The Evolution of Cloud Computing in ATLAS

2015 J. Phys.: Conf. Ser. 664 022038

(http://iopscience.iop.org/1742-6596/664/2/022038)

Journal of Physics: Conference Series **664** (2015) 022038 doi:10.1088/1742-6596/664/2/022038

The Evolution of Cloud Computing in ATLAS

Ryan P Taylor,[1] Frank Berghaus,[1] Franco Brasolin,[2] Cristovao Jose Domingues Cordeiro,[3] Ron Desmarais,[1] Laurence Field,[3] Ian Gable,[1] Domenico Giordano,[3] Alessandro Di Girolamo,[3] John Hover,[4] Matthew LeBlanc,[1] Peter Love,[5] Michael Paterson,[1] Randall Sobie[1] and Alexandr Zaytsev,[4] on behalf of the ATLAS Collaboration

[1] University of Victoria, Victoria, Canada
[2] INFN Bologna, Bologna, Italy
[3] CERN, Geneva, Switzerland
[4] Brookhaven National Laboratory, Upton, USA
[5] Lancaster University, Lancaster, UK

Abstract. The ATLAS experiment at the LHC has successfully incorporated cloud computing technology and cloud resources into its primarily grid-based model of distributed computing. Cloud R&D activities continue to mature and transition into stable production systems, while ongoing evolutionary changes are still needed to adapt and refine the approaches used, in response to changes in prevailing cloud technology. In addition, completely new developments are needed to handle emerging requirements.

This paper describes the overall evolution of cloud computing in ATLAS. The current status of the virtual machine (VM) management systems used for harnessing Infrastructure as a Service resources are discussed. Monitoring and accounting systems tailored for clouds are needed to complete the integration of cloud resources within ATLAS' distributed computing framework. We are developing and deploying new solutions to address the challenge of operation in a geographically distributed multi-cloud scenario, including a system for managing VM images across multiple clouds, a system for dynamic location-based discovery of caching proxy servers, and the usage of a data federation to unify the worldwide grid of storage elements into a single namespace and access point. The usage of the experiment's high level trigger farm for Monte Carlo production, in a specialized cloud environment, is presented. Finally, we evaluate and compare the performance of commercial clouds using several benchmarks.

1. Introduction

In 2011, the ATLAS experiment [1] at the LHC initiated a Cloud Computing R&D project [2], which explored the landscape of cloud computing and virtualization technology available at the time, and investigated how to take advantage of the emerging cloud computing paradigm in the context of ATLAS' pre-existing grid-based computing model (the "grid of clouds" approach). This work continued throughout 2012 and 2013, as additional development on cloud systems, including the ATLAS high level trigger (HLT) farm and commercial clouds, resulted in the growth of cloud resources integrated into ATLAS Distributed Computing (ADC) [3].

During Long Shutdown 1 (LS1), the Cloud Computing Operations & Integration team was formed in late 2013, and the primary focus of cloud activity in the ATLAS experiment began to shift from R&D towards operations, in preparation for LHC Run 2. Whereas the role of the R&D activity is to explore, evaluate and test new cloud computing methods and technologies, the mandate of the Operations & Integration group is to apply available solutions which have gained maturity, in order to streamline and standardize the process of putting clouds into production, and to ensure their smooth operation and integration in ADC.

2. Virtual Machine Image and Contextualization

In 2013 all ATLAS grid resources transitioned from the EL5 operating system[1] to EL6, and cloud resources followed suit. To provide an EL5 environment on the cloud, we had been using the CernVM 2 virtual machine (VM) image [4]. An EL6 environment was available in CernVM 3, which used a novel microkernel-based image called μCernVM [5].[2] However, we encountered an incompatibility between the format of user-data contextualization needed by μCernVM and that needed for use on Nimbus clouds. Several years previously Nimbus [6] had been prevalent in North America, but by 2014 OpenStack[3] had become ascendant and the Nimbus cloud platform was no longer supported. Therefore, we used plain EL6 images as an interim solution, until we phased out the usage of Nimbus by mid-year, allowing us to fully adopt CernVM 3.

To configure VM instances, we initially used Puppet,[4] a powerful and flexible configuration management tool. This choice was made because some commercial clouds did not allow users to upload images. Instead, users had to use a provided image, such as plain EL6, and modify it according to their needs. However, this restriction was later removed, allowing us to use CernVM everywhere, and adopt Cloud-Init[5] for contextualizing VMs. This alternative configuration method, while less powerful, proved to be much simpler to use and maintain.

3. Operation of a Distributed Cloud Computing System

We operate two instances of a distributed cloud computing system: one at the University of Victoria and another at CERN. The system, based on HTCondor [7] for batch job scheduling and Cloud Scheduler [8] for dynamic VM provisioning, harnesses the resources of many Infrastructure as a Service (IaaS) resources (primarily academic OpenStack clouds) across North America, Europe and Australia for use by ATLAS [9]. Fig. 1 shows a high-level view.

HTCondor was designed as a cycle scavenger, making it an ideal job scheduler for a dynamic cloud environment, where VMs appear and disappear on demand. We use the dynamic slot functionality to satisfy the resource requirements of arbitrary job types. For example, multi-core, single-core and high-memory jobs can be handled without any additional configuration.

The distributed, multi-cloud nature of the system presents unique challenges. VMs — especially μCernVM — require a responsive connection to a Squid cache[6] in order to effectively access CVMFS, which provides application software, conditions data, and the CernVM 3 environment [10]. We have deployed Shoal [11] to enable each VM to discover the optimal Squid cache to use. By deploying Squid caches as open reverse proxies with Shoal, a robust and scalable Squid network can be created to enable global cloud usage without the need for topological configuration. We are investigating the use of a data federation [12] to simplify the access to data storage in a similar manner.

When using increasing numbers of clouds, it becomes unsustainably time-consuming and error-prone to interact with the image repository of each cloud individually. To solve the problem

[1] "EL" denotes an enterprise Linux variant such as Red Hat Enterprise Linux, Scientific Linux, or CentOS.
[2] Whereas CernVM 2 is a simple disk image, CernVM 3 is a virtual appliance consisting of the μCernVM image and the CernVM OS repository. The μCernVM image is very small ($\sim$20 MB) and contains only a microkernel and CVMFS [10] client. When a VM is instantiated from this image, it provides the CernVM 3 environment and EL6 operating system via CVMFS.
[3] https://www.openstack.org/
[4] https://puppetlabs.com/
[5] https://launchpad.net/cloud-init
[6] http://www.squid-cache.org/

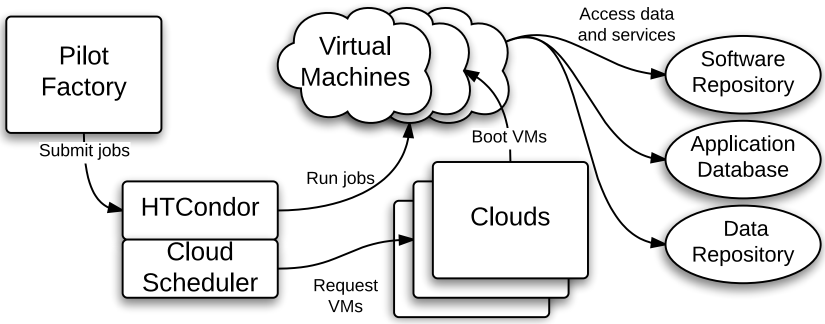


Figure 1. A high-level view of the distributed cloud computing system used by ATLAS since 2012. Jobs are submitted by an ATLAS pilot factory to HTCondor. Cloud Scheduler reviews the requirements of queued jobs, and instantiates VMs on available clouds using a user-specified image to meet those requirements. Once a VM is booted, it joins the HTCondor resource pool and starts to run jobs, accessing experiment software, conditions data and input data. VMs are shut down when no longer needed by queued jobs.

of image management and distribution, we have employed an OpenStack plugin called Glint [13], which allows the user to specify the identity endpoint and credentials for each cloud, easily and quickly transfer images from any cloud to any other cloud, and modify and delete images on many clouds at once.

4. Amazon Pilot Project at Brookhaven Lab RHIC/ATLAS Computing Facility

The goal of this project is to run all ATLAS workloads at large scale on the EC2 spot market. The project was enabled by a $200,000 grant-in-credit provided by Amazon. To use the EC2 spot market economically, several aspects of the ATLAS computing model must be considered, including infrastructure provisioning, data storage, networking, and workload system support. For provisioning, we are using the ATLAS AutoPyFactory pilot system [14] (based on Condor-G), to submit pilot jobs and programmatically manage VM lifecycles for scaling to workload. VMs are authored using imagefactory[7] and a custom templating system. Runtime configuration is handled via Cloud-Init and on-VM Puppet with Hiera[8] running in masterless mode.

To support large-scale operation, dedicated network peering and link management are being put in place. ESNet has established peering with Amazon near all three US EC2 regions, and is doing so in Europe as well. As a result, Amazon agreed to waive data egress fees, provided the transfer charges are less than 15% of the total cost. To fully exploit spot market pricing, ATLAS has accelerated development of the Event Service [15], which allows jobs to perform useful work in very small units. To further minimize data egress volume, ATLAS is working to enable S3 as a native storage element, both for job stage-in and stage-out, and for intermediate results of Event Service jobs.

A scaling test in November 2014 duplicated the scale of the ATLAS Tier 1 at BNL for several days, running simulation jobs on approximately 2,500 8-core nodes, and staging data in and out from the BNL storage element. The test cost $25,000, and the data egress fee ($2,500) was small enough to be waived. The run performed useful work at a cost-effectiveness similar to that of the Tier 1, so it was considered economical. Once the network peering and Event Service functionalities are in place, we will perform scaling tests with 50,000 and 100,000 cores, which,

[7] http://imgfac.org/
[8] http://docs.puppetlabs.com/hiera/

if successful, should lead directly to ongoing commercial use of EC2 by US ATLAS.

5. Simulation at Point 1

The Simulation at Point 1 (Sim@P1) project provides additional computing resources to the ATLAS experiment by opportunistically exploiting the trigger and data acquisition (TDAQ) farm when it is not being used for data taking. This project started at the beginning of LS1 in 2013, and is a result of the efforts of the ATLAS TDAQ administrator teams, CERN IT, the University of Wisconsin-Madison and the RHIC & ATLAS Computing Facility at BNL.

The ATLAS HLT farm contains more than 1,300 compute nodes, which are particularly suitable for running CPU-bound, low I/O workloads such as event generation and Monte Carlo production. The farm can run up to 2,700 VMs with 8 vCPUs each. Sim@P1 uses OpenStack Icehouse to isolate VMs on a separate VLAN, in order to avoid any interference with the ATLAS Technical Network machines and the ATLAS Detector Control System. This approach also enables efficient decoupling of support at both the physical (hardware and virtualization) and logical (grid site support and VM lifecycle) levels of infrastructure.

During LS1, Sim@P1 was one of the most prolific ATLAS sites: it delivered 33 million CPU hours and generated more than 1.1 billion Monte Carlo events between January 2014 and February 2015. In 2014, based on experience gained during LS1, Sim@P1 was enhanced with a fast switching tool to adapt to the conditions of the upcoming LHC Run 2 period. This tool makes the transformation between Sim@P1 mode and TDAQ mode fast, reliable and automated, in order to exploit the HLT farm more efficiently by using it on short notice or for short periods.

6. Helix Nebula

Following the purchasing process of commercial cloud resources within the Helix Nebula marketplace, the computing resources of a European commercial cloud provider were connected to the ATLAS workload management system in order to run Monte Carlo production jobs. Over a period of one month, up to 3,000 single-core virtual machines were deployed concurrently, providing about 963,000 CPU hours of processing and generating 8.8 million events, with a job efficiency of 97%. The resources were benchmarked using the Atlas Kit Validation tool [16], running Monte Carlo simulation of single muon events. Further analysis of CPU performance of production jobs shows that it is consistent with the measured benchmark performance [17].

7. Comparison of a Commercial OpenStack cloud with a WLCG Grid Site

In collaboration with Lancaster University, ATLAS has been given access to a commercial OpenStack instance in the UK. Initially, ATLAS workloads were used to stress-test the OpenStack Havana deployment; this experience helped to refine the hardware requirements for a later upgrade to OpenStack Icehouse, and a subsequent allocation of 220 cores and 500 GB RAM for ATLAS. This resource was then integrated into the ADC infrastructure using Cloud Scheduler.

Metric	Grid Site	Cloud Site
Job success rate (%)	96.3	97.6
Software setup time (s)	15.4 ± 6.7	45 ± 15
Data stage-in time (s)	10.9 ± 3.9	54 ± 21
Total walltime (h)	4.7 ± 0.5	7.1 ± 1.0
CPU efficiency (%)	97.9 ± 0.8	97.4 ± 1.0

Table 1. The means and standard deviations of several HammerCloud metrics comparing a grid site and a cloud site. See `http://cern.ch/go/86fz` for complete results.

The HammerCloud [18] benchmarking tool was used to compare this cloud site with a WLCG site. It launched a continuous 24-hour stream of Monte Carlo simulation jobs, which read a single 100 MB input dataset located at the grid site. Several results are shown in Table 1. The software initialization relies on the CVMFS cache on a node, as well as access to a squid proxy located at the grid site. Data access and software setup are significantly slower on the cloud site because of the remote network access. Also, newly instantiated VMs on the cloud are disadvantaged by starting with an empty CVMFS cache that needs to be filled. However, these effects are negligible compared to the total walltime of the jobs. Most importantly, the grid and cloud site have the same high CPU efficiency.[9] This analysis demonstrates the suitability of simulation workloads for running on clouds, while also identifying areas that would be susceptible to future optimizations, such as deploying squid caches and storage resources in clouds.

8. Benchmarking Commercial and Academic Clouds for HEP Applications

Commercial IaaS cloud providers are now widespread, and offer potential benefits for distributed computing in High Energy Physics (HEP). They could help stabilize the load on experiments' pledged resources in periods of increased demand, such as before major conference deadlines or during large-scale data simulation or reprocessing campaigns. Since ATLAS regularly uses commercial clouds for simulation, it is valuable to compare the performance of purchased IaaS resources with academic clouds, from a HEP perspective. This study evaluates the performance of various VM types using the HEP-SPEC06 (or HS06) benchmark [19], which is based on the SPEC CPU 2006 benchmarking suite [20]. The proportion of integer and floating-point operations executed in HS06 is representative of typical HEP simulation and analysis workloads. This benchmarking method follows previous efforts to evaluate the suitability of virtual machines for HEP applications [21].

Commercially available resources from Google Compute Engine (GCE) and Amazon EC2 were studied, as well as two academic clouds maintained by Compute Canada at the University of Victoria and l'Université de Sherbooke. The CernVM image was used to instantiate VMs. In each VM, a single HS06 job ran using all cores in the VM. For each VM type,[10] 50 benchmark runs were conducted and averaged. The data are plotted in Fig. 2, with a summary of results and VM types shown in Table 2. The benchmark scores on the commercial clouds are quite consistent whereas the academic clouds show more variation; this is under investigation.

9. Monitoring

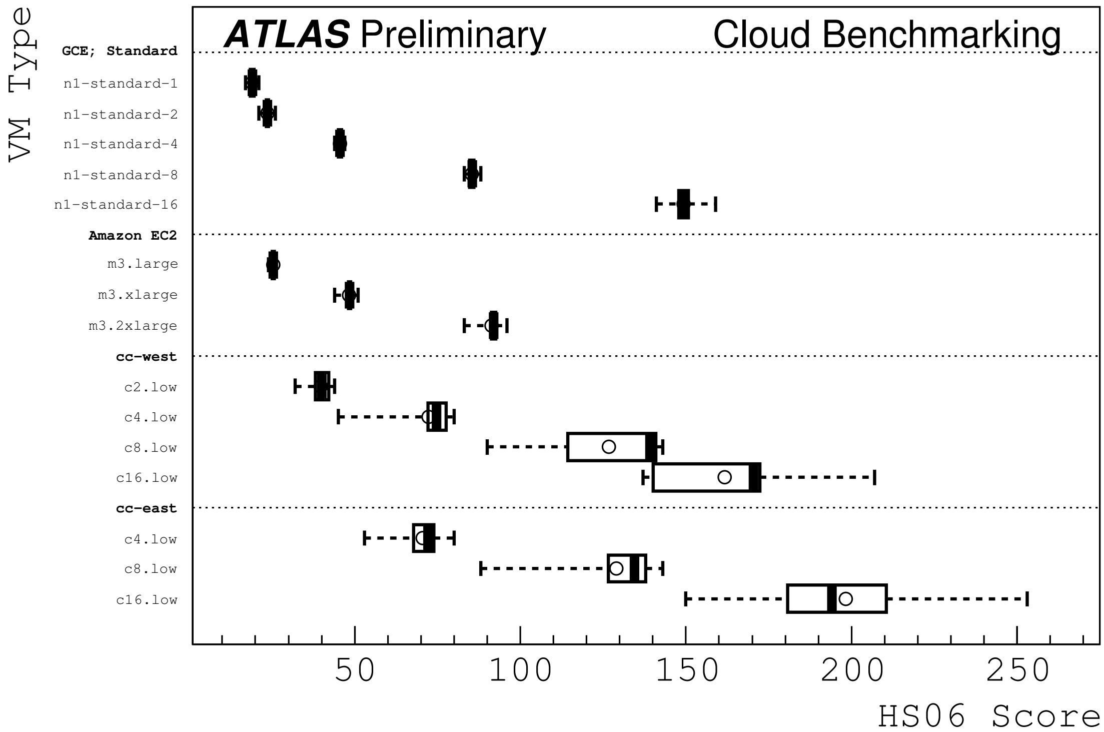
The use of IaaS resources leads toward a more centralized operational model, where resources are managed and monitored by the experiment, rather than by resource sites. This shift requires additional effort from experiment operational teams to investigate and fix issues. Therefore, to lower the operational burden it is essential to have a central monitoring framework in order to easily detect problematic VMs. Moreover, since VM monitoring is a generic task, operational costs can be further lowered by implementing a common monitoring framework shared by multiple experiments.

To address these needs, we have deployed a monitoring service at `http://agm.cern.ch/` based on the open-source distributed monitoring system, Ganglia.[11] Ganglia's scalable design is vital, since there are many VMs reporting from numerous clouds around the world, and IaaS resource deployment must not be limited by a monitoring bottleneck. The scalability of this approach depends on the number of metrics that are collected and the frequency at which they are reported, both of which can be adjusted in the Ganglia configuration. Since this monitoring service reports system information, it complements pre-existing monitoring services that report job information.

[9] Note that the walltime on the cloud site is significantly longer, but the CPU efficiency shows that this is simply due to a less powerful CPU, and not a performance problem.

[10] GCE also has the `highcpu` VM types, which are cheaper and have less memory than the `standard` types, but are otherwise identical. We verified that they have the same HS06 performance, as expected since HS06 does not benefit from additional memory beyond the required amount.

[11] http://ganglia.sourceforge.net/



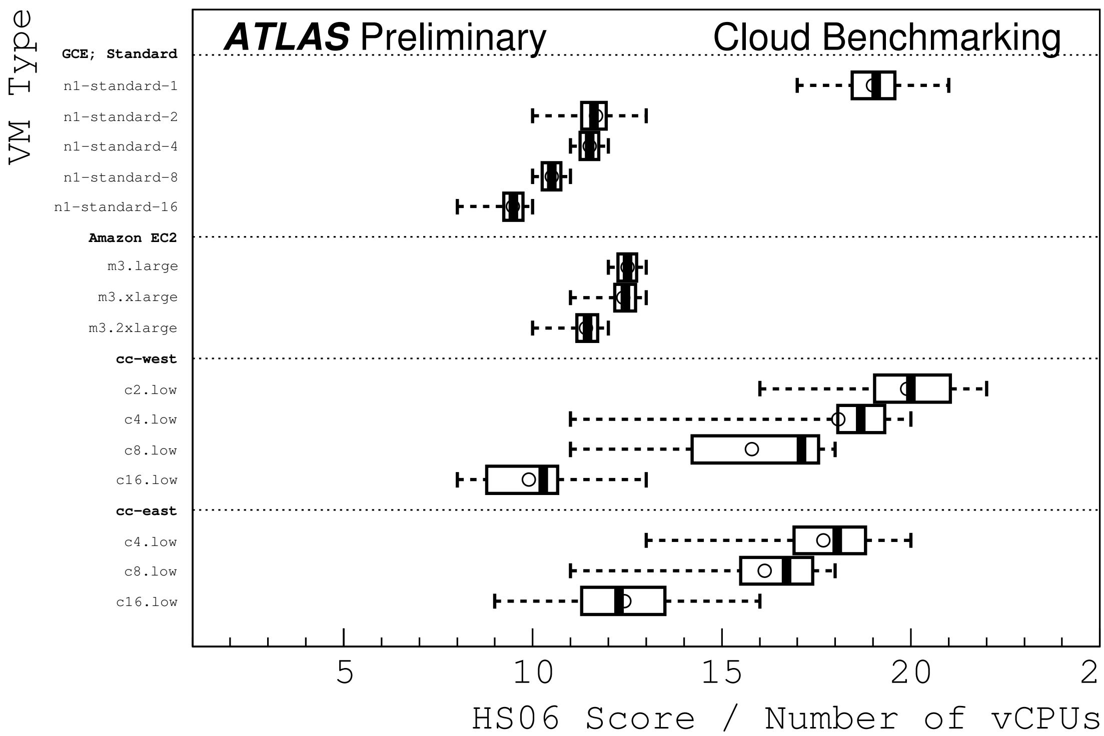


Figure 2. The benchmark scores of different VM types at several commercial and academic clouds. Top: absolute HS06 score. Bottom: HS06 score per vCPU.

Cloud	CPU Model	VM Type	vCPUs	RAM (GB)	HS06 Score
GCE (us-central1-b)	E5-2670	`n1-standard-1`	1	3.75	18.97 ± 0.66
		`n1-standard-2`	2	7.50	23.66 ± 0.62
		`n1-standard-4`	4	15.0	45.46 ± 0.55
		`n1-standard-8`	8	30.0	85.46 ± 0.92
		`n1-standard-16`	16	60.0	149.40 ± 2.95
Amazon EC2	E5-2670, E5-2670v2	`m3.large`	2	7.5	25.20 ± 0.20
		`m3.xlarge`	4	15	48.26 ± 1.12
		`m3.2xlarge`	8	30	91.31 ± 2.46
Victoria	E5-2650v2	`c2.low`	2	7.68	38.64 ± 15.81
		`c4.low`	4	15.36	61.83 ± 5.98
		`c8.low`	8	30.72	119.71 ± 24.39
		`c16.low`	16	61.44	147.24 ± 17.64
Sherbrooke	E5-2650v2	`c4.low`	4	15.36	70.61 ± 5.92
		`c8.low`	8	30.72	129.03 ± 13.78
		`c16.low`	16	61.44	198.24 ± 27.55

Table 2. A summary of the clouds, VM types, HS06 results, and underlying CPU hardware.

10. Accounting

Accounting of computing resource usage is important for both pledged and beyond-pledge sources, in order to recognize and quantify the contributions from member institutions, volunteers, and opportunistic resources, and to prioritize development and support effort according to the value derived from each computing resource. There are two complementary approaches to resource accounting: provider accounting and consumer accounting.

Provider accounting measures resources delivered by an infrastructure provider. The business model of commercial IaaS providers hinges on the ability to track the usage of a customer and generate an invoice for resources provided during the billing period, accompanied by an itemized description of charges. Similarly, WLCG federations report the resources they deliver to each LHC experiment. The reported resource metrics for commercial providers and WLCG federations may be similar, but the mechanisms for collecting the information differ.

Consumer accounting is the measurement of resource usage from the consumer's perspective. It is used to cross-check provider accounting and identify potential inefficiencies or discrepancies, and is particularly important in the case of commercial cloud usage for validating the legitimacy of invoices. However, in the case of cloud resources new functionality is needed to generate resource consumption reports. Another challenge in cloud accounting is obtaining reliable benchmarks of computing power so that resource metrics can be properly quantified.

The Ganglia monitoring system reports metrics that are also of interest for accounting (including the quantity of VMs and vCPUs, and memory and network usage); this can be leveraged to implement consumer-side accounting for clouds. Ganglia reports the value of each metric periodically,[12] so performing a numerical integration of this data over a given period yields the desired accounting records, with an accuracy dependent on the reporting interval. A prototype of this approach is available at `http://cloud-acc-dev.cern.ch/accounting/` and is being used to cross-check the provider accounting from WLCG federations, and to validate the invoices received from commercial providers for the Helix Nebula project.

[12] Version 3.5.12 of the Ganglia web front-end offers the ability to download this information in the form of a JSON file.

11. Summary

This paper summarizes the recent activity of the ATLAS Cloud Computing group. Based on experience gained during Run 1, we have made evolutionary changes during LS1, such as simplification of VM contextualization and enhancement of the Sim@P1 system. Also, several new services have been deployed to facilitate the operation of a distributed cloud computing system. We have characterized the performance and capabilities of commercial clouds using HEP-specific benchmarks and jobs, and conducted large-scale production runs. With the introduction of monitoring and accounting tools suited for clouds, the operational model for harnessing cloud resources has become fully production-ready. These adaptations will prepare the ATLAS experiment to continue successfully leveraging cloud resources to help meet the challenging computing demands of Run 2.

References

[1] The ATLAS Collaboration 2008 The ATLAS Experiment at the CERN Large Hadron Collider *JINST* **3** S08003 (doi:10.1088/1748-0221/3/08/S08003)
[2] Barreiro Megino F H *et al* on behalf of the ATLAS Collaboration 2012 Exploiting Virtualization and Cloud Computing in ATLAS *J. Phys.: Conf. Ser.* **396** 032011 (doi:10.1088/1742-6596/396/3/032011)
[3] Panitkin S *et al* on behalf of the ATLAS Collaboration 2014 ATLAS Cloud R&D *J. Phys.: Conf. Ser.* **513** 062037 (doi:10.1088/1742-6596/513/6/062037)
[4] Buncic P *et al* 2010 CernVM — a virtual software appliance for LHC applications *J. Phys.: Conf. Ser.* **219** 042003 (doi:10.1088/1742-6596/219/4/042003)
[5] Blomer J *et al* 2014 Micro-CernVM: Slashing the Cost of Building and Deploying Virtual Machines *J. Phys.: Conf. Ser.* **513** 032009 (doi:10.1088/1742-6596/513/3/032009)
[6] Keahey K and Freeman T 2008 Contextualization: Providing One-Click Virtual Clusters *Fourth IEEE International Conference on eScience* pp 301-308 (doi:10.1109/eScience.2008.82)
[7] Thain D, Tannenbaum T and Livny M 2005 *Concurr. Comput. Pract. Exper.* **17** p 323
[8] Fransham K *et al* 2010 Research computing in a distributed cloud environment *J. Phys.: Conf. Ser.* **256** 012003 (doi:10.1088/1742-6596/256/1/012003)
[9] Gable I *et al* HEP cloud production using the Cloud Scheduler/HTCondor Architecture, proceedings of the CHEP 2015 conference *J. Phys.: Conf. Ser.*
[10] Blomer J, Buncic P and Fuhrmann T 2011 CernVM-FS: delivering scientific software to globally distributed computing resources *First international ACM workshop on Network-aware Data Management (NDM 11)* p 49 (doi:10.1145/2110217.2110225)
[11] Gable I *et al* 2014 Dynamic web cache publishing for IaaS clouds using Shoal *J. Phys.: Conf. Ser.* **513** 032035 (doi:10.1088/1742-6596/513/3/032035)
[12] Furano F *et al* 2012 Dynamic federations: storage aggregation using open tools and protocols *J. Phys.: Conf. Ser.* **396** 032042 (doi:10.1088/1742-6596/396/3/032042)
[13] Berghaus F *et al* 2015 Glint: VM image distribution in a multi-cloud environment, proceedings of the CHEP 2015 conference *J. Phys.: Conf. Ser.*
[14] Caballero J *et al* on behalf of the ATLAS Collaboration 2012 AutoPyFactory: A Scalable Flexible Pilot Factory Implementation *J. Phys.: Conf. Ser.* **396** 032016 (doi:10.1088/1742-6596/396/3/032016)
[15] Wenaus T *et al* 2015 The ATLAS Event Service: A New Approach to Event Processing, proceedings of the CHEP 2015 conference *J. Phys.: Conf. Ser.*
[16] De Salvo A and Brasolin F 2010 Benchmarking the ATLAS software through the Kit Validation engine *J. Phys.: Conf. Ser.* **219** 042037 (doi:10.1088/1742-6596/219/4/042037)
[17] Giordano D *et al* 2015 Accessing commercial cloud resources within the European Helix Nebula cloud marketplace, proceedings of the CHEP 2015 conference *J. Phys.: Conf. Ser.*
[18] van der Ster D C, Elmsheuser J, García M, Paladin M 2011 HammerCloud: A Stress Testing System for Distributed Analysis *J. Phys.: Conf. Ser.* **331** 072036 (doi:10.1088/1742-6596/331/7/072036)
[19] Michelotto M *et al* 2010 A comparison of HEP code with SPEC benchmarks on multi-core worker nodes *J. Phys.: Conf. Ser.* **219** 052009 (doi:10.1088/1742-6596/219/5/052009)
[20] Spradling D 2007 SPEC CPU2006 Benchmark Tools *SIGARCH Comput. Archit. News* **35** pp 130-134 (doi:10.1145/1241601.1241625)
[21] Alef M and Gable I 2010 HEP specific benchmarks of virtual machines on multi-core CPU architectures *J. Phys.: Conf. Ser.* **219** 052015 (doi:10.1088/1742-6596/219/5/052015)