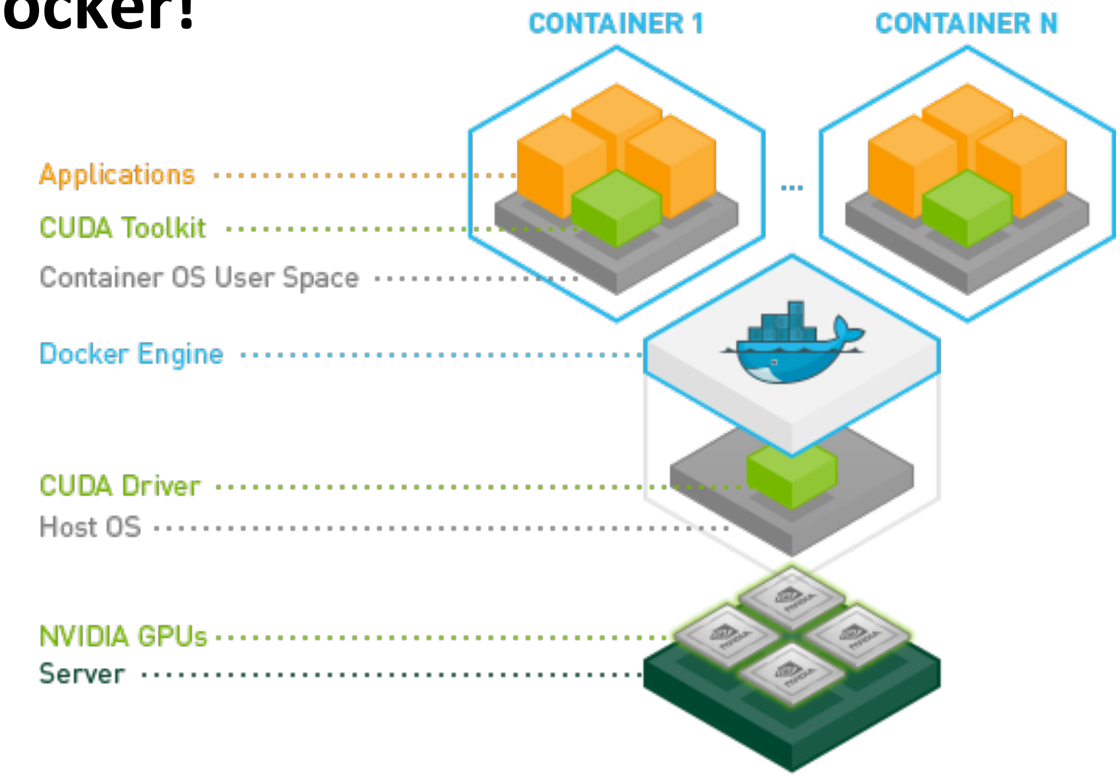
GPU: Stream benchmark

```
lucasbe@santis01 ~/shifter-gpu> sbatch ./nvidia-docker/samples/cuda-stream/
benchmark.sbatch
Submitted batch job 496

lucasbe@santis01 /scratch/santis/lucasbe/jobs> cat shifter-gpu.out.log
Launching GPU stream benchmark on nid00012 ...
STREAM Benchmark implementation in CUDA
Array size (double precision) = 1073.74 MB
using 192 threads per block, 699051 blocks
Function      Rate (GB/s)  Avg time(s)  Min time(s)  Max time(s)
Copy:         184.3169     0.01167758   0.01165104   0.01170397
Scale:        183.1849     0.01175387   0.01172304   0.01178598
Add:          180.3075     0.01790012   0.01786518   0.01792288
Triad:        180.1056     0.01790700   0.01788521   0.01794291
```

Success stories: NVidia DGX-1

- **NVIDIA DGX-1 uses the engineered solution for Shifter in the management of its software stack… with Docker!**



https://github.com/NVIDIA/nvidia-docker

High Energy Physics

WLCG Swiss Tier-2

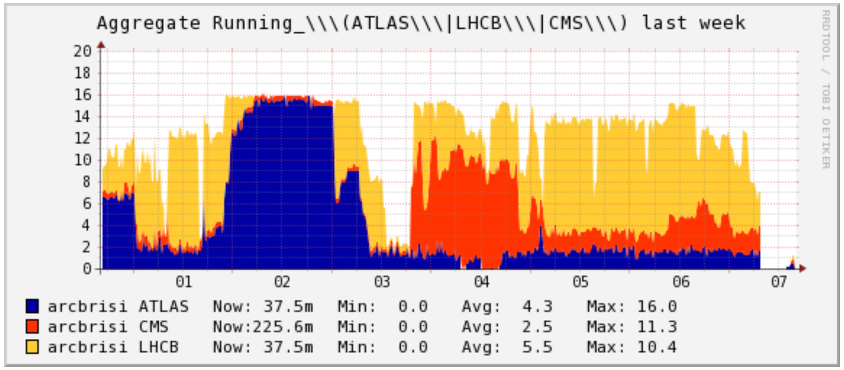
- CSCS operates the cluster Phoenix on behalf of CHIPP, the Swiss Institute of Particle Physics
- Phoenix runs Tier-2 jobs for ATLAS, CMS and LHCb, 3 experiments of the LHC at CERN and part of WLCG (Worldwide LHC Computing Grid)
- WLCG jobs need and expect RHEL-compatible OS. All software is precompiled and exposed in a cvmfs[*] filesystem
- But Cray XC compute nodes run CLE5, a modified version of SLES 11 SP3

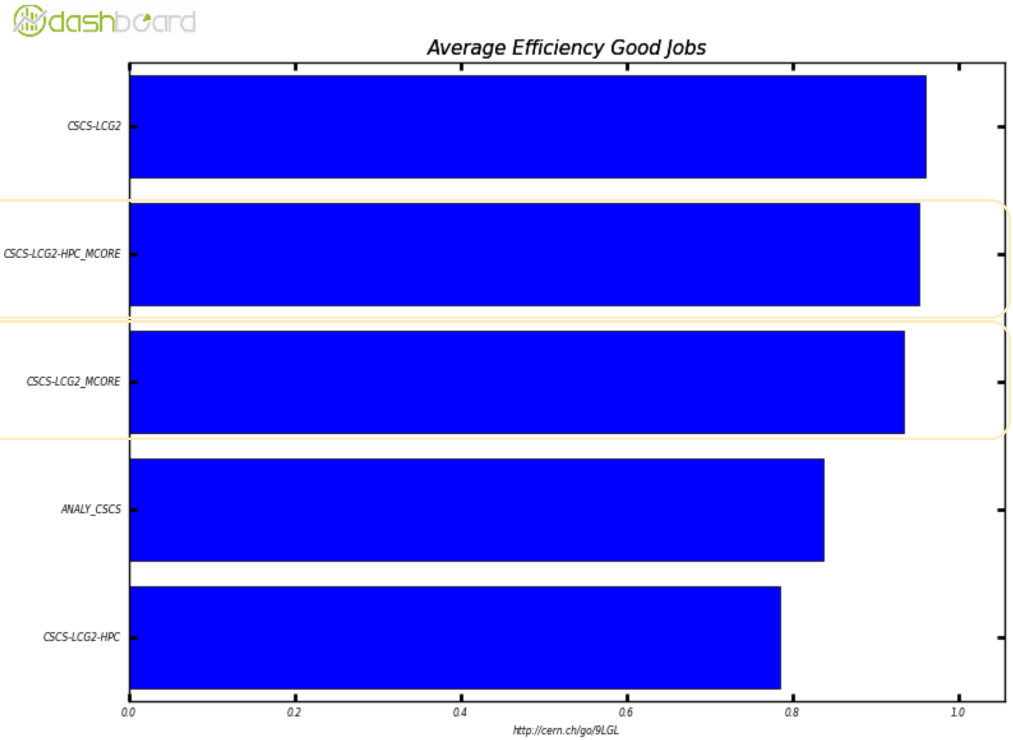
- So, how do we get these jobs to run on a Cray?

WLCG Swiss Tier-2

- **Using Shifter, we are able to run unmodified ATLAS, CMS and LHCb production jobs on a Cray XC TDS**
- **Jobs see standard CentOS 6 containers**
- **Nodes are shared: multiple single-core and multi-core jobs, from different experiments, can run on the same compute node**
- **Job efficiency is comparable in both systems**

WLCG Swiss Tier-2





```
JOBID  USER     ACCOUNT           NAME        NODELIST  ST    REASON  START_TIME     END_TIME  TIME_LEFT  NODES CPU
82471  atlasprd atlas   a53eb5f8_34f0_        nid00043   R      None    15:03:33    Thu 15:03 1-23:54:18     1    8
82476  cms04    cms            gridjob        nid00043   R      None    15:08:39 Tomorr 03:08   11:59:24     1    2
82451  lhcbplt  lhcb           gridjob        nid00043   R      None    15:00:10 Tomorr 03:00   11:50:55     1    2
82447  lhcbplt  lhcb           gridjob        nid00043   R      None    14:59:31 Tomorr 02:59   11:50:16     1    2
82448  lhcbplt  lhcb           gridjob        nid00043   R      None    14:59:31 Tomorr 02:59   11:50:16     1    2
82449  lhcbplt  lhcb           gridjob        nid00043   R      None    14:59:31 Tomorr 02:59   11:50:16     1    2
82450  lhcbplt  lhcb           gridjob        nid00043   R      None    14:59:31 Tomorr 02:59   11:50:16     1    2
82446  lhcbplt  lhcb           gridjob        nid00043   R      None    14:49:01 Tomorr 02:49   11:39:46     1    2
82444  lhcbplt  lhcb           gridjob        nid00043   R      None    14:48:01 Tomorr 02:48   11:38:46     1    2
82445  lhcbplt  lhcb           gridjob        nid00043   R      None    14:48:01 Tomorr 02:48   11:38:46     1    2
```

Shifter and MPI

- **In Image**
  - Add required libraries directly into image.
  - Users would have to maintain libraries and rebuild images after an upgrade.
- **Managed Base Image (Golden Images)**
  - User builds off of a managed image that has required libraries.
  - Images are built or provided as part of a system upgrade.
  - Constrained OS choices and a rebuild is still required.
- **Volume Mounting**
  - Applications built using ABI compatibility.
  - Appropriate libraries are volume mounted at run time.
  - No rebuild required, but may not work for all cases.

Advanced example with MPI support

```
FROM cern/slc6-lite:latest
## update packages and install dependencies
##    csh, tar, perl needed for cctbx
##    gcc, zlib-devel needed to build mp4ipy
RUN yum upgrade -y && \
    yum install csh tar numpy scipy matplotlib gcc -y

WORKDIR /

## replace psdm mpi4py with cray-tuned one
ADD optcray_alva.tar /
ADD mpi4py-1.3.1.tar.gz /usr/src
ADD mpi.cfg /usr/src/mpi4py-1.3.1/
RUN cd /usr/src/mpi4py-1.3.1  && \
    chmod -R a+rX /opt/cray && \
    chown -R root:root /opt/cray && \
    python setup.py build && \
    export MPI4PY_LIB=$( rpm -ql $(rpm -qa | grep mpi4py | head -1) | egrep "lib$" ) && \
    export MPI4PY_DIR="${MPI4PY_LIB}/.." && \
    python setup.py install && \
    cd /  && rm -rf /usr/src/mpi4py-1.3.1 && \
    printf "/opt/cray/wlm_detect/default/lib64/libwlm_detect.so.0\n" >>
            /etc/ld.so.preload && \
    (echo "/opt/cray/mpt/default/gni/mpich2-gnu/48/lib\n/opt/cray/pmi/default/lib64”;\
     echo “/opt/cray/ugni/default/lib64\n/opt/cray/udreg/default/lib64”;\
     echo “/opt/cray/xpmem/default/lib64\n/opt/cray/alps/default/lib64”) \
     >> /etc/ld.so.conf && \
    ldconfig
```

Why Users will like Docker and Shifter

- Develop an application on your desktop or laptop and easily run it on a Cray or other Supercomputer
- Enables the user to solve their dependency problems themselves
- Run the (Linux) OS of their choice and the software versions they need
- Improves application performance in many cases
- Improves reproducibility
- Improves sharing (through sites like Dockerhub)

How is Shifter similar to Docker?

- **Sets up user-defined image under user control**
- **Allows volume remapping**
  - mount /a/b/c on /b/a/c in container
- **Containers can be "run"**
  - Environment variables, working directory, entrypoint scripts can be defined and run
- **Can instantiate multiple containers on same node**

How does Shifter differ from Docker?

- **User runs as the user in the container – not root**
- **Image modified at container construction time:**
  - Modifies /etc, /var, /opt
    - replaces /etc/passwd, /etc/group other files for site/security needs
    - adds /var/hostsfile to identify other nodes in the calculation (like $PBS_NODEFILE)
    - Injects some support software in /opt/udiImage
  - Adds mount points for parallel filesystems
    - Your homedir can stay the same inside and outside of the container
    - Site configurable
- **Image readonly on the Computational Platform**
  - to modify your image, push an update using Docker
- **Shifter only uses mount namespaces, not network or process namespaces**
  - Allows your application to leverage the HSN and more easily integrate with the system
- **Shifter does not use cgroups directly**
  - Allows the site workload manager (e.g., SLURM, Torque) to manage resources
- **Shifter uses individual compressed filesystem files to store images, not the Docker graph**
  - Uses more diskspace, but delivers high performance at scale
- **Shifter integrates with your Workload Manager**
  - Can instantiate container on thousands of nodes
  - Run parallel MPI jobs
- **Specialized sshd run within container for exclusive-node for non-native-MPI parallel jobs**
  - PBS_NODESFILE equivalent provided within container (/var/hostsfile)
  - Similar to Cray CCM functionality
  - Acts in place of CCM if shifter “image” is pointed to /dsl VFS tree

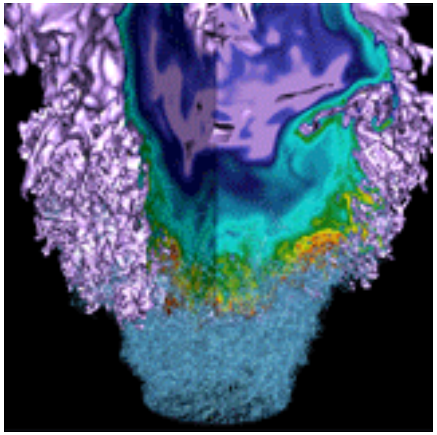
Shifter Security Model

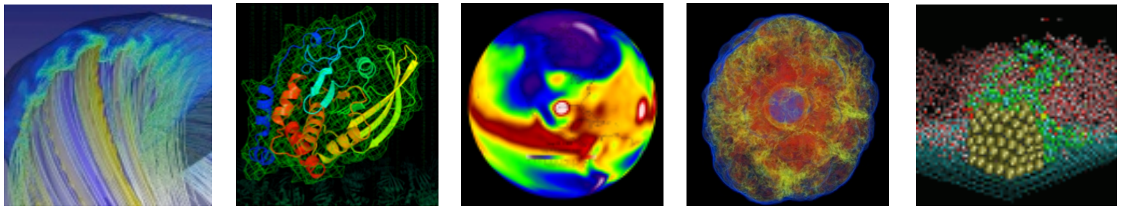
- **User *only* accesses the container as their uid, never root or contextual root**
- **Generated site /etc/passwd, /etc/group (filtered) is placed in container**
  - no need for shifter containers to interoperate with LDAP or concerns about image PAM
- **Optional sshd run within image is statically linked, only accessible to that container's user**
- **All user-provided data (paths within image, environment variables, command line arguments) are filtered**
- **Executables that run with root privileges are implemented in C with only glibc dependencies**
- ***All* filesystems in container are remounted no-setuid**
- ***All* processes that run with privilege carefully manage environment to prevent accidental/intentional manipulation**

Roadmap

- **16.05 Release:**
  - Support for RHEL 6/7, SLES 11/12, Rhine/Redwood
  - RPM builds
  - Improved scaling
  - UI Improvements
  - Per-node write cache
  - Bug Fixes
- **16.08 Release**
  - ACL support (private and authenticated images)
  - Image expiry and removal
  - Image usage statistics and metrics
  - Overlayfs support (stretch)
  - Debian packages for Ubuntu LTS

Discussion





Questions – Security and Support

- **What other security concerns or questions do you have?**
  - What about security and user images?
- **Support questions?**
  - How should images be maintained?
  - How do we troubleshoot problems with images?

Questions - Adoption

- **What is the level of adoption for you institution and users?**
  - Are you using Docker or containers already?
  - Do any of your users have Dockerized applications?
  - Have your users asked about Docker or containers?

Questions – User Training

- **How should we train users?**
  - Do sites already have materials?
  - What can we leverage from the Docker community?
  - Can we use materials from other communities (e.g. Software Carpentry)?
  - Are there any best practices that are worth capturing and sharing?

Questions – Advanced Issues

- How do we handle MPI and GPU?
- How do you handle licensed, proprietary or sensitive software?
- How should users distribute and share images?

Questions – Next Steps

- **Next Steps?**
  - What do we need from the Vendors?
  - Are sites interested in deploying Shifter?

NERSC

National Energy Research Scientific Computing Center