Containers at US ATLAS

Wei Yang
SLAC National Accelerator Laboratory

On behalf of the US ATLAS Computing

Outline

1. ATLAS Computing Model
2. New approach on Grid middleware deployment
3. Containers for scientific computing at Grid sites
4. Containers on HPCs

ATLAS Computing / Grid Computing / Containers

Computing at ATLAS experiment of Large Hadron Collider is

- CPU intensive - simulation
- Data intensive - data processing (reconstruction, etc.)
  - With a long story of data management, etc. not to be discussed here.
- Chaotic - user analysis by 3000+ physicist around the world

Adopted Grid computing model more than a decade ago

- To utilize computing / storage resources at 130+ institutions worldwide
  - and AWS, GCP, Azure
- Works! But operational cost is high due to environment variation

Container is viewed as essential to standardize the environment, reduce cost

Container is also essential to utilize the HPC resources

Containers for Grid / Services deployment

SLATE project: use Kubernetes to manage/deploy containerized services

- Dockerized Grid CE / Data transfer nodes / Cache / Customized user analysis environment, etc.

Future (monitoring, etc.) analytic platform at CERN

- Kubernetes and dockerized Elasticsearch / Kibana, etc.

Individual containerized services / environment:

- Docker and/or Singularity
- High performance general purpose cache (e.g. Xcache)
- Single ATLAS software release environment for (physics) software developers.

Container for Scientific Computing at Grid Sites

Historically batch nodes needs heavy customization to run ATLAS batch jobs

- Every site has to do this
- Customization can all go to a Singularity container
- Singularity is preferred over Docker due to privilege concern

Two methods to start a Singularity container

- Encapsulate batch job payload in a Singularity container
  - Batch configuration based on use cases - BNL and SLAC
  - Work well but may need a configuration for every use case - good for learning, not scalable
- Payload starts container ✔
  - Simple script checks environment and starts container when needed
  - Applications take care of bind mount, etc.

Containers on HPCs

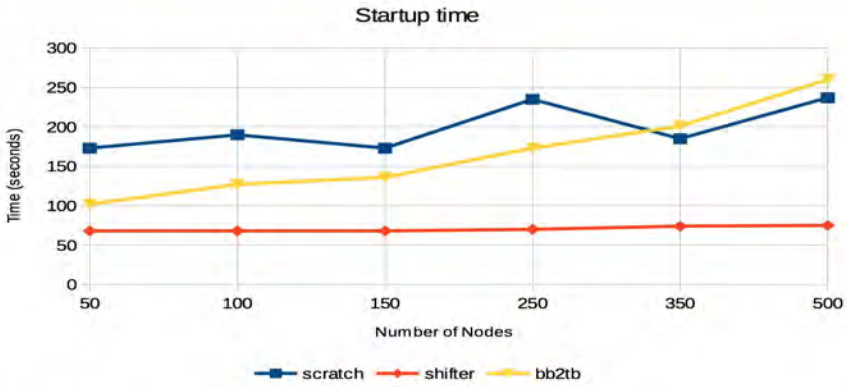
We use containers to address two problems on HPCs

1. Metadata IO (stat(), open()) and small file IO (loading python scripts, shared libraries (.so), etc.) overwhelming HPC shared filesystems.
2. Absence of CVMFS on HPCs
    - A posix file system with global replications, with cache at various level to make it efficient.
    - Replicas on batch nodes are read-only, replicate individual files on demand.
    - ATLAS uses CVMFS to distribute its complete software environment
    - CVMFS requires FUSE on batch node - FUSE (thus CVMFS) is not available on most HPCs
    - Initially we manually build software environment on HPC shared filesystems
        i. costly to install and maintain - also each HPC is different
        ii. Run into problem in 1) above on all HPC sites

Container Tech: Shifter (NERSC), Singularity (OLCF Titan, ALCF Aurora/Theta)

Solution: FAT Container on HPCs

Startup time



- The best scaling obtained with **Shifter**
- **BB** visibly outperforms **Lustre** for small number of nodes
- Very good scaling on **Cori Lustre** comparing to **Edison Lustre** (see slide 13)

U.S. DEPARTMENT OF ENERGY | Office of Science

BERKELEY LAB

Vakho Tsulaia @ Lawrence Berkeley National Laboratory

At Startup time, ATLAS jobs load many python and .so files:

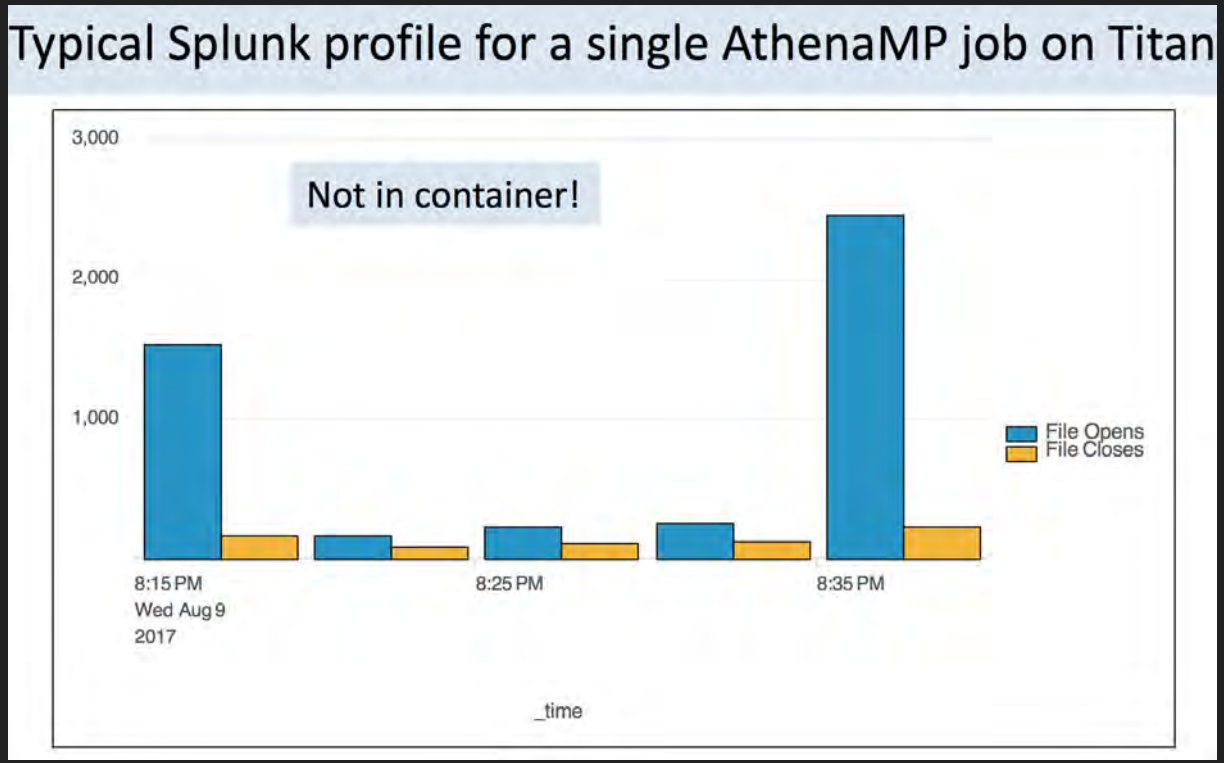
Putting ATLAS software in Shifter image out perform putting software in Burst Buffer (SSD) and shared filesystem

Solution: FAT Container for HPCs, cont'd

CentOS 6 + CVMFS - build CVMFS contents in container image:

1. Have everything we need from CVMFS to run all production job flows
   - Software environment on HPCs is identical to the familiar Grid sites.
   - Container image building process is automated
2. 400GB FAT image after deduplication, not compressed
   - Trimmed 800GB of obsolete SW - can also make much smaller, single SW release containers
   - Think of the FAT container as loopback mount
   - Eliminate all metadata / small file IO to the shared filesystem
   - 400GB image can be mounted/loaded just as fast as 400MB image.
3. Scale up at NERSC Cori KNL:
   - Without container, ran up to 1000 nodes (136K cores) and hit limits on shared file system
   - With container, scaled up to 3000 nodes and still not hitting limits on shared file system

FAT Container and IO Reduction on Titan

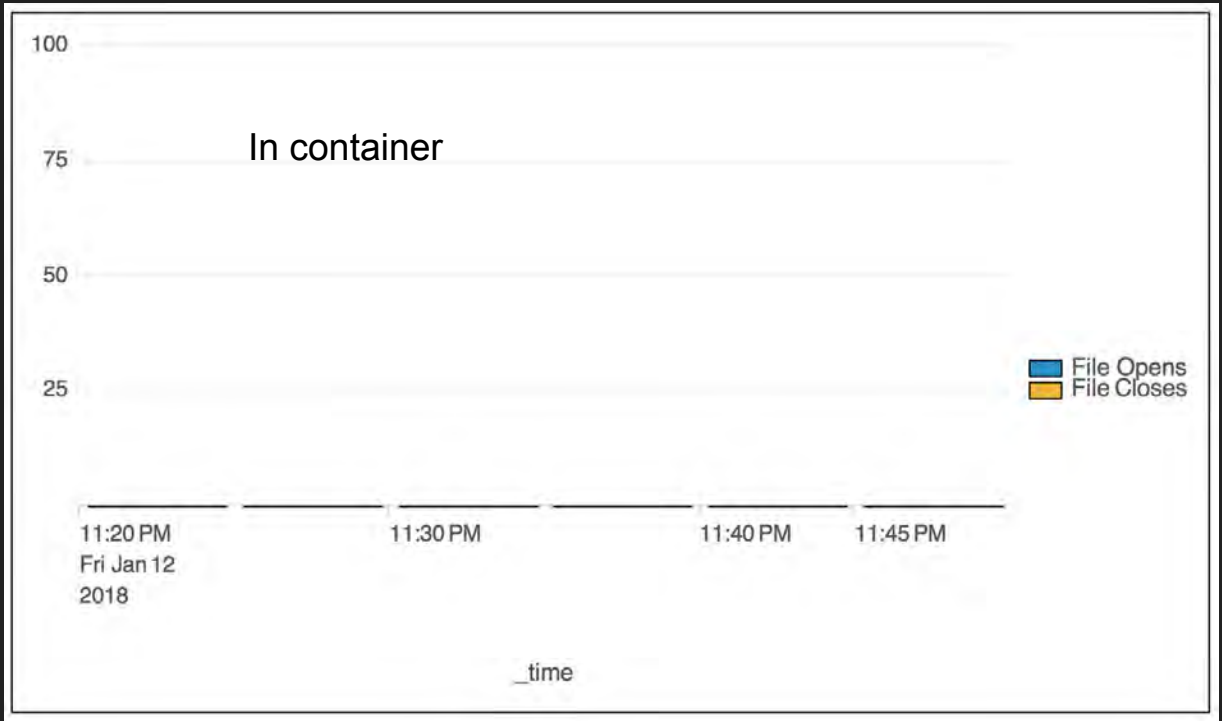


Dalina Oleynik @
University of Texas, Arlington
Sergey Panitkin @
Brookhaven National Laboratory

Single ATLAS job on a Titan node, not in container

File opens and closes against the Titan shared filesystem

FAT Container and IO Reduction on Titan, cont'd



Sergey Panitkin @
Brookhaven National Laboratory
Dalina Oleynik @
University of Texas, Arlington

Single ATLAS job on a Titan node, in container

File opens and closes against the Titan shared filesystem is reduced to almost none!

Summary

ATLAS is working on using Docker/Kubernetes to deploy services

Experimenting various Singularity deployment models on Grid sites

Use FAT container on HPCs

- Standardized / automated software distribution to HPC sites
- Significantly reduced metadata IO and small file IO to HPC shared filesystems.
    - Thus scale up concurrent running jobs.

"Any opinions, findings, conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the Networking and Information Technology Research and Development Program."

The Networking and Information Technology Research and Development (NITRD) Program

Mailing Address: NCO/NITRD, 2415 Eisenhower Avenue, Alexandria, VA 22314

Physical Address: 490 L'Enfant Plaza SW, Suite 8001, Washington, DC 20024, USA Tel: 202-459-9674, Fax: 202-459-9673, Email: nco@nitrd.gov, Website: https://www.nitrd.gov