PROCEEDINGS OF SCIENCE

Developments in Architectures and Services for using High Performance Computing in Energy Frontier Experiments

J. Taylor Childers[a]*, **Lisa Gerhardt**[b]*
Deborah Bard[b], *Doug Benjamin*[c], *Wahid Bhimji*[b]
Steve Farrell[d], *Markus Fasel*[b], *Tom LeCompte*[a],
Jeff Porter[b], *Prabhat*[b], *Vakhtang Tsulaia*[d], *Tom Uram*[a]
[a]*Argonne National Laboratory, Lemont, IL*
[b]*Nuclear Energy Research Scientific Computing, Berkeley, CA*
[c]*Duke University, Durham, NC*
[d]*Lawrence Berkeley National Laboratory, Berkeley, CA*
E-mail: jchilders@anl.gov,lgerhardt@lbl.gov

The integration of HPC resources into the standard computing toolkit of HEP experiments is becoming important as the needs of experiments outpace traditional resources. We will describe solutions that address some of the difficulty in running data-intensive pipelines on HPC systems. Users of NERSC HPCs are benefiting from a newly developed package called "Shifter" that creates Docker-like images and the deployment of the new "Burst Buffers" NVRAM file system designed to offer extreme I/O performance, supporting terabyte-per-second bandwidth and over 10 million IO operations per second. These tools have enabled particle physicists from multiple experiments to routinely run their entire multi-TB CVMFS-based software stacks across tens of thousands of compute cores. In addition, an Edge Service has been developed to provide a uniform interface for HEP job management systems to access supercomputers. It is based on the Python Django framework and is composed of two processes, of which one runs inside the supercomputing environment and one runs outside. It has been used to run over 100 million core-hours of LHC experiment jobs on the Mira supercomputer at the Argonne Leadership Computing Facility.

38th International Conference on High Energy Physics
3-10 August 2016
Chicago, USA

*Speaker.

PoS(ICHEP2016)178

1. Introduction

High Performance Computing (HPC) is becoming an increasingly important resource for HEP experiments. Projected needs for the High Luminosity phase of the LHC vary, but the order of magnitude increase in luminosity will drive a commensurate increase in demand for computing. HPC centers offer a combined total of 15B core-hours per year: Mira at Argonne, a 48k node BlueGeneQ PowerPC system, delivers 6.8B core-hours per year and Cori at NERSC, a 9,304 node Intel Xeon Phi (Knights Landing) and 1600 node Intel Haswell system, will deliver 5.5B core-hours per year. LHC experiments currently use approximately 3B core-hours annually. HEP experiments are already looking to HPC centers to provide a fraction of their computing, with ATLAS using 100M core-hours annually at US HPC centers.

Tapping into HPC resources is often difficult. Getting needed software installed can be painful, transferring data can be a nontrivial process, and higher security requirements can often break production pipelines. This paper will describe several services designed to make running on supercomputers easier. The HPC Edge Service provides a uniform interface for HEP job management systems to access HPC sites. Shifter gives users the ability to create Docker-like images containing their entire software stack and the Burst Buffer file system delivers an extremely fast IO layer. Together these services are lowering the barrier for HEP access to HPC resources.

2. The Edge Service for High Performance Computers

The HPC Edge Service (HES) provides a uniform interface between the computing framework of an experimental collaboration and the resources of HPC facilities. This service is designed to address the following concerns of users and site administrators building on work described in [1, 2, 3]. The Worldwide LHC Computing Grid (WLCG) relies on uniformity across many computing clusters for application compatibility and job management. The varied local operating systems, environments, and batch scheduling systems of HPC facilities breaks this uniformity, making them hard to incorporate into these automated systems. Creating an edge service between the experiment job manager, which is accustomed to the WLCG uniformity, and the HPC facilities provides a single API with which to access many large computing resources.

Unlike the typical WLCG site, some HPC facilities have a higher level of security requiring the use of two-factor authentication (TFA). In addition, large data transfers into and out of the facility are managed through Data Transfer Nodes (DTNs) as apposed to inside worker nodes, which may not have WAN access. The HES addresses these requirements.

2.1 Operation

Figure 1 diagrams the HES, which is primarily composed of two pieces, the Balsam and Argo services. These two services are python-based and employ the Django framework to utilize the web authoring (for monitoring) and database capabilities (for record keeping). Argo and Balsam communicate via message queues hosted on a server that is accessible to both services. Data motion is handled via data servers reachable to both user and HPC facility.

Here is an example job submission. A user submits a job description to Argo via a message sent to the appropriate message queue. Argo receives the message and submits the work to the

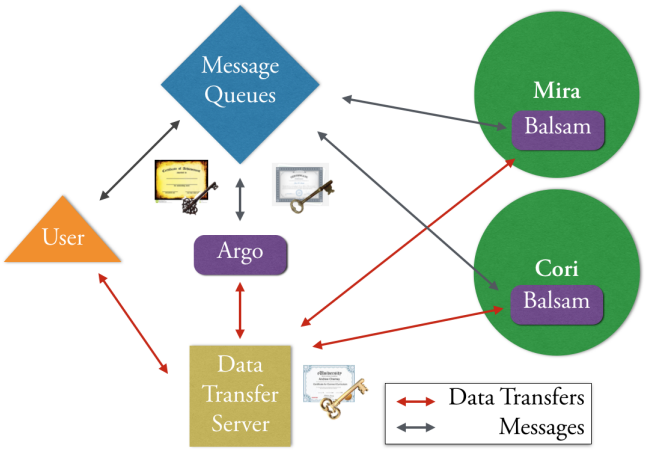

Figure 1: Diagram of the HEP Edge Service. Users upload input data to a commonly accessible data server, in this example a GridFTP server. Then, users submit a job description to a message queue to which Argo is subscribed. Argo parses the job and sends it via message queue to a running Balsam service on an HPC resource such as Mira or Cori. Balsam retrieves input data and when the job is completed send output data. Argo alerts the user that their data is ready. Each service (RabbitMQ or GridFTP) requires a unique openssl certificate and key pair to authenticate with that services in order to limit access to the system.

Balsam message queue associated with the requested HPC resource. Balsam receives the message, optionally retrieves input data from the DTN, and submits the job to the local batch queue. When the job is completed, Balsam transfers the output data to the DTN, and notifies Argo using the message queues that the job has completed. Argo emails the user that the job has completed.

2.2 Security

Throughout this process, it is important to block rogue job submission to HPC facilities because the misuse of resources would have negative effects on HEP access. There are barriers in place to ensure only approved users and applications run on HPC systems. Balsam executes only approved applications effectively limiting the capabilities of illegitimate users. To avoid command injection nothing coming directly from the user is executed on the command line. The message queue server is configured that only approved key/certificate pairs can access message queues, effectively limiting who can submit a job to Argo or Balsam. In addition, the key/certificate pair used to submit jobs to Argo is different from the pair used for communication between Balsam and Argo.

2.3 Performance and Implementation

The HES has been deployed at Argonne since April 2015 to manage event generation jobs on Mira and Edison. The usage from April 2015 to August 2016 is shown in Figure 2. Over two thousand jobs have passed through the HES, representing over 122M core-hours of usage for generating LHC events. At its peak, the HES handled tens of concurrent jobs represented 2M core-hours in HPC usage.

The HES was built on the Django framework in Python because Django offers an SQL database interface and web content authoring tools. Using these tools, HES tracks user jobs using SQL

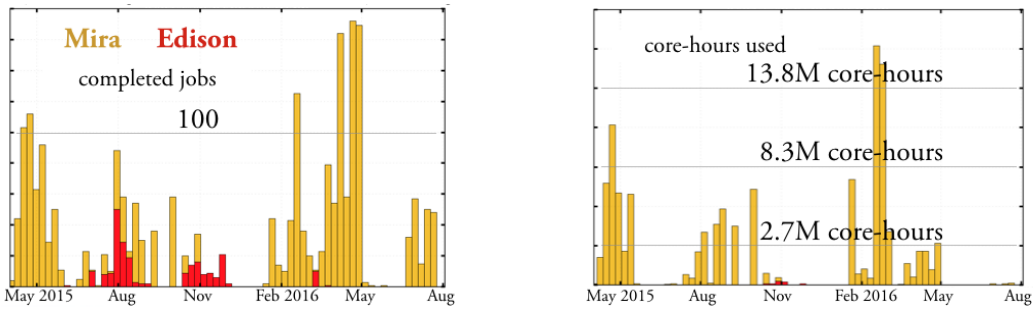

Figure 2: (Left) Number of completed jobs from April 2015 to August 2016. (Right) Number of core-hours used by jobs run using the HES. Both plots show usage from the Mira and Edison supercomputers.

databases and presents job status via searchable web tables. The RabbitMQ messaging systems was used to provide the message queues for these tests and a GridFTP data transfer server was used for the DTN.

3. Shifter: Docker for HPC

Docker [4] is an open source platform for containerizing software stacks and operating systems. Docker images contain everything that is needed to run (code, runtime, system tools, system libraries, etc.) and can be deployed on any hardware using the Docker framework. While Docker has been widely accepted in industry, there are security and scaling considerations that prevent Docker from being installed on HPC systems. This has prompted engineers at NERSC to develop Shifter [8], a framework for deploying images at large scale. Shifter can support Docker images as well as several other standard image formats (vmware, ext4, squashfs, etc.) and is tied into the batch system at NERSC. This allows users to use Docker, or some other image creation framework, to create images in the OS of their choice, install complicated software stacks and run computation in those images on NERSC machines. As a proof of concept, Shifter has been used to construct images that mimic the CVMFS functionality for ATLAS, ALICE, and CMS. This was done using python code called uncvmfs [5] to deduplicate the entire CVMFS software stack and copy it into an ext4 image. Even after deduplication of files, the resulting images were quite large: about 3.5 TB in size with more than 50 million files and directories for the ATLAS CVMFS repository. The ext4 image was then compressed into a 300 GB squashfs image. This image was used for ATLAS Geant4 simulations and showed excellent scaling in start up time out to 500 nodes on Cori (Fig. 3).

The ALICE group at NERSC has been investigating using Shifter and parrot (a toolkit that can be used to access a CVMFS repository when CVMFS is not installed on your system, see [6] for more details) to access their CVMFS repository. They compared the startup times for parrot pulling from a CVMFS stack installed on Shifter and a stack installed on Cori's dedicated 30 PB Lustre file system. They found that access from the Lustre file system were about 30% slower than from Shifter, but were still under a minute for concurrencies out to 400 processes. At higher concurrencies, the delays from the Lustre file system began to become non-negligible compared to a purely Shifter-based solution. For now, ALICE plans to run at a low enough concurrency that

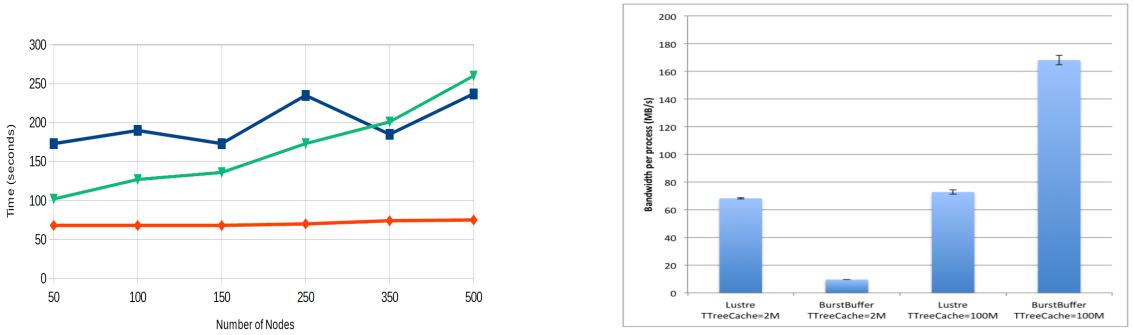

Figure 3: (Left) Load time in seconds for ATLAS simulation as a function of number of nodes for Shifter (red), Burst Buffer (green), and Lustre file system (blue). (Right) Bandwidth per process for the ATLAS analysis application with the default 'TTreeCache' size of 2M and 100M.

using parrot to read from Lustre is sufficient, but they are exploring a more Shifter-based solution for higher concurrency runs.

Shifter has opened the door for groups that require complicated software stacks like CVMFS to run. A number of astronomy and HEP groups, including ATLAS, ALICE, and CMS, are investigating how Shifter can be incorporated into their workflows to run at NERSC and Cray will incorporate Shifter functionality into future Cray systems. Shifter has been released through an open source BSD license [7] and has been successfully installed on several systems around the world.

4. Burst Buffer: A Super Fast IO Layer

Phase 1 of Cori was deployed with a Burst Buffer made up of 900 TB of NVRAM-based storage [9]. The state-of-the-art SSD hardware has a total aggregate bandwidth of roughly 900 GB/s and is well suited for high-bandwidth streaming reads and writes (e.g. check pointing) and complex I/O patterns, including those with high I/O operations per second such as those typically found in high-energy physics analysis. Data on the Burst Buffer can be striped across multiple server nodes, to achieve high bandwidth. The data on the Burst Buffer is managed with Cray's DataWarp software, which presents users with a POSIX interface to the file system. Access to the Burst Buffer is controlled via the batch system. Users can request persistent or non-persistent reservations of space on the Burst Buffer using Slurm batch directives. Input data can be staged in from the Lustre file system and jobs will start once the data has finished transferring to the Burst Buffer.

The ATLAS group at NERSC investigated the effects of using the Burst Buffer for simulation and for data analysis and found improvements for both. A relatively I/O intensive set of ATLAS simulations were scaled out to 9600 cores, and event time was measured when outputting to the Burst Buffer or to the Lustre file system. The Burst Buffer outperformed the Lustre file system at low node concurrences and was comparable to Luster at high concurrences.

For the data analysis workload on 475 GB xAOD dataset, the Burst Buffer did worse than Lustre when reads from the root file were done in small sizes (the TTreeCache=2M in Fig 3). This is because Lustre has some client side caching of data, while the Burst Buffer does not. Caching

functionality is expected to be included in future releases of the DataWarp software, so reads of this type are expected to improve in the future. However, increasing the TTreeCache to a value that is more appropriate for this type of analysis led to the Burst Buffer significantly outperforming the Lustre file system (the TTreeCache=100M in Fig 3).

Since it has been demonstrated that I/O heavy workloads can run well on the Burst Buffer, ATLAS is working on incorporating this layer into their future production.

4.1 Summary

High Energy Physics experiments have begun to use High Performance Computers and have quickly reached the annual usage of 100M core-hours. The tools presented here will help experiments and facilities manage the growth in usage to the 1B core-hour level and expand the accessibility of these resources.

4.2 Acknowledgments

This research used resources of the Argonne Leadership Computing Facility, a DOE Office of Science User Facility supported under Contract DE-AC02-06CH11357, and the National Energy Research Scientific Computing Center, a DOE Office of Science User Facility supported under Contract No. DE-AC02-05CH11231. A grant of computing time on these facilities was awarded through the ASCR Leadership Computing Challenge (ALCC). This research was supported by the Center for Computational Excellence under the DOE Office of High Energy Physics.

References

[1] J T Childers, T D Uram, T J LeCompte, M E Papka, D P Benjamin, 'Achieving production-level use of HEP software at the Argonne Leadership Computing Facility,' JOP: Conf. Series, 664, 6, 062063, doi:10.1088/1742-6596/664/6/062063, 2015

[2] J T Childers, T D Uram, T J LeCompte, M E Papka, D P Benjamin, 'Simulation of LHC events on a millions threads,'JOP: Conf. Series, 664, 9, 092006, doi:10.1088/1742-6596/664/9/092006, 2015

[3] J T Childers, T D Uram, T J LeCompte, M E Papka, D P Benjamin, 'Adapting the serial Alpgen parton-interaction generator to simulate LHC collisions on millions of parallel threads,' Accepted in Comp. Phys. Comm., doi:10.1016/j.cpc.2016.09.013, 2016

[4] Docker web site: `https://www.docker.com/`

[5] Uncvmfs git hub repository: `https://github.com/ic-hep/uncvmfs`

[6] Parrot Connector to CernVM-FS: `https://cernvm.cern.ch/portal/filesystem/parrot`

[7] Shifter git hub repository: `https://github.com/NERSC/shifter`

[8] R S Canon and D Jacobsen 'Shifter: Containers for HPC' Proceedings of Cray Users Group, `https://cug.org/proceedings/cug2016_proceedings/includes/files/pap103.pdf`, 2016

[9] W Bhimji and others 'Accelerating Science with the NERSC Burst Buffer Early User Program' Proceedings of Cray Users Group, `https://cug.org/proceedings/cug2016_proceedings/includes/files/pap162.pdf`, 2016