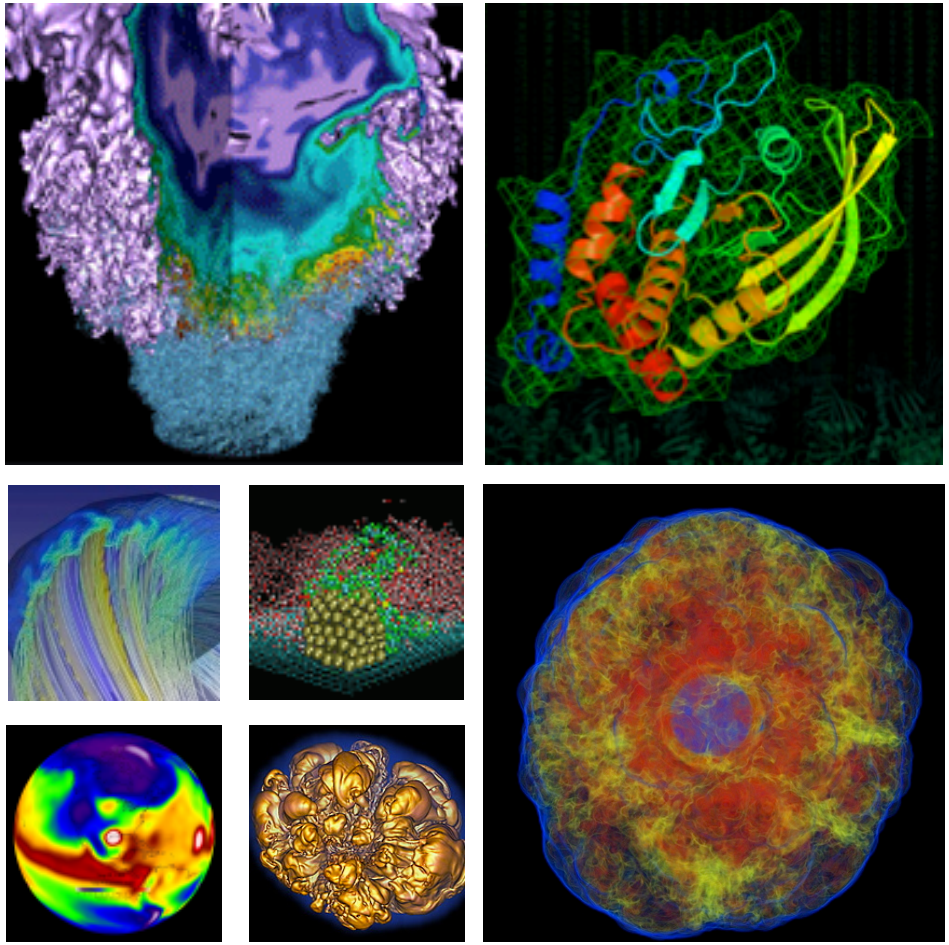
Intro to Shifter



Shane Canon

SC17 - Tutorial

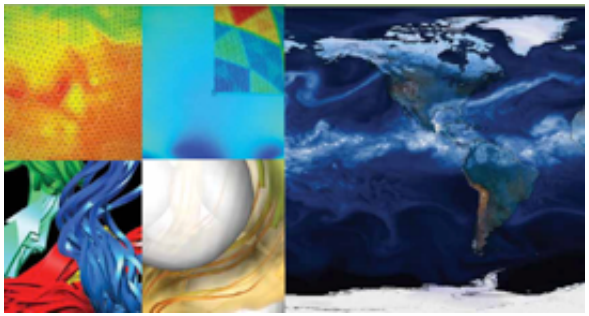
November 13, 2017

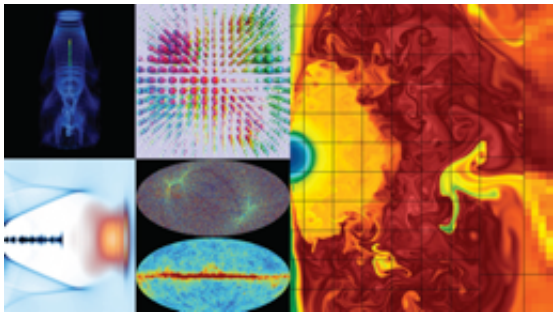
Topics

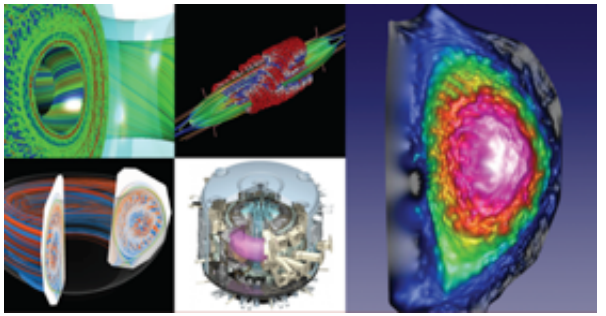
- Motivation
- Shifter Architecture and Design
- Shifter in Action
- Related Work and Discussion
- Installing Shifter

NERSC is the mission HPC computing center for the DOE Office of Science

- 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 6000 users and 750 projects
- Over 2000 publication resulting in NERSC resources per year
- New Data Initiative: *Pioneer new capabilities to enable scientists to make large-scale data-intensive science discoveries.*







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
```

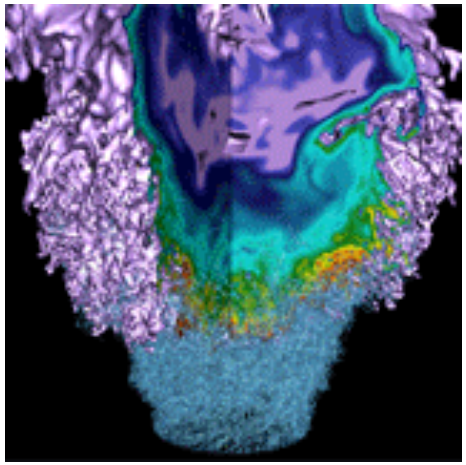
- **System Architecture: Docker assumes local disk**
- **Integration: Docker doesn't play nice with batch systems.**
- **System Requirements: Docker typically requires very modern kernel**

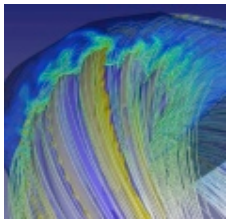
- **Complexity: Running real Docker would add new layers of complexity**
- **Scale: Stock Docker would create a several scaling issues.**

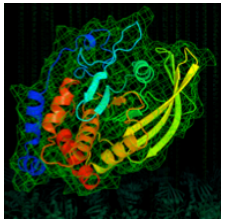
Solution: Shifter

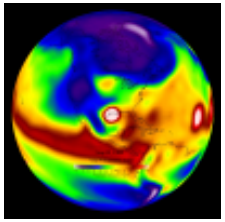
- **Design Goals:**
  - User independence: Require no administrator assistance to launch an application inside an image
  - Shared resource availability (e.g., file systems and network interfaces)
  - Leverage Docker image ecosystem (i.e. DockerHub)
  - Seamless user experience
  - Robust and secure implementation
  - Run at scale
- **Hosted at GitHub:**
  - https://github.com/nersc/shifter

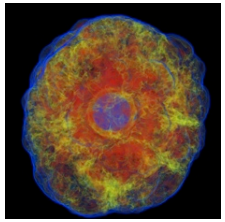
Implementation

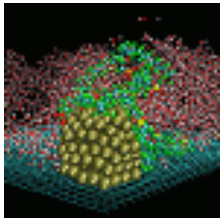












Basic Idea

Once for each image

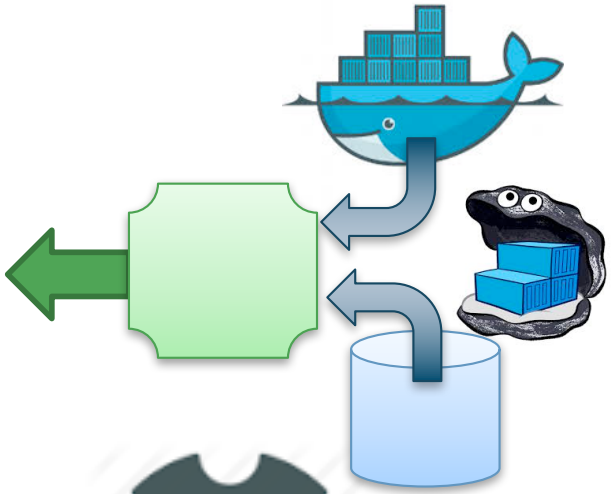
1. **Download the contents of an image**
2. **Unpack the contents**
3. **“Flatten” the image into a squashfs file**
4. **Copy that squash file to a global/cluster file system**

At Runtime

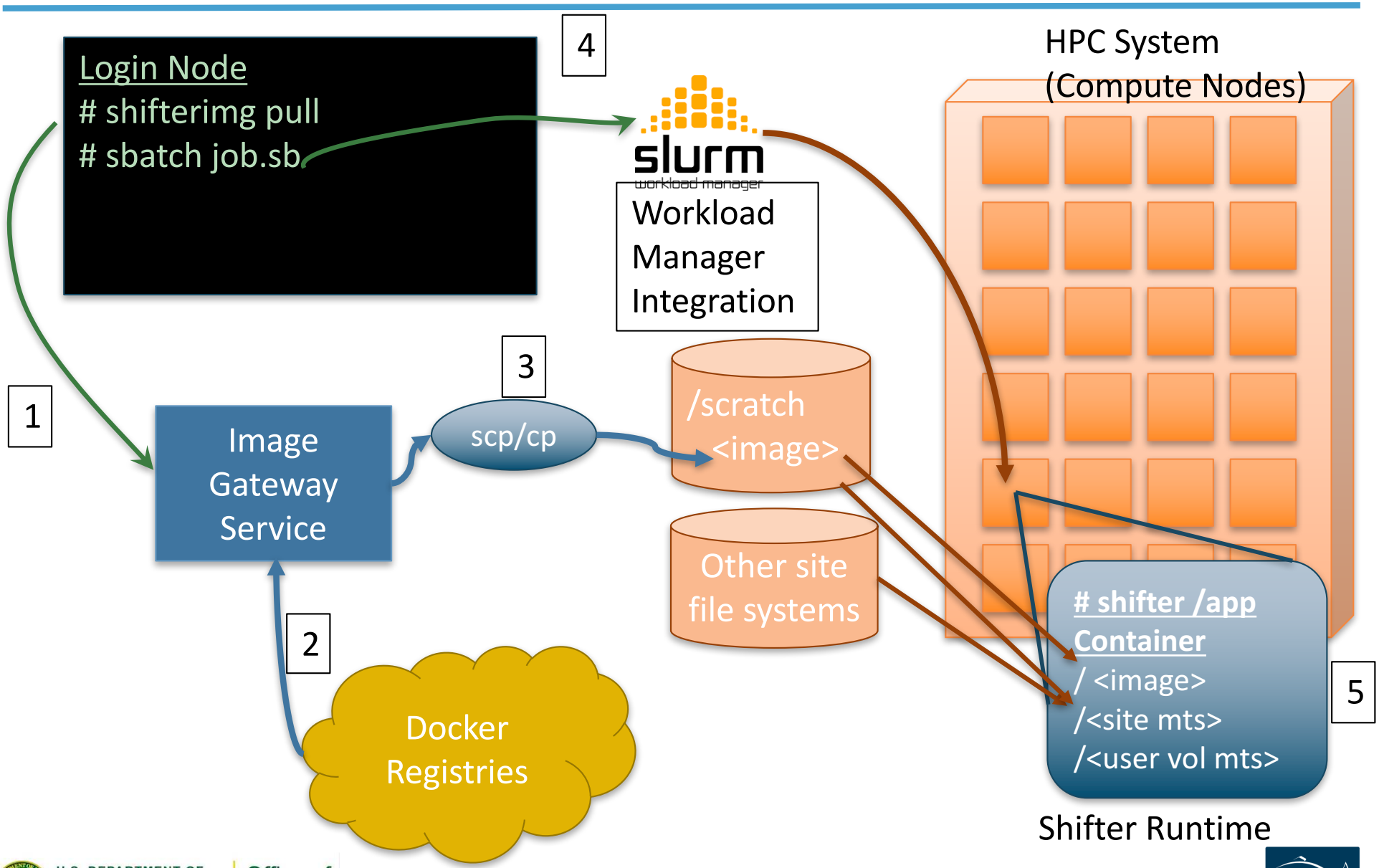
1. **Mount the image with a loop back mount at run-time**

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



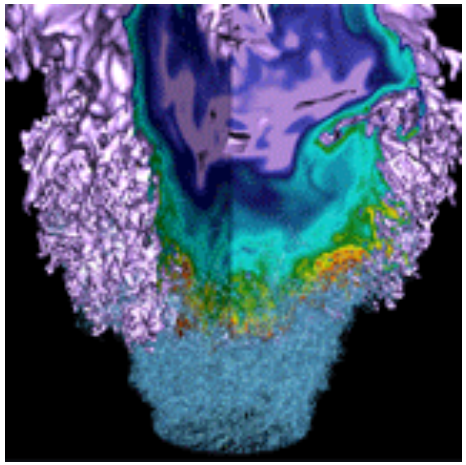
Shifter Architecture and Flow

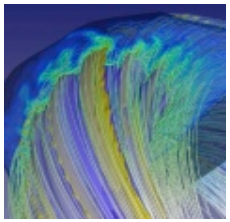


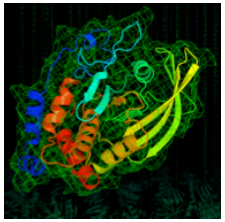
Workload Integration - Slurm

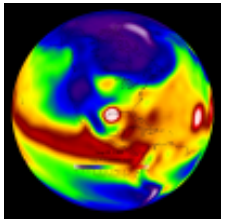
- Custom Plugin using the SLURM plugin architecture
- Allows users to specify images, volumes and other options directly in the batch submission.
- Extensions pre-create a common Shifter area (optimal for MPI applications).
- Support for Integration with other batch systems exist (Torque/MOAB, UGE)

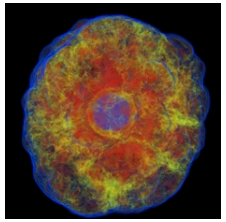
Shifter in Action

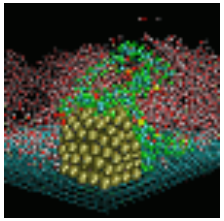












Basic Usage

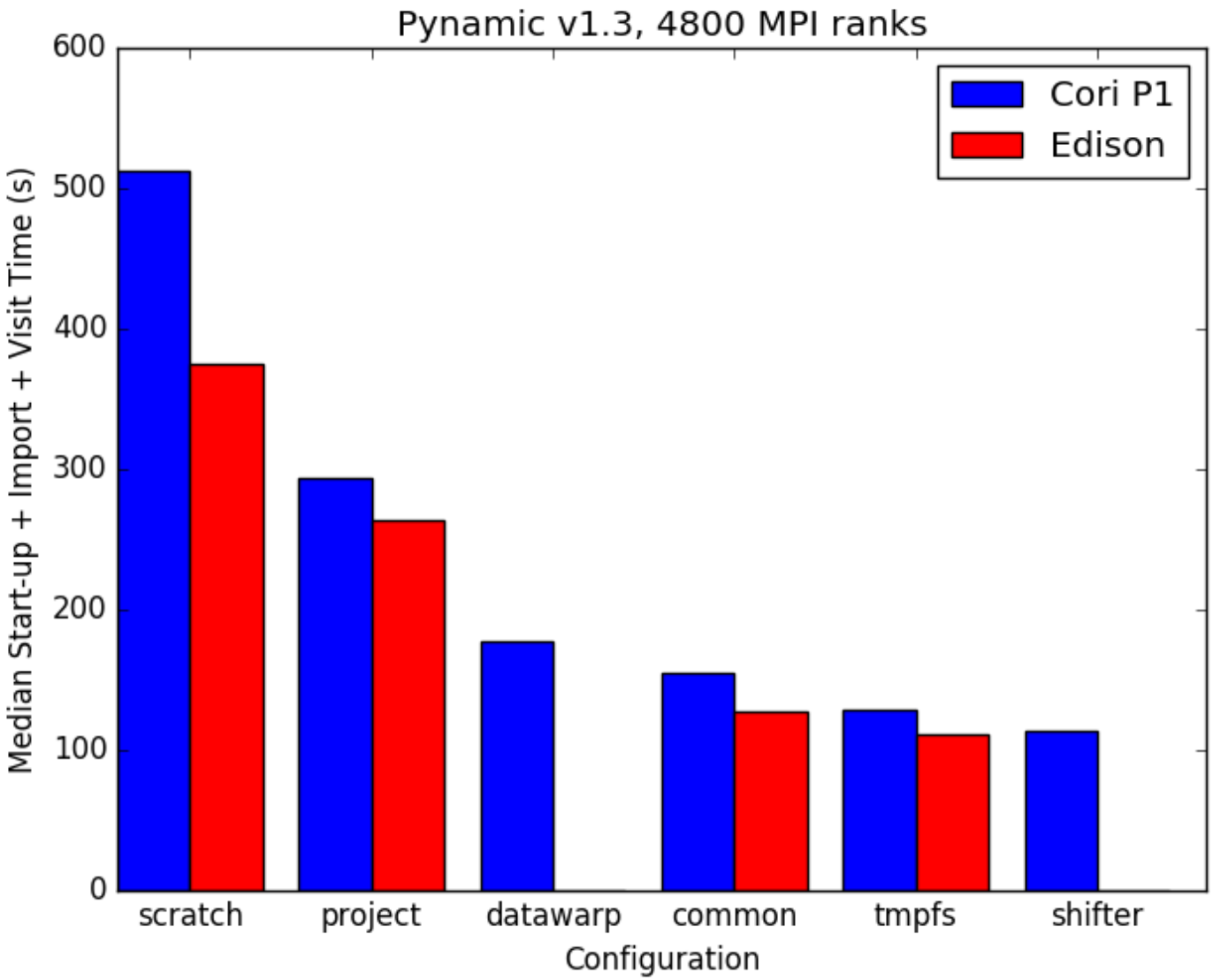
- **Pull image with shifterimg**

```
> shifterimg pull scanon/myapp:1.1
```

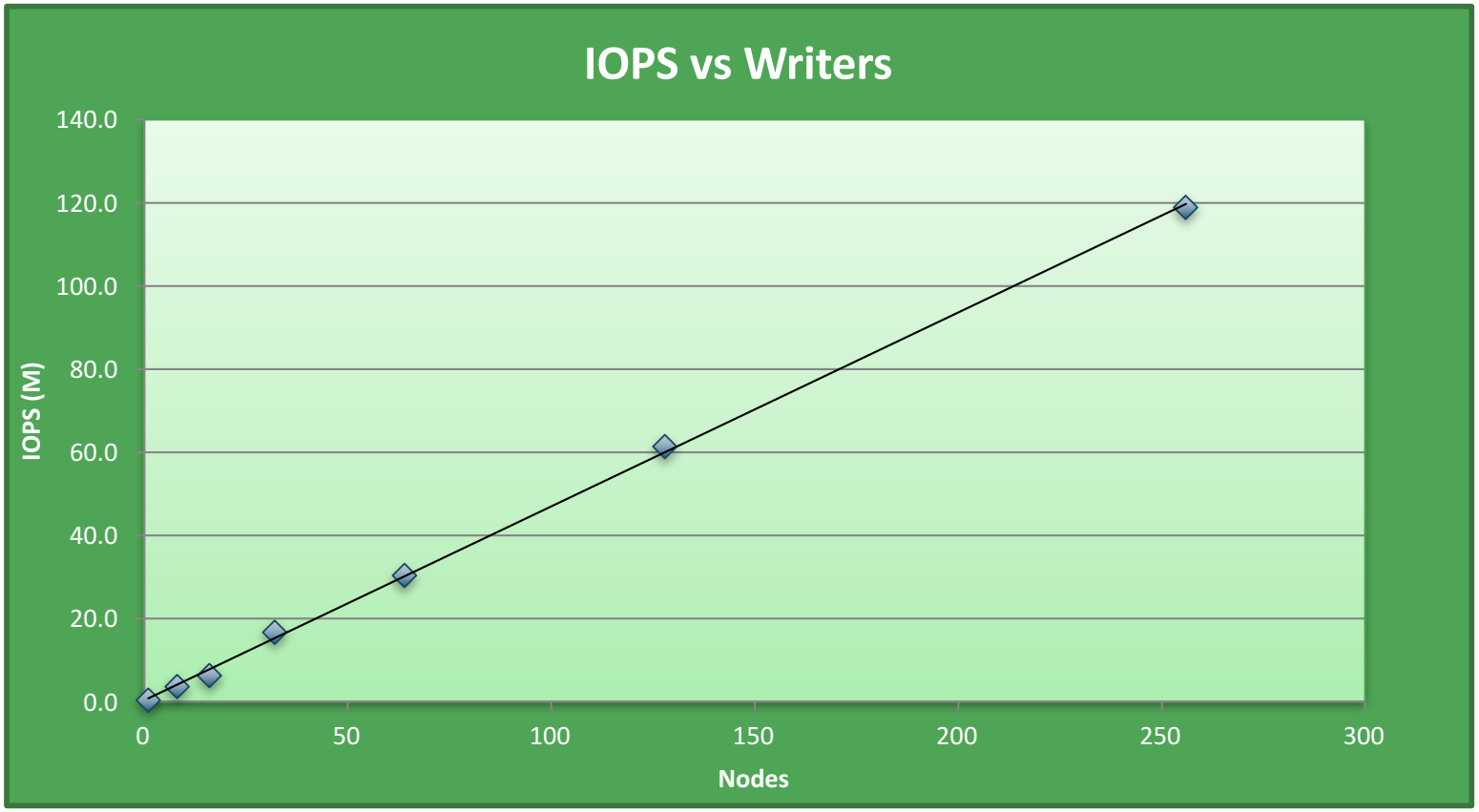
- **Use shifter to use the image**

```
> salloc –N 1 –C haswell –image=scanon/myapp:1.1
# shifter /myapp/app
```

Shifter accelerates Python Apps



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.

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.

Volume Mount Approach

```
FROM nersc/mpi-ubuntu:14.04

ADD . /app
RUN cd /app && \
    mpicc -o hello helloworld.c
```

Dockerfile

```
> docker build –t scanon/hello-vm:latest .
> docker push scanon/hello-vm:latest
```

Use the Image with Shifter

Submit script

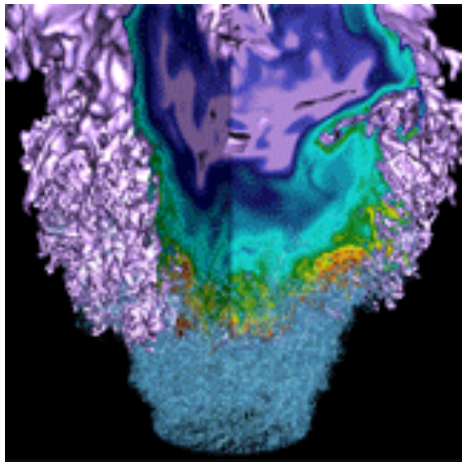
```
#!/bin/bash
#SBATCH -N 16 -t 20
#SBATCH --image=scanon/myapp:1.1

srun -n 16 shifter /myapp/app
```

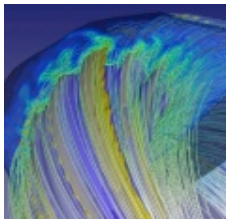
```
> shifterimg pull scanon/myapp:1.1
> sbatch ./job.sl
```

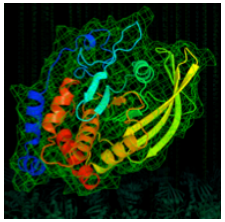
Use the Image with Shifter

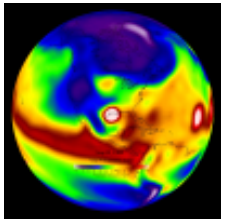
```
> shifterimg pull scanon/myapp:1.1
> salloc –N 1 –C haswell
salloc: Pending job allocation 8345797
salloc: job 8345797 queued and waiting for resources
salloc: job 8345797 has been allocated resources
salloc: Granted job allocation 8345797
salloc: Waiting for resource configuration
salloc: Nodes nid00443 are ready for job
nid00443:~> shifter –image=scanon/myapp:1.1
$
```

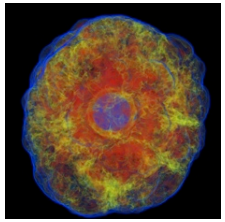
Related Work and Discussion

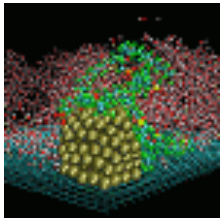












Other HPC Container Solutions

- **Singularity**
  - Very popular
  - Easy Installation
  - Runtime similar to Shifter
  - Native Image format in addition to Docker
- **CharlieCloud**
  - Very light-weight
  - No special privileges required
  - Separate tools to unpack Docker images

Why Users will like Shifter

Enables regular users to take advantage of Docker on HPC systems at scale.

This enables users to:

- Develop an application on the desktop or laptop and easily run it on a cluster or Supercomputer
- Solve their dependency problems themselves
- Run the (Linux) OS of their choice and the software versions they need

And…

- Improves application performance in many cases
- Improves reproducibility
- Improves sharing (through sites like Dockerhub)

Roadmap

- **17.11 Release (Pending)**
  - ACL support (private and authenticated images)
  - Improved GPU and MPI support
  - Image expiry and removal
  - Basic Image usage statistics and metrics
  - Integrated Image Import Support
  - Simplified Installation
- **Future**
  - Support for other image formats
  - OverlayFS support (write-able images)

Future Work

- Expand support for other image types and batch systems (with outside help)
- Continue to promote Docker and Shifter within the HPC community to increase access
- Explore other ways to integrate with Docker (e.g. Moby Project)

How does Shifter differ from Docker?

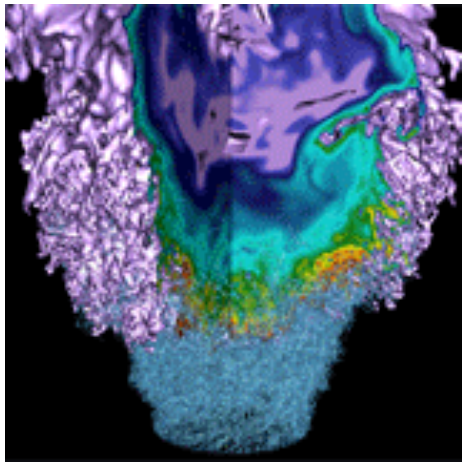
Most Noticeable

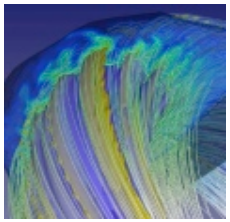
- Image read-only on the Computational Platform
- User runs as the user in the container – not root
- Image modified at container construction time (e.g. additional mounts)

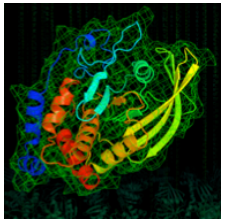
Less Noticeable:

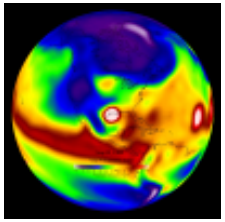
- Shifter only uses mount namespaces, not network or process namespaces
- Shifter does not use cgroups directly (integrated with the Workload Manager)
- Shifter uses individual compressed filesystem files to store images, not the Docker graph (slows down iterative updates)
- Shifter starts some additional services (e.g. sshd in container space)

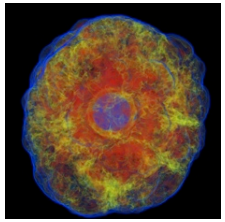
Installing Shifter

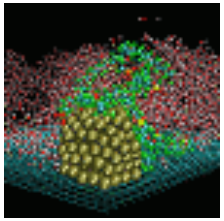












Testing with Shifter-in-a-box

- **Shifter-in-a-box is a complete Shifter install running in a docker container**
- For Docker4Mac install squashfs module first

```
docker run -it --rm --privileged scanon/squashfs
```

- Start shifter-in-a-box

```
> docker run -it --rm --privileged -v shifter:/data \
   scanon/shifterbox
# /src/start.sh
# su - auser
auser# shifterimg pull busybox
```

Quick Installation (CentOS7)

- Install EPEL
- Install Mongo and Munge
- Initialize and start Mongo and Munge

```
yum -y install epel-release
yum -y install mongodb-server mongodb wget munge
#
mkdir -p /data/db
mongod --smallfiles --fork \
    --logpath=/var/log/mongodb/mongo.log \
    --pidfilepath /mongo.pid
#
dd if=/dev/urandom of=/etc/munge/munge.key count=1
chown munge /etc/munge/munge.key
chmod 600 /etc/munge/munge.key
runuser -u munge -- /usr/sbin/munged  --force -F &
```

Quick Installation (CentOS 7)

- **Install Shifter RPMs**

```
R=http://portal.nersc.gov/project/bigdata/shifter/
T=17.11.b1-1.nersc.el7.centos
yum install $R/shifter{,-imagegw,-runtime}-$T.x86_64.rpm
```

- **Create config files**

```
cp /etc/shifter/imagemanager.json{.example,}
cp /etc/shifter/udiRoot.conf{.example,}
sed -i 's|etcPath=|etcPath=/etc/shifter/shifter_etc_files|' \
  /etc/shifter/udiRoot.conf
sed -i 's|imagegwapi|localhost|' /etc/shifter/udiRoot.conf
mkdir -p /images/cache
```

- **Start Services**

```
L=/var/log/shifter_imagegw
/usr/bin/gunicorn -D -b 0.0.0.0:5000 \
  --access-logfile$L/access.log --log-file$L/error.log \
  shifter_imagegw.api:app
```

Quick Installation (CentOS7)

- **Add user and test**
- **Create config files**
- **Start Services**

```
> useradd -m auser
> getent passwd > /etc/shifter/shifter_etc_files/passwd
> su auser
auser> shifterimg images
auser> shifterimg pull alpine
auser> shifter –image alpine
```

Summary

- Shifter enables HPC systems to easily and securely run Containers even at large scale
- Shifter provides the flexibility of Docker without sacrificing security, scalability or performance.
- Shifter opens the door to the many benefits of Docker including easy sharing of images, reproducibility, etc.

Acknowledgements

Cray

NERSC – Benchmarks and Use Cases

Lisa Gerhardt, Rollin Thomas, Wahid Bhimji, Debbie Bard

Others – Contributors and Use Cases

CSCS, Vakho Tsulaia, Ted Kisner

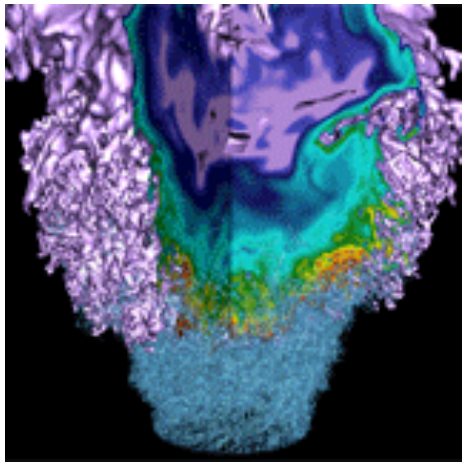
This work was supported by the Director, Office of Science, Office of Advanced Scientific Computing Research of the U.S. Department of Energy under Contract No. **DE-AC02-05CH11231**.

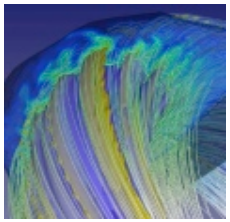
Questions?

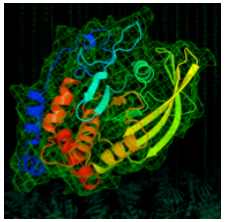
Why?

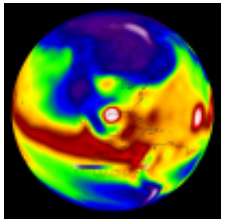
- Python must walk through the python libraries to construct the namespace
- Python must load up any dynamic libraries that are required
- The loader must traverse the LD_LIBRARY_PATH to find the libraries to load

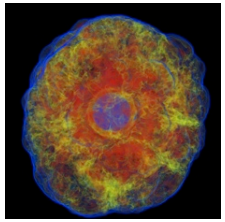
Advanced Installation Info











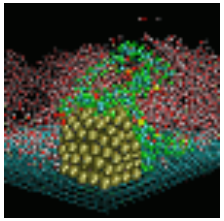
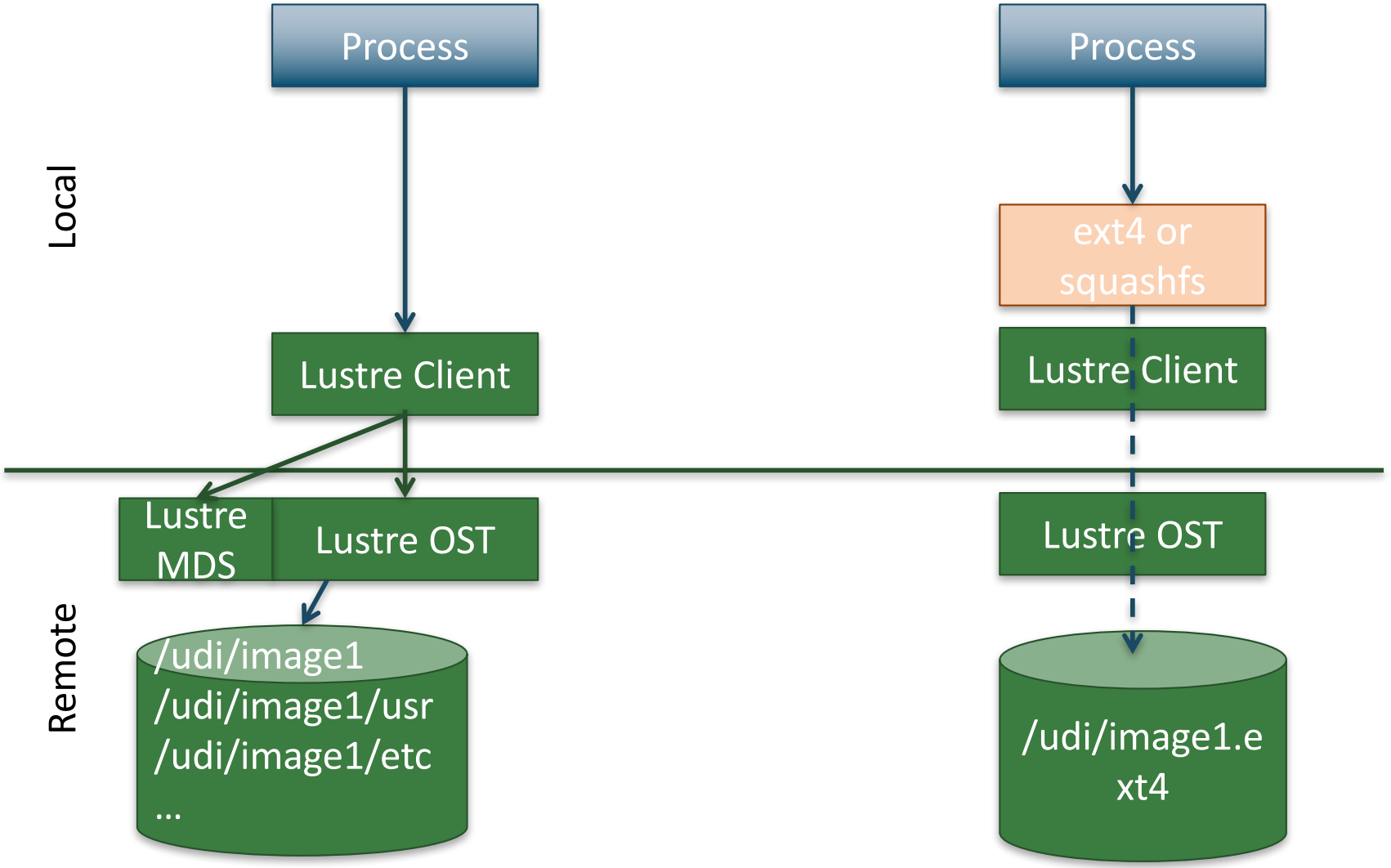


Image Gateway Options and Considerations

- Image Gateway can write image directly (local) or copy images via SCP
- Image Gateway can support multiple systems
- Image Gateway should have access to any external registries (e.g. DockerHub)
- Image Gateway should have sufficient local disk or ramdisk for caching layers and unpacking images

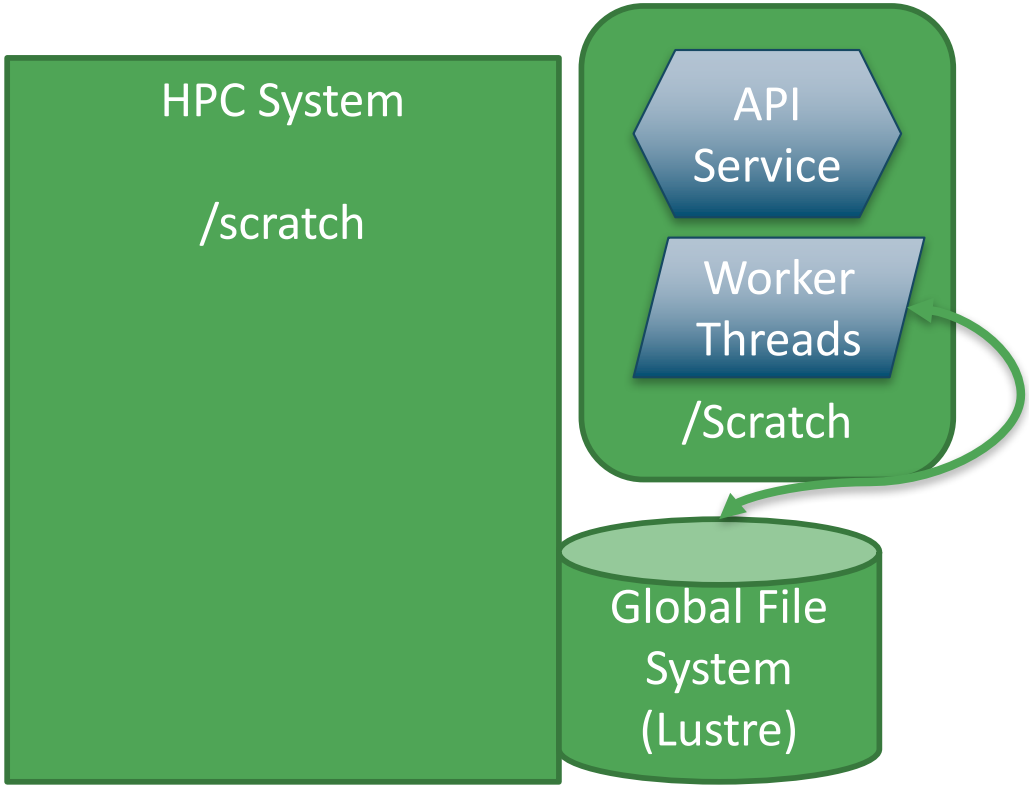
File System flow – Traditional vs Shifter



Deployment Options

1. Gateway service runs on a system with direct access to the cluster file system
2. Gateway service runs on a remote system with ssh access to cluster file system

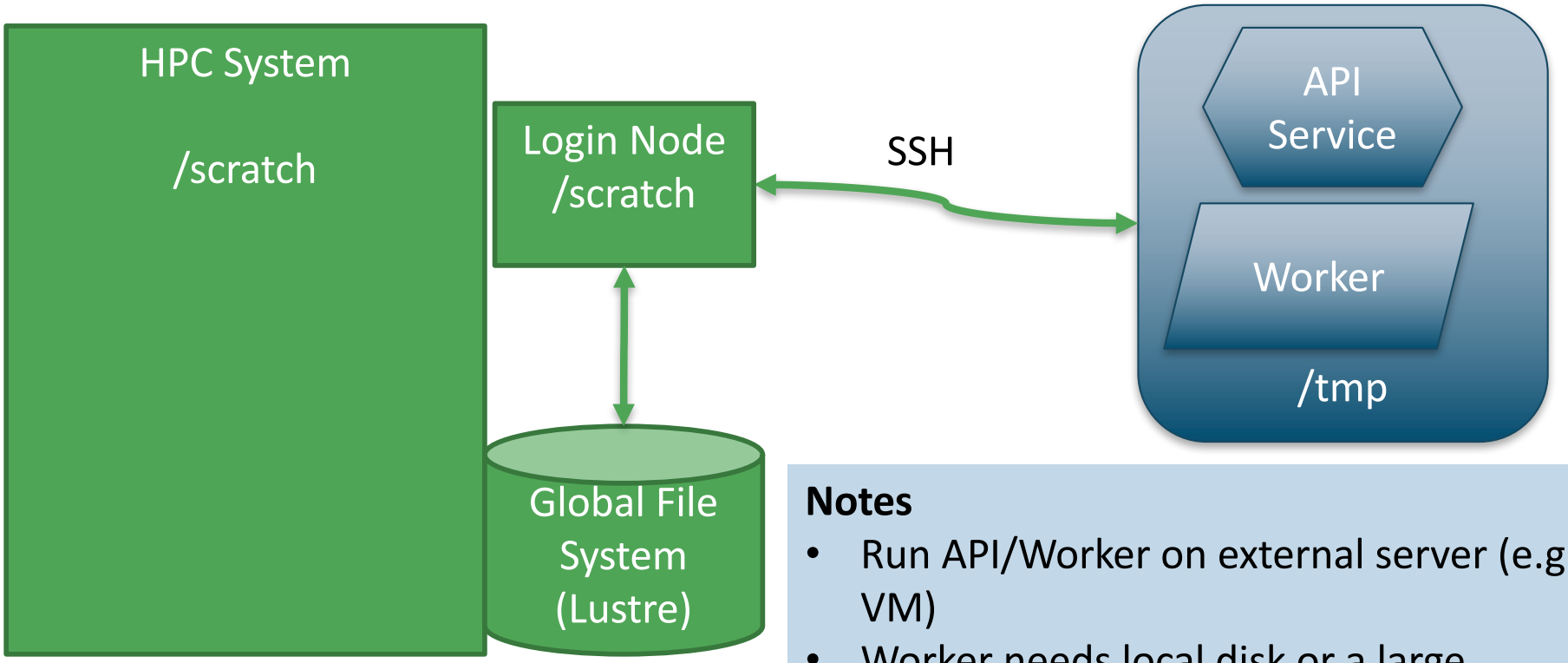
Co-located/Local



Notes

- Typically run on a login-class node
- Worker needs local disk or a large ramdisk space
- This is the easiest model and recommended for most cases

Co-located/Remote



Notes

- Run API/Worker on external server (e.g. VM)
- Worker needs local disk or a large ramdisk space
- Worker needs ssh keys configured to copy image to a node that has the file system mounted.

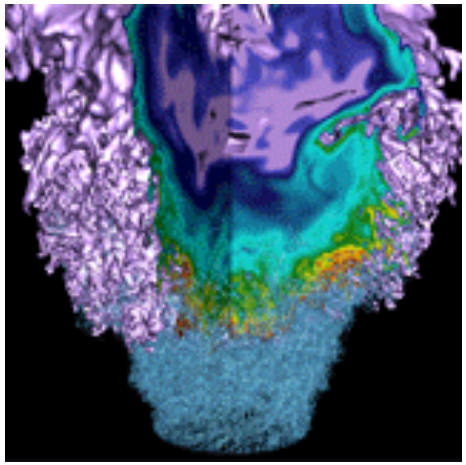
Registries

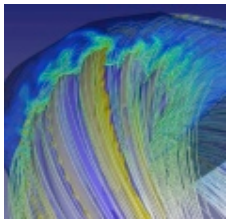
- Adding registries is done in the image manager
- This can be used to add both private and public registries
- Configured in the Locations seciton of the imagemanager.json config file

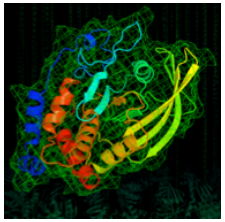
Registries

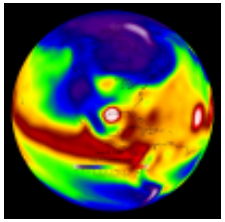
```
…
"Locations": {
        "index.docker.io": {
            "remotetype": "dockerv2",
            "authentication": "http”
        },
"local": {
            "remotetype": "dockerv2",
            "url": "https://localhost:5000",
            "authentication": "http”,
            “sslcacert”: “local.crt”
        }
    },
…
```

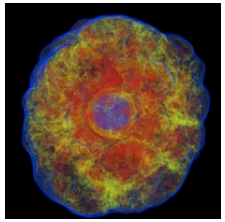
Success Stories

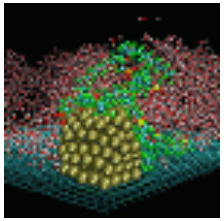










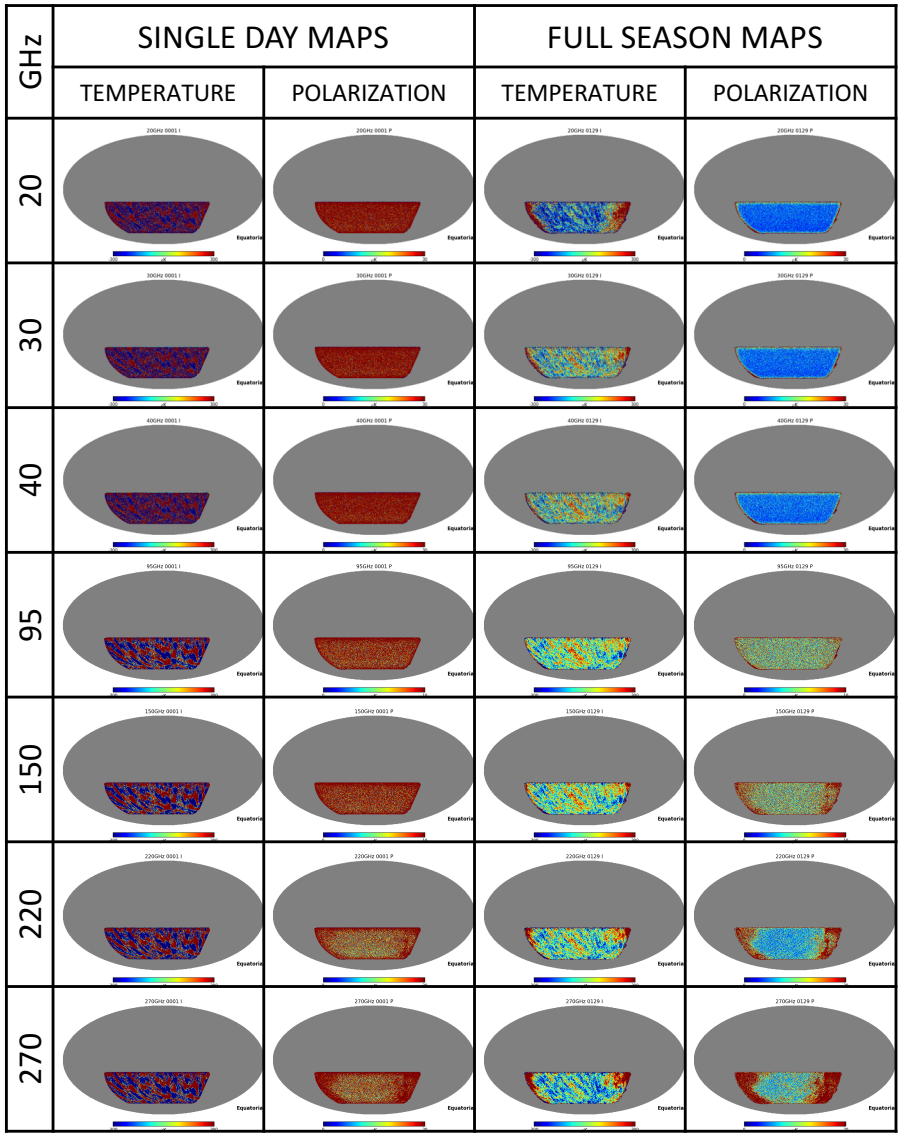


Spark

- "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.

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

Per-Node Write Cache (New Feature)

Per-Node Write Cache provides local disk like functionality but is backed by the Parallel File System.

Nodes/Writers per node	Lustre (MB/s) per writer	Shifter (MB/s) per writer	Real Local Disk (MB/s) per writer
1/1	83	594	416
10/10	87	625	416*
10/20	67	616	165*
10/40	55	589	53*
20/40	71	627	165*
20/80	55	588	53*

Results of a simple "dd" test to simulate writing ~5GB of small transaction I/O (**`dd if=/dev/zero of=$TARGET bs=512 count=10M`**)

* Extrapolated from a single node test