Homogenizing OSG and XSEDE: Providing Access to XSEDE Allocations through OSG Infrastructure

Suchandra Thapa, Robert W. Gardner, Jr
University of Chicago
Chicago, IL, USA
sthapa@ci.uchicago.edu,rwg@uchicago.edu

David Lesny
University of Illinois Urbana-Champaign
Urbana, IL, USA
ddl@illinois.edu

Dirk Hufnagel, Ken Herner
Fermi National Laboratory
Batavia, IL, USA
hufnagel@fnal.gov,kherner@fnal.gov

Mats Rynge
University of Southern California
Los Angeles, California, USA
rynge@isi.edu

ABSTRACT

We present a system that allows individual researchers and virtual organizations (VOs) to access allocations on Stampede2 and Bridges through the Open Science Grid (OSG), a national grid infrastructure for running high throughput computing (HTC) tasks. Using this system, VOs and researchers are able to run larger workflows than can be done with OSG resources alone. This system allows a VO or user to run on XSEDE resources (with their allocation) using the same framework used with OSG resources. The system consists of two parts: the compute element (CE) that routes workloads to the appropriate user accounts and allocation on XSEDE resources, and simulated access to the CernVM Filesystem (CVMFS) servers used by OSG and VOs to distribute software and data. This allows jobs submitted through this system to work on a homogeneous environment regardless of whether they run on XSEDE HPC resources (like Stampede2 and Bridges) or OSG.

CCS CONCEPTS

• **Computer systems organization → Grid computing**;

KEYWORDS

OSG, XSEDE, distributed data access, CVMFS, CMS, ATLAS, TACC, Stampede2, Bridges, PSC

ACM Reference Format:
Suchandra Thapa, Robert W. Gardner, Jr, David Lesny, Dirk Hufnagel, Ken Herner, and Mats Rynge. 2018. Homogenizing OSG and XSEDE: Providing Access to XSEDE Allocations through OSG Infrastructure. In *PEARC '18: Practice and Experience in Advanced Research Computing, July 22–26, 2018, Pittsburgh, PA, USA.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3219104.3219157

PEARC 2018, July 22–26 2018, Pittsburgh, USA

ACM ISBN 978-1-4503-6446-1/18/07…$15.00
https://doi.org/10.1145/3219104.3219157

1 INTRODUCTION

The Open Science Grid (OSG) [11] provides an infrastructure for executing distributed high throughput computing (DHTC) workflows across many loosely coupled computational facilities. As opposed to a traditional HPC resource, jobs on OSG tend to use relatively few cores (1-8) and can run without any communication between jobs. When a researcher submits a job, he or she does not know *a priori* at which computing site the job will run. Rather they specify functional job requirements such as the memory needed, number of cores (for multi-core applications), or whether there is support for containers at the site. A metascheduling system then attempts to match their job request to available resources. Although OSG tracks and monitors accounting information for jobs, OSG does not use this information to limit user job execution. Facilities may prioritize jobs from resource owners and/or projects, but users running opportunistically do so in a fair share (and potentially preemptable) fashion regardless of the total wall hours users may have used through OSG.

On the other hand, the Extreme Science and Engineering Discovery Environment (XSEDE) [15] requires users to access resources through granted allocations in order to run jobs and workflows on its resources. Users must apply for an allocation which allows them to use a limited amount of service units (SUs) on specific resources. In order to run workflows on a resource, they must log in via SSH to each individual resource and submit their workflow there. There is no provision for running workflows across multiple resources. Once a researcher exhausts their allocation, they are not able to run any additional jobs.

Historically, users have not been able to submit jobs from OSG to XSEDE resources. Although there is a gateway that allows users with XSEDE allocations to submit jobs to OSG, a general mechanism to do the reverse does not exist.

In this paper, we present a system that bridges the differences between OSG and XSEDE compute environments and allows users to treat XSEDE resources as if they were OSG resources. This system allows virtual organizations (VOs) like ATLAS [2] and CMS [4] to consume allocations on XSEDE resources as if the resources were an OSG resource. In order to achieve this, the system had to handle several issues: software access and distribution (the size of experiment-specific software releases with their dependencies is rather large), authenticating to XSEDE resources, and routing jobs

to the correct allocation on those resources. We've configured this system with access to accounts on Stampede2 [14] and Bridges [10],

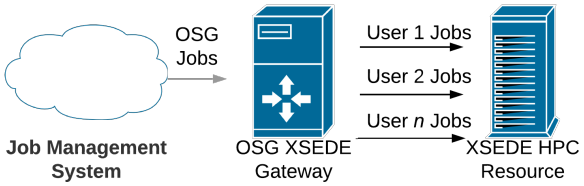

Figure 1: High level architecture for OSG - XSEDE gateway

2 BACKGROUND

Most OSG researchers run workflows by utilizing a pilot model [12]. This allows users to easily and efficiently run their workflow on available sites. Users submit their workflows to a central submit system. The system then matches jobs within the workflow to pilots running on computational facilities. Once a job is matched to a pilot, the pilot then downloads the job information, inputs, and binaries (the job "payload") and executes it.

A central "factory" such as GlideinWMS[13], HEPCloud [6], or PanDA [8] is responsible for tracking the number of idle jobs on the central submit system queue. If the number of idle jobs exceed certain thresholds, the factory then submits pilot jobs to batch systems at OSG computational facilities. All facilities on OSG use the HTCondor-CE [3] compute element to accept incoming jobs and submit them to a local batch manager so the factory can submit pilot jobs to any OSG facility using the same method.

Although previous work has been done to incorporate XSEDE resources into workflows run by OSG users [16, 17], these attempts were either specific to a given researcher or required resources unique to a given XSEDE resource. LIGO was able to use 2M SUs on Stampede [17] as part of their analysis campaign in 2016. However this infrastructure was specific to LIGO's processing framework. The Comet Virtual Clusters [16] was used to by several VOs (LIGO, CMS, Xenon1T) to run workflows on OSG and the Comet system at the San Diego Supercomputer Center (SDSC). However, these jobs were limited to using a portion of the resources set aside on Comet. In addition, the virtual clusters required several services to be commissioned at SDSC in order to operate.

3 OSG-XSEDE GATEWAY: HTCONDOR-CE

OSG uses HTCondor-CE [3] to allow jobs to run on resources. The typical mode of operation is to authenticate and authorize incoming jobs using a X.509 proxy certificate [19]. The HTcondor-CE then submits the pilot job to the local batch manager (e.g. HTCondor, SGE, Slurm, PBS). By using the BOSCO [18] module for HTCondor-CE, this can be altered. Instead of directly calling the submit commands for a batch manager, the HTCondor BOSCO module will submit jobs to a remote system. BOSCO does this by using SSH to login to a specified host, transferring any input files to the host, and then calling the appropriate submit command. The HTCondor BOSCO module will then periodically log in to the remote system to track the job's status and update its own records for the job. Once the job has completed, BOSCO will then transfer the job outputs back. We utilized this method of operation to allow OSG jobs to be submitted to XSEDE resources.

This implementation allows for users on OSG to run their workflows seamlessly on XSEDE and OSG resources. We set up and ran a HTCondor-CE system for each XSEDE resource that will accept jobs from OSG VOs. We then created entries for OSG VOs on the HTCondor-CE for those that have an allocation on the corresponding XSEDE resource. This allows OSG users with XSEDE allocations to submit jobs while preventing OSG users without allocations from running jobs on XSEDE resources. The appropriate pilot factory can then submit pilots to these HTCondor-CEs and run jobs on the XSEDE resource for VOs.

3.1 Routing Jobs to Allocations

A key requirement for the OSG-XSEDE gateway is to properly route incoming jobs. When an user submits a job through the gateway, the job must run using that user's account and allocation. In addition, the gateway must reject jobs from users without allocations. Finally, adding or removing an OSG to XSEDE routing for an user should be straightforward and easy.

The standard installation and configuration for HTCondor-CE and BOSCO is to use a single account to run all incoming jobs. Naturally, this is incompatible with properly running jobs on XSEDE resources. We solved this problem by modifying the job routing in HTCondor-CE and by modifying the job submission scripts for each account that is used to submit jobs.

We used the JobRouter component of HTCondor-CE to send jobs to the correct accounts on XSEDE resources. Figure 2 shows how the job routing works in our modified setup. All incoming jobs are submitted with a X.509 proxy that authenticates the origin of the job. We configured the JobRouter to examine the VO name attribute of the proxy. Based on that, the JobRouter was set to route incoming jobs to the `cms` or `atlas` user accounts. In addition, we modified the BOSCO configuration for each user account so that BOSCO used different usernames and public keys when logging into corresponding account on the XSEDE resource. Jobs submitted with a proxy that does not match a configured entry are rejected by HTCondor-CE. Adding or removing support for a VO, consists of modifying a single configuration file for the JobRouter and another configuration file for the X.509 proxy to user mapping.

Finally, we made modifications to the BLAH[9] submit scripts used for each user. HTCondor-CE installs a set of submit scripts in user accounts on the individual resources (e.g. the `uscms` account on Stampede2). HTCondor-CE uses these submit scripts to translate job parameters (e.g. memory or cores required) into a submit file that can be understood by the batch manager on the local resource (e.g. SLURM, PBS, SGE). We modified these scripts to generate submit files incorporating site and user specific requirements for jobs. For example, running jobs on a particular SLURM partition or loading needed modules such as Singularity [7] into the environment for the user job.

3.2 Authenticating to Resources

The HTCondor-CE gateway requires the ability to login into the appropriate XSEDE resource in order to run OSG jobs using a

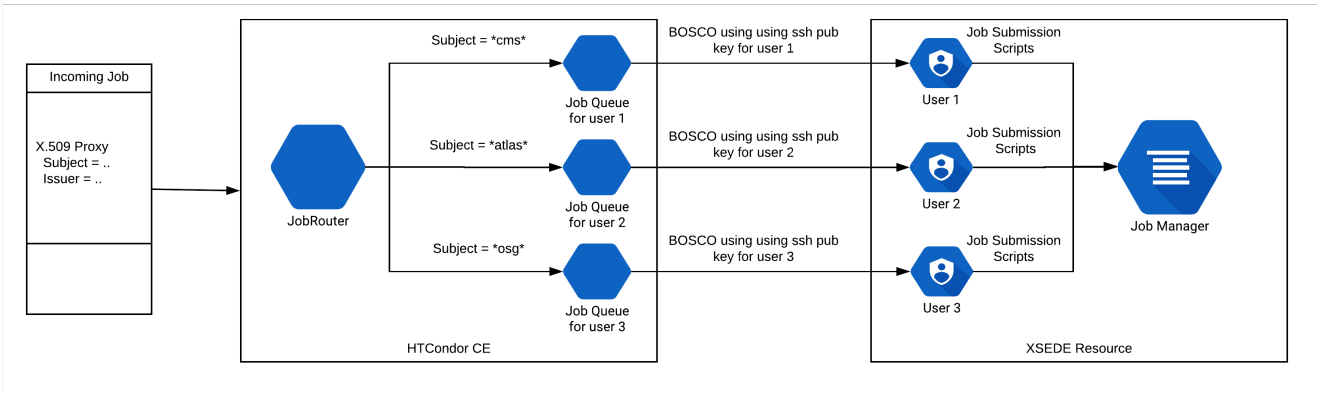

Figure 2: Job Routing from OSG gateway to XSEDE resource

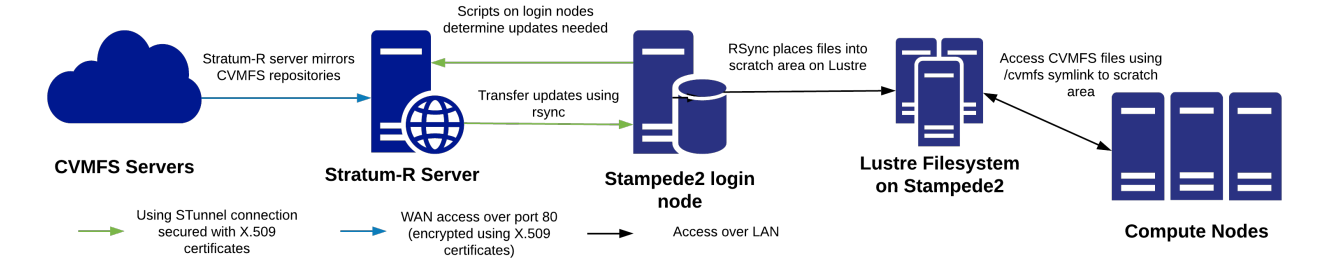

Figure 3: Stratum-R Architecture, the system which replicates the contents of a CVMFS repository to local filesystem.

SSH key pair. HTcondor-CE uses a SSH connection in order to transfer input files and submit jobs. The SSH connections are then subsequently used to monitor, remove jobs, and retrieve job output from the XSEDE resource's batch system. The gateway does not support password based SSH logins or other SSH login methods such as Kerberos or GSI authentication.

For Bridges, we configured the gateway to log into a community account. A community account is required since the OSG-XSEDE gateway can use the account to submit jobs on behalf of any member of a VO. After the community account was created, we registered the appropriate SSH public key through the Pittsburgh Supercomputing Center (PSC) SSH Key management system. The gateway was then able to use corresponding SSH key to log into Bridges.

However, several resources such as Stampede2 require multi-factor authentication (MFA) using a one time token generated by a hardware device or separate software application in order to login.

For access to Stampede2, we also needed to use community accounts for access. Here, we requested separate community accounts for the ATLAS and CMS VOs. In order to satisfy MFA requirements, the system initially used the IP of the OSG-XSEDE gateway as a factor in conjunction with a SSH key pair. Since the OSG gateway would only login from a fixed IP address, logins to the community account can be restricted to attempts originating from that IP address. The gateway's IP and the SSH constituted two different factors satisfying Stampede's MFA requirements.

Later on, we requested and obtained an MFA exception from the TACC staff in order to allow to greater flexibility to allow alternate gateways that we operate to submit jobs as well. The exception allowed us to login using just a SSH key. For established projects that can justify the need for a MFA exception, this is a viable alternative to using IP address/SSH keys as factors to satisfy MFA requirements.

4 SOFTWARE ACCESS AND DISTRIBUTION

Most virtual organizations and a large majority of users on OSG have moved to distributing and accessing software using the CernVM Filesystem (CVMFS) [1]. CVMFS distributes files and directories through a hierarchy of servers and caches. Communications between CVMFS servers and a CVMFS client use the HTTP protocol. The CVMFS client then uses FUSE to convert the filesystem information from a CVMFS server to a read-only POSIX filesystem available to local applications.

For performance reasons, the CVMFS client maintains a local cache of portions of the filesystem that have been used. In addition, a site installation of the Squid HTTP proxy is used to cache data for multiple clients within a computational facility.

In the last few years, CVMFS has been increasingly used by virtual organizations participating with OSG to distribute software

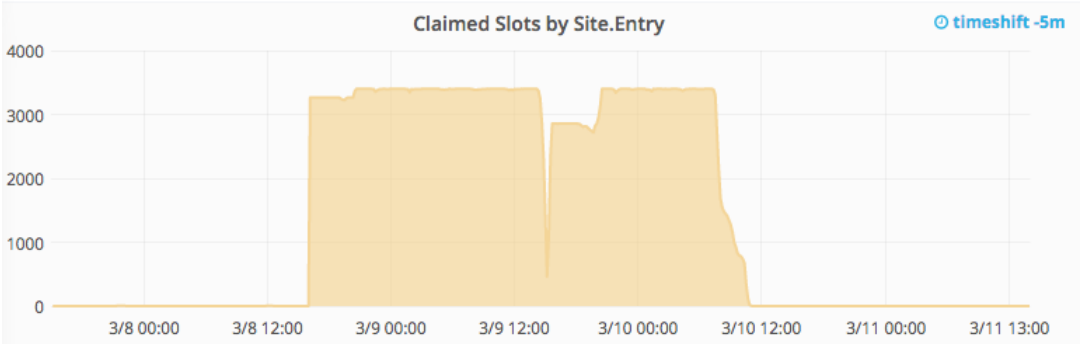

Figure 4: Cores used on Stampede2 by CMS - HEPCloud Factory Monitoring

Figure 5: CMS Job Status on Stampede2

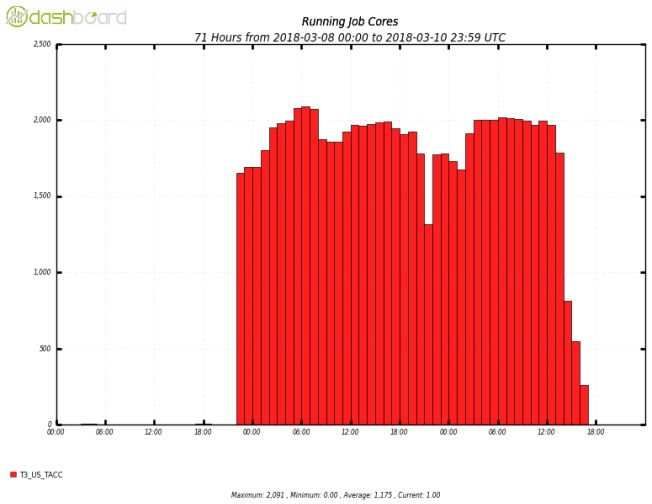

Figure 6: Cores used on Stampede2 by CMS - CMS Monitoring

and configuration to computational facilities. In order to allow CMS and ATLAS jobs to run on XSEDE resources, we needed to provide access to repositories currently distributed through CVMFS. We used two different methods to access software on CVMFS on Bridges and Stampede2.

4.1 Access Through the CVMFS Client

On Bridges, we were able to provide access to software on CVMFS using the CVMFS client. The PSC admins installed FUSE, the CVMFS client, automount, and configuration files provided by OSG on each compute node. In addition, a modified Squid server was installed at Bridges to provide a resource wide cache. Once installed and started the CVMFS client accepts access attempts to files and directories within the `/cvmfs` mountpoint and provides the appropriate information.

4.2 Access Through Stratum-R

On Stampede2, we were not able to use the CVMFS client to access repositories on the compute nodes due to FUSE not being available. Instead, we used a system called Stratum-R to provide jobs access to these repositories. Figure 3 shows the high level architecture of the Stratum-R system.

The Stratum-R server runs two CVMFS services: a CVMFS server and CVMFS client. The CVMFS server component integrates with existing CVMFS servers and replicates full copies of repositories from these servers. The CVMFS client then talks to the locally running CVMFS server and creates a read-only filesystem with these repositories under the `/cvmfs` mountpoint.

The Stratum-R client then uses rsync to transfer files from the Stratum-R server to the scratch area on Stampede2. The client compares files installed in the scratch area with the latest version of these files under `/cvmfs` and then updates the scratch area to match the latest repository version. In order to prevent a man-in-the-middle attack, the rsync connection is secured with stunnel using X.509 certificates. Since CVMFS repositories are updated multiple times a day, the Stratum-R client is run periodically using a cronjob.

5 CMS WORKFLOWS

CMS workflows are submitted through the regular CMS systems and jobs are submitted though CMS HTCondor schedds connected to HEPCloud. Once that happens, the HEPCloud factory sends pilot jobs to Stampede2 or Bridges through the OSG-XSEDE gateway.

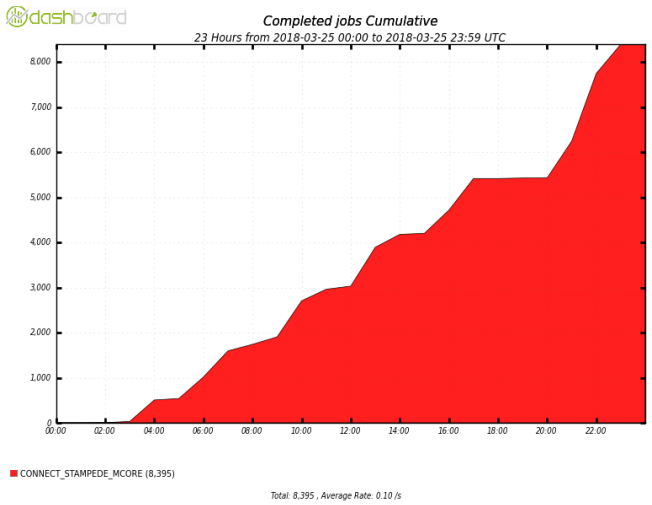

Figure 7: Cumulative # of jobs run on Stampede2 by ATLAS on 3/25/2018

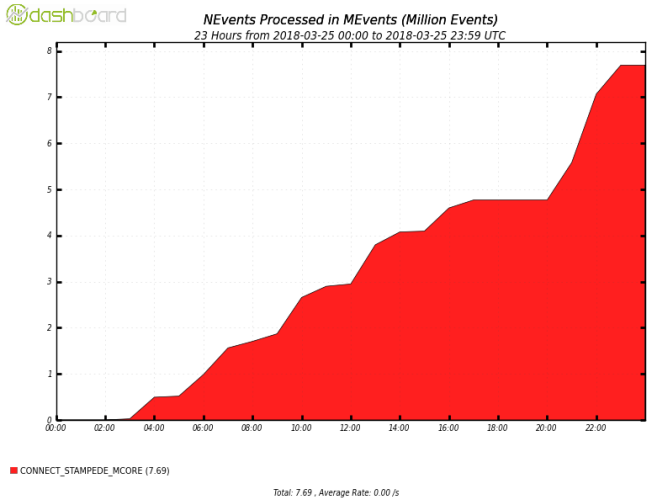

Figure 8: Cumulative # of events processed on Stampede2 by ATLAS on 3/25/2018

When these pilots start running they match and run jobs queued in the CMS schedd.

To provide a CMS-compatible runtime environment, jobs use Singularity with standard CMS containers provided via CVMFS. We access the CMS software and site configuration via CVMFS, conditions through locally provided Squid proxies and read job input data remotely via XRootD from Fermilab (and possibly other CMS sites).

5.1 CMS-Stampede2 Integration

We needed to change several job parameters to run CMS jobs optimally on Stampede2. Incoming jobs were set to run only on the

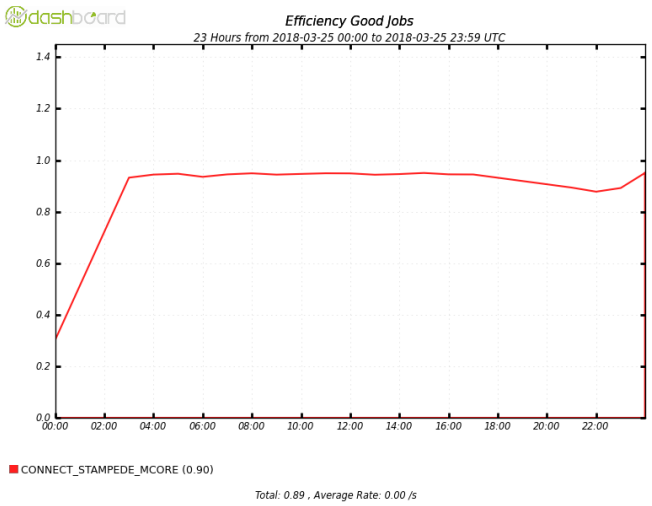

Figure 9: Efficiency of ATLAS jobs on 3/25/2018

Knights Landing (KNL) nodes. Although the KNL nodes have 272 cores available, jobs only used 68 cores due to memory limitations. In addition, incoming jobs only claimed a single node in order to simplify testing and integration. With these changes, jobs could use a total of 3400 KNL cores simultaneously.

We then ran several CMS workflows ranging from simple hello world type jobs to workflows with 10,000 Monte Carlo simulations. We were able to consistently see jobs utilizing over 3000 cores (Figures 4 and 6). The difference in the number of cores is due to differences between how HEPCloud Factory and CMS dashboard record available job slots. Figure 5 shows the the final status Monte Carlo workflow with all 10000 jobs having been successfully completed.

5.2 CMS-Bridges Integration

We made similar modifications to job parameters on Bridges. All jobs were set to use 8-core/36GB RAM slots on the RSM-Shared partition. These jobs ran on shared nodes with 28 cores and 128GB of RAM. The submit scripts were also modified to load Singularity so that jobs could get a Scientific Linux 6 environment while running. As with the Stampede2 integration, a site entry was added to the HEPCloud integration factory. As with Stampede2, several workflows from hello world jobs to progressively more complicated workflows were run.

6 ATLAS WORKFLOWS ON STAMPEDE2

ATLAS uses a job management system called PanDA to manage and run jobs on OSG sites. We used a variation of the OSG-XSEDE gateway to allow PanDA to run pilots on Stampede2. In this variation, PanDA submitted pilots to a standard HTCondor-CE installation. Once pilots were submitted, they then flocked [5] to a server that used BOSCO to SSH to the Stampede2 login node.

Two types of production jobs were run on Stampede2: Event Service (ES) jobs and standard jobs. ES jobs process blocks of events and can be preempted or stopped at any time. Standard jobs must

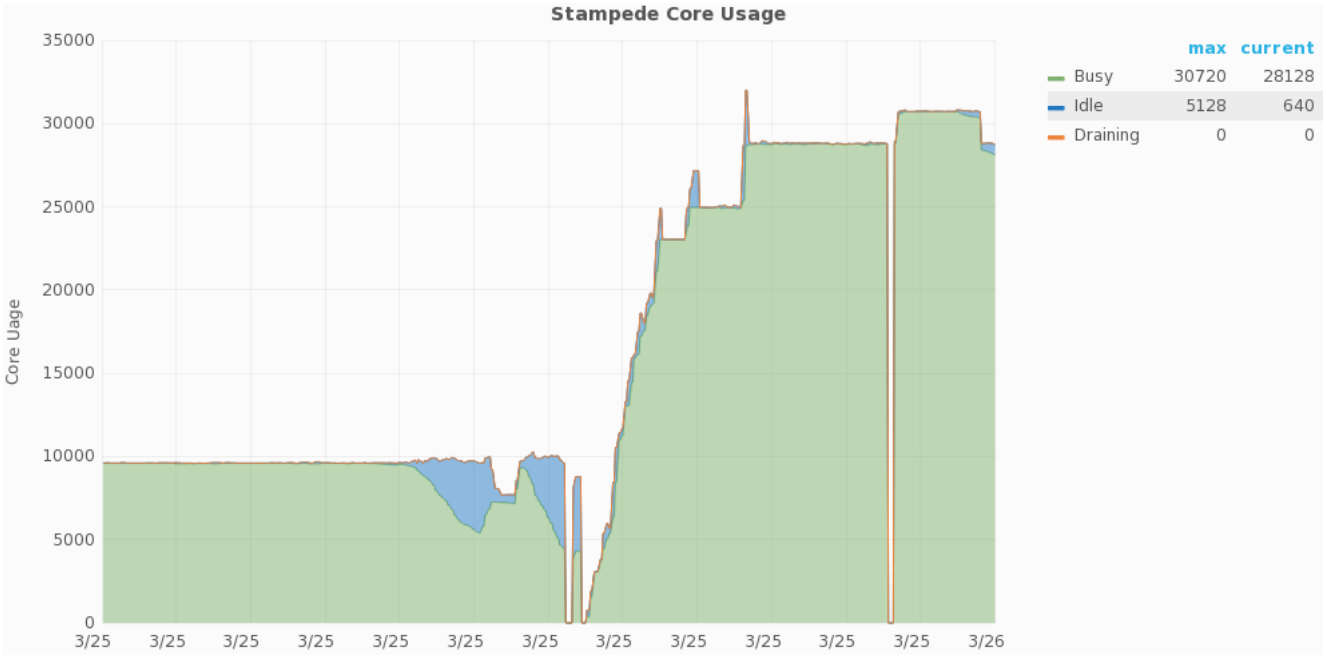

Figure 10: Cores on Stampede2 used by ATLAS vs. Time

run to completion. The two job types can be combined to avoid wasting CPU cycles by running ES jobs to fill in gaps where standard jobs can't run.

6.1 ATLAS-Stampede2 Integration

We needed to change several job parameters in order to run ATLAS jobs more optimally on Stampede2. First, due to limitations on the number of jobs allowed in the queues on Stampede2, the submit files generated were changed so each job claimed multiple nodes and the launcher command was used to run pilots on all of the nodes. Second, the mix of ATLAS jobs was altered to utilize the cores and memory more efficiently. Finally, jobs were set so that they ran on Skylake nodes rather than KNL nodes in order to get better I/O performance.

We also utilized software called pCache to cache data that jobs request during processing. When a job requests data, the copy command checks pCache to see if a cached copy is available. If the data is not available locally, it is requested from a remote resource and is placed into the local cache. This allows us to minimize the network bandwidth that jobs use and to increase the CPU efficiency by reducing I/O latency.

Once that was done, we were able validate and run production jobs from ATLAS. Figure 10 shows monitoring information for cores available to ATLAS from Stampede2. During March 25, 2018, we were able to obtain and run on over 30000 cores. Figure 7 shows the cumulative number of jobs run during that day. ATLAS was able to run over 8000 jobs during this time period. The cumulative number of events generated (figure 8) during that day was almost 8 million events. Figure 9 shows that the jobs also averaged more 89% CPU efficiency.

7 CONCLUSIONS AND FUTURE WORK

We have presented a system that allows XSEDE HPC resources to be integrated into the OSG. This system allows VOs to incorporate allocations on XSEDE resources into their existing job management systems and to seamlessly utilize those allocations alongside resources on the OSG.

The plans for moving forward includes further work on integrating the CMS workflows with Bridges and Stampede2 and moving CMS usage of these resources into production status. Additionally, we plan on integrating Comet, which should be fairly easy as CVMFS is already natively installed on the resource, and work with some individuals with XSEDE allocations to get them access to XSEDE via OSG.

ACKNOWLEDGMENTS

The work is supported by the National Science Foundation under Grant No.: NSF PHY 1148698.

This work used the Extreme Science and Engineering Discovery Environment (XSEDE), which is supported by National Science Foundation grant number OCI-1053575. Specifically, it used the Bridges system, which is supported by NSF award number ACI-1445606, at the Pittsburgh Supercomputing Center (PSC).

The authors would like to acknowledge Derek Simmel (PSC) , Anirban Jana (PSC), and Todd Evans (TACC) for their assistance and the Texas Advanced Computing Center (TACC) at The University of Texas at Austin for providing HPC, storage, and network resources that have contributed to the research results reported within this paper.

REFERENCES

[1] C. Aguado Sanchez, J. Bloomer, P. Buncic, L. Franco, S. Klemer, and P. Mato. 2008. CVMFS - a file system for the CernVM virtual appliance. In *Proceedings of XII Advanced Computing and Analysis Techniques in Physics Research*. Article 52, 52 pages.

[2] W. W. Armstrong et al. 1994. ATLAS: Technical proposal for a general-purpose p p experiment at the Large Hadron Collider at CERN. (1994).

[3] B Bockelman, T Cartwright, J Frey, E M Fajardo, B Lin, M Selmeci, T Tannenbaum, and M Zvada. 2015. Commissioning the HTCondor-CE for the Open Science Grid. *Journal of Physics: Conference Series* 664, 6 (2015), 062003. http://stacks.iop.org/1742-6596/664/i=6/a=062003

[4] S. Chatrchyan et al. 2008. The CMS Experiment at the CERN LHC. *JINST* 3 (2008), S08004. https://doi.org/10.1088/1748-0221/3/08/S08004

[5] D.H.J. Epema, M. Livny, R. van Dantzig, X. Evers, and J. Pruyne. 1996. A worldwide flock of Condors: Load sharing among workstation clusters. *Future Generation Computer Systems* 12, 1 (1996), 53 – 65. https://doi.org/10.1016/0167-739X(95)00035-Q Resource Management in Distributed Systems.

[6] Burt Holzman, Lothar A. T. Bauerdick, Brian Bockelman, Dave Dykstra, Ian Fisk, Stuart Fuess, Gabriele Garzoglio, Maria Girone, Oliver Gutsche, Dirk Hufnagel, Hyunwoo Kim, Robert Kennedy, Nicolo Magini, David Mason, P Spentzouris, Anthony Tiradani, Steve Timm, and Eric W. Vaandering. 2017. HEPCloud, a New Paradigm for HEP Facilities: CMS Amazon Web Services Investigation. 1 (12 2017).

[7] Gregory M. Kurtzer, Vanessa Sochat, and Michael W. Bauer. 2017. Singularity: Scientific containers for mobility of compute. *PLOS ONE* 12, 5 (05 2017), 1–20. https://doi.org/10.1371/journal.pone.0177459

[8] T. Maeno, K. De, T. Wenaus, P. Nilsson, G. A. Stewart, R. Walker, A. Stradling, J. Caballero, M. Potekhin, and D. Smith. 2011. Overview of ATLAS PanDA workload management. *J. Phys. Conf. Ser.* 331 (2011), 072024. https://doi.org/10.1088/1742-6596/331/7/072024

[9] Massimo Mezzadri, Francesco Prelz, and David Rebatto. 2011. Job submission and control on a generic batch system: the BLAH experience. *Journal of Physics: Conference Series* 331, 6 (2011), 062039. http://stacks.iop.org/1742-6596/331/i=6/a=062039

[10] Nicholas A. Nystrom, Michael J. Levine, Ralph Z. Roskies, and J. Ray Scott. 2015. Bridges: A Uniquely Flexible HPC Resource for New Communities and Data Analytics. In *Proceedings of the 2015 XSEDE Conference: Scientific Advancements Enabled by Enhanced Cyberinfrastructure (XSEDE '15)*. ACM, New York, NY, USA, Article 30, 8 pages. https://doi.org/10.1145/2792745.2792775

[11] Ruth Pordes, Don Petravick, Bill Kramer, Doug Olson, Miron Livny, Alain Roy, Paul Avery, Kent Blackburn, Torre Wenaus, Frank WÃijrthwein, Ian Foster, Rob Gardner, Mike Wilde, Alan Blatecky, John McGee, and Rob Quick. 2007. The open science grid. *Journal of Physics: Conference Series* 78, 1 (2007), 012057. http://stacks.iop.org/1742-6596/78/i=1/a=012057

[12] I. Sfiligoi, D. C. Bradley, B. Holzman, P. Mhashilkar, S. Padhi, and F. Wurthwein. 2009. The Pilot Way to Grid Resources Using glideinWMS. In *2009 WRI World Congress on Computer Science and Information Engineering*, Vol. 2. 428–432. https://doi.org/10.1109/CSIE.2009.950

[13] Igor Sfiligoi, Daniel C. Bradley, Burt Holzman, Parag Mhashilkar, Sanjay Padhi, and Frank Wurthwrin. 2009. The pilot way to Grid resources using glideinWMS. *WRI World Congress* 2 (2009), 428–432. https://doi.org/10.1109/CSIE.2009.950

[14] TACC 2018. Texas Advanced Computing Center Stampede2. (mar 2018). Retrieved March 24, 2018 from https://www.tacc.utexas.edu/systems/stampede2

[15] J. Towns, T. Cockerill, M. Dahan, I. Foster, K. Gaither, A. Grimshaw, V. Hazlewood, S. Lathrop, D. Lifka, G. D. Peterson, R. Roskies, J. R. Scott, and N. Wilkins-Diehr. 2014. XSEDE: Accelerating Scientific Discovery. *Computing in Science Engineering* 16, 5 (Sept 2014), 62–74. https://doi.org/10.1109/MCSE.2014.80

[16] Rick Wagner, Philip Papadopoulos, Dmitry Mishin, Trevor Cooper, Mahidhar Tatineti, Gregor von Laszewski, Fugang Wang, and Geoffrey C. Fox. 2016. User Managed Virtual Clusters in Comet. In *Proceedings of the XSEDE16 Conference on Diversity, Big Data, and Science at Scale (XSEDE16)*. ACM, New York, NY, USA, Article 24, 8 pages. https://doi.org/10.1145/2949550.2949555

[17] Derek Weitzel, Brian Bockelman, Duncan A. Brown, Peter Couvares, Frank Würthwein, and Edgar Fajardo Hernandez. 2017. Data Access for LIGO on the OSG. In *Proceedings of the Practice and Experience in Advanced Research Computing 2017 on Sustainability, Success and Impact (PEARC17)*. ACM, New York, NY, USA, Article 24, 6 pages. https://doi.org/10.1145/3093338.3093363

[18] D Weitzel, I Sfiligoi, B Bockelman, J Frey, F Wuerthwein, D Fraser, and D Swanson. 2014. Accessing opportunistic resources with Bosco. *Journal of Physics: Conference Series* 513, 3 (2014), 032105. http://stacks.iop.org/1742-6596/513/i=3/a=032105

[19] Von Welch, Ian Foster, Carl Kesselman, Olle Mulmo, Laura Pearlman, Steven Tuecke, Jarek Gawor, Sam Meder, and Frank Siebenlist. 2004. X. 509 proxy certificates for dynamic delegation. (01 2004).