Sim@P1: Using Cloudscheduler for offline processing on the ATLAS HLT farm

Frank Berghaus[1,*], *Franco* Brasolin[3], *Kevin* Casteels[1], *Colson* Driemel[1], *Marcus* Ebert[1], *Colin* Leavatt-Brown[1], *Chris* Lee[2], *Peter* Love[5], *Michael* Paterson[1], *Diana* Scannicchio[4], *Rolf* Seuster[1], *Randall* Sobie[1], and *Tahya* Weiss-Gibbons[1]on behalf of the ATLAS Collaboration

[1]University of Victoria, Canada
[2]University of Cape Town, South Africa
[3]Universita e INFN, Bologna, Italy
[4]University of California, Irvine, USA
[5]Lancaster University, UK

Abstract. The Simulation at Point1 (Sim@P1) project was established in 2013 to take advantage of the Trigger and Data Acquisition High Level Trigger (HLT) farm of the ATLAS experiment at the LHC. The HLT farm is a significant compute resource, which is critical to ATLAS during data taking. This large compute resource is used to generate and process simulation data for the experiment when ATLAS is not recording data. The Sim@P1 system uses virtual machines, deployed by OpenStack, in order to isolate the resources from the ATLAS technical and control network. During the upcoming long shutdown in 2019 (LS2), the HLT farm including the Sim@P1 infrastructure will be upgraded. A previous paper on the project emphasized the need for "simple, reliable, and efficient tools" to quickly switch between data acquisition operation and offline processing.In this contribution we assess various options for updating and simplifying the provisional tools. Cloudscheduler is a tool for provisioning cloud resources for batch computing that has been managing cloud resources in HEP offline computing since 2012. We present the argument for choosing Cloudscheduler, and describe technical details regarding optimal utilization of the Sim@P1 resources.

1 Introduction

ATLAS [1] employs a large computer farm to facilitate data acquisition and event selection. The Sim@P1 project aims to opportunistically use the trigger and data acquisition high level trigger resources for offline computing. The High Level Trigger (HLT) is a mission critical part of ATLAS data taking and is physically connected to the control network of the detector and the "data" network which allows connections to the CERN data centre through a switch at P1. When working with Sim@P1 it is important to ensure the secure isolation from the physical resources at P1, seamless integration into the ATLAS distributed computing system, and reliable transition between the functions of the resources. A system satisfying these

*e-mail: berghaus@cern.ch

Table 1. The hardware at P1 currently available for use with Sim@P1. The C6100 nodes are the decommissioned old HLT. They provide 11008 hyper-threaded (HT) cores permanently running in Sim@P1 mode. Not all cores are used to ensure the virtual machines provide sufficient memory for ATLAS offline workloads. The other hardware is switched to Sim@P1 mode when data taking is not foreseen in the next 24 hours. These opportunistic resources provide up to 97216 additional cores. Usually the trigger and data acquisition team retains some resources for their needs.

Product name	Intel® Xeon®	HT Cores	Memory [GB]	VM Cores	Nodes
C6100	X5650	24	24	16	688
Centerprise	E5-2650 v4	48	64	48	360
Persy	E5-2660 v4	56	64	56	440
MegWare	E5-2680 v3	48	64	48	680
QuantaPlex T41S-2U	E5-2680 v3	48	64	48	472

criteria was developed during the first long shutdown of the LHC [2]. Isolation is achieved by running virtual machines on the physical HLT hardware. The virtual machines are managed using the cloud framework OpenStack [3]. The virtual machines share the "data" connection of the HLT hardware through a tagged VLAN providing network isolation on the level of the Ethernet frame managed by the switches. This VLAN allows the Virtual Machines to connect to a controlled list of interfaces in the CERN general purpose network. This list specifies the interface of the machines needed to allow offline workloads to be delivered and executed. To minimize impact of Sim@P1 on the Trigger and Data Acquisition (TDAQ) operation, only simulation tasks from the central production system are submitted to run at P1.

The original implementation of Sim@P1 ran successfully during the first long shutdown of the LHC facilities between 2013 and 2015. Once the experiment resumed data-taking, the system was used in opportunistic mode [4]. The HLT was switched from TDAQ function to Sim@P1 mode for intervals of a few days during technical stops and machine development. To allow this opportunistic usage a set of scripts were developed to manage the transition of resources between TDAQ and Sim@P1 mode.

During the second long shutdown of the LHC facilities, starting in 2019, the HLT will be upgraded. Along with the upgrades to the HLT we plan to upgrade the Sim@P1 infrastructure. We plan to replace the controller nodes, upgrade to a new version of OpenStack, and replace the scripts managing the virtual machines with Cloudscheduler [5].

2 The Sim@P1 infrastructure

During the year end technical stop from December 2017 to March 2018, the computing hardware of the HLT was replaced with new nodes. The old HLT hardware was retained at P1 and operates permanently in Sim@P1 mode. The new hardware has been used in Sim@P1 mode during the various technical and machine development stops throughout 2018. The current hardware configuration is summarized in table 1. With the replacement of the HLT nodes the Sim@P1 system has become significantly larger.

The virtual machines are contextualized using amiconfig[1] [6]. The contextualization sets up the computing environment for the ATLAS offline workloads and sets the virtual machines to advertise themselves to a HTCondor [7] system running in the CERN general purpose network. The virtual machines are configured to use condor's dynamic partitioning feature to map workloads to the resources. Process isolation is achieved using cgroups. The HTCondor system for Sim@P1 was rebuilt with a single central manager and four schedulers. The

[1]Originally a project by rPath, Inc. now maintained by the CernVM team.

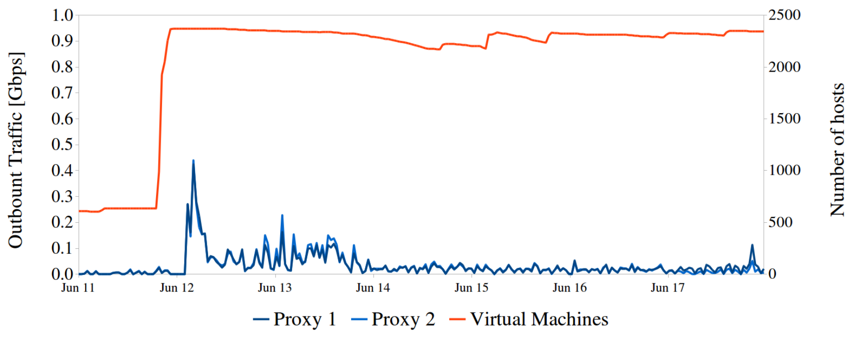

Figure 1. The content of the CernVM file system and the ATLAS repositories is loaded into the virtual machines on demand. This figure shows the number of virtual machines running in Sim@P1 mode in red. The network load on the two proxy caches distributing the file system to the virtual machines is plotted in blue throughout the same time range. There was a delay introduced between launching and configuring the virtual machines in the morning of June 12, and distributing ATLAS workload around noon on June 12.

system is managed by CERN's configuration management system, allowing easy addition of more schedulers as needed. Work is submitted to the schedulers using the PanDA Harvester [8]. With Harvester Sim@P1 presents a single unified queue to the PanDA system. Jobs request the resources they need leveraging the dynamic partitioning of workers. Furthermore the Sim@P1 system now directly notifies the PanDA workload management system when resources are added or removed from Sim@P1. This means PanDA will broker workloads to Sim@P1 immediately when resources are added. We encountered scaling issues with the new software elements introduced to the configuration each time the full farm was brought up throughout the year. Each of these issues could be solved within a day of the resource handover. This prevented a reliable measurement of the time from handover to full utilization of the resources for comparison with previous results.

The virtual machines which are the HTCondor workers were upgraded to CernVM4 [9]. CernVM is a good solution for Sim@P1 because the micro-image can be distributed to all the HLT nodes from a few serves providing the OpenStack image management service, glance. To reduce stress on the glance and CVMFS infrastructure whole-node virtual machines are used. Caching in the software as needed from the CernVM file system distributes the load over time, even when the entire farms is switched from TDAQ mode to Sim@P1 as illustrated in figure 1. The figure shows that the highest stress on the cvmfs infrastructure for Sim@P1 is due to caching in the files for the ATLAS workloads. Two squid caches provide sufficient bandwidth to serve the file system to Sim@P1 with room for fail over.

3 Operational experience

The motivation for the upgrades described in section 2 is to make Sim@P1 more flexible and responsive. These were tested by setting the system to run event generation and reconstruction tasks and observe how the system reacted. The tests were performed in coordination with the TDAQ System Administrators to avoid negative effects on data taking. We summarize our experiences here.

3.1 Simulation

When the opportunistic resources of the HLT are switched to Sim@P1 mode, the farm is one of the largest resources for ATLAS offline computing. Like most opportunistic resources utilized by ATLAS, Sim@P1 runs only simulation. While operating the entire HLT in Sim@P1 mode we repeatedly drained the queue of simulation tasks. Then the resource was idle until more tasks could be defined.

Simulation tasks require generated events as input. These inputs are moved to CERN by Rucio for the tasks running at Sim@P1. Transfers scheduled for Sim@P1 are at a lower priority than those scheduled for the standard grid resources at CERN. The result is that Sim@P1 did not receive sufficient input data when any problems occurred with the CERN storage infrastructure or the file transfer system.

Considering the limitations of running only simulation it was attempted to run other workloads on Sim@P1.

3.2 Reconstruction and derivation

Reconstruction and derivation tasks require large files as input. Generally no such tasks are submitted to Sim@P1, because the resulting network load would likely interfere with the transition from Sim@P1 mode back to TDAQ mode. That means network traffic between the Sim@P1 virtual machines and the CERN general purpose network (GPN) should be kept at a minimum to avoid interfering with data taking. During the 2018 winter shutdown we had sufficient flexibility to attempt running tasks which would stress the network connection. A data intensive task was assigned to Sim@P1 for this purpose. The switches at the top of each rack provide a 1GB line shared between all Sim@P1 workers in that rack to connect to the data network. Most transfers timed out as many jobs were downloading input at the same time. It should be noted that the eight 10GB lines connecting P1 to the GPN saw only a few Gbps of network traffic as a result.

3.3 Event generation

Event generation tasks use programs, such as Pythia or Madgraph, to calculate the interactions of particles at high energies. The output of these tasks forms the input of simulations tasks. The computing requirements of the event generation programs are very variable, which makes estimating the requirements of the jobs in the tasks difficult.

PanDA calculates the requirements of jobs within a certain task by executing a sample of "scout jobs" measuring their resources consumption to determine their memory and disk requirements. The brokerage of the jobs to sites thereafter is determined by which sites can provide those requirements. In the case of Sim@P1 Harvester communicates the job memory requirements to condor using the `request_memory` ClassAd. In turn condor reserves that amount of memory for the job. Many event generation jobs significantly exceed the amount of memory requested. This leads to the virtual machines becoming unresponsive at which point the jobs fail[2]. Manual intervention is required to terminate and restart the virtual machine. This lack of error correction in the current Sim@P1 system in one motivation for the use of Cloudscheduler proposed in section 4.

Event generation tasks run on a single core only. This means that the condor system must track many more jobs. Four condor submission nodes allow a reasonably large number of jobs to run on Sim@P1. Each such node provided 16 GB of memory and proved to be able to manage about 5000 jobs.

[2]Reporting a lost heartbeat as no logs may be retrieved from the unresponsive system.

By iterative increases of the share of event generation tasks allowed in Sim@P1 we found that the farm can maintain around 10% event generation tasks before some virtual machine workers become unresponsive.

4 Plans

During the upcoming second long shutdown upgrades to the TDAQ system are planned. The Sim@P1 infrastructure will be upgraded simultaneously. This section presents four possible upgrade scenarios that were investigated. Arguments are given why some options where rejected. The most likely upgrade plan is discussed in section 4.1. The container based option discussed in section 4.4 may be of interest depending on the evolution of the HLT infrastructure for the high-luminosity LHC.

4.1 Cloudscheduler

To reduce the amount of manual intervention needed for error correction we propose to use Cloudscheduler to drive the OpenStack resources. Cloudscheduler has already been operating an opportunistic system of distributed cloud resources in the ATLAS and Belle-II distributed computing systems. It has a proven track record of managing cloud resources for batch workload execution [10, 11]. As illustrated in figure 2 Cloudscheduler needs to communicate with both the cloud interfaces and the batch system. In the case of Sim@P1 the cloud interface is on a private network that strongly restricts the network interfaces allowed to connect for security reasons.

We realized that Cloudscheduler may be separated into two components: the processes that track the state of the HTCondor system and the processes that manages life cycle of the virtual machine instances. An arbitrary number of threads collects information from HTCondor and publishes it into the database[3]. The part of Cloudscheduler controlling the cloud interface uses this information to make scheduling decisions and keeps its own state information in the same database. This means we need a database that both systems may communicate with. Figure 3 illustrates how such a system could be built while minimizing the network exposure of the point one infrastructure. We propose to run the Cloudscheduler in the P1 network, directly managing the virtual machine life-cycle. The batch system polling thread would run in the TDAQ testbed, feeding a database in the testbed. The Cloudscheduler may connect to the database through the single port link already in use for monitoring the current Sim@P1 system. Alternatively, the database could be placed inside the P1 network, with the polling threads communicating though the same single port link.

4.2 Vacuum

The Vac software [12] developed at GirdPP offers an elegant solution. Instead of managing the virtual machines running on the HLT nodes with a central OpenStack controller they would be managed by a Vac daemon running on each HLT node. When a rack is switched to Sim@P1 mode the Vac daemon would be started on the node. The Vac daemon would instanciate virtual machines contextualized in two possible ways: to launch a pilot job to connect to PanDA to retrieve a workload, or to connect to HTCondor running in the general purpose network and retrieve pilot jobs from the condor system.

Removing the cloud and batch system would ease the operational burden. Unfortunately, the sole developer of Vacuum is busy with many other responsibilities. ATLAS lacks the manpower to take over the project should this be necessary.

[3]We use MariaDB

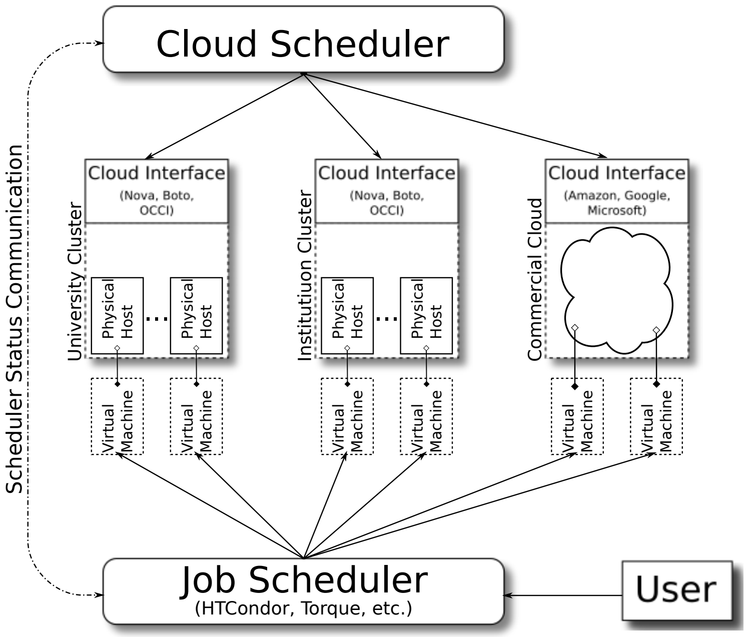

Figure 2. Cloudscheduler monitors the status of a batch system, such as HTCondor. When jobs are submitted resources are requested on a cloud interface to execute the workload. The virtual machines are contextualized to provide the needed software and connect to the batch queue. When no jobs require the virtual machine instances they are delete. Cloudscheduler does some automatic error correction of virtual machine instances in the cloud systems.

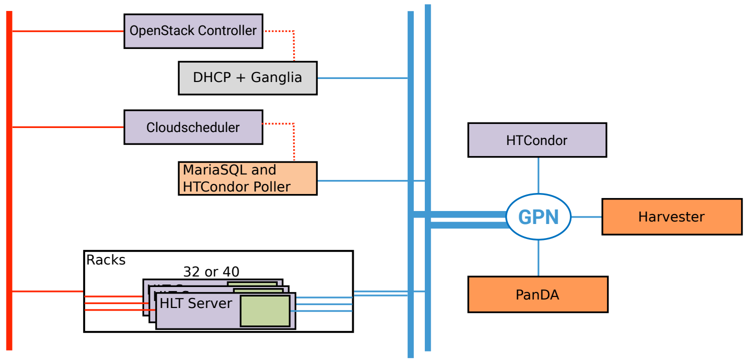

Figure 3. The data network which the virtual machines instances use to communicate with the ATLAS distributed computing services is in blue. The control network which connects the TDAQ HLT hardware to the ATLAS detector is in red. A connection in the TDAQ testbed allowing information exchange on a single port is in the dotted red line.

4.3 Virsh

The Sim@P1 system could be simplified even more by simply using virsh to boot and contextualize the virtual machines. Instead of enabling the OpenStack of Vac daemons on the node one could use create or destroy a virtual machine using libvirt [13]. This lightweight solution reduces the central points of failure which are the controller nodes. Managing virtual machines is more difficult however, since it then becomes necessary to manually intervene on the HLT nodes themselves.

4.4 Containers

The overheads encountered by the Vacuum solution described in section 4.2 could be mitigated by using containers instead of virtual machines. Current solutions allowing ATLAS workloads to be executed in containers require the host to provide a cvmfs file system to be mounted in the container. The HLT nodes cannot provide cvmfs for the containers. Furthermore, the TDAQ System Administrators expressed a strong preference for virtual machines, trusting them to provide better isolation between the offline environment and the physical host.

5 Summary

The HLT of the ATLAS experiment at the LHC was operated as an opportunistic resource for offline computing during the LHC run 2. Sim@P1 was scaled up to operate the new HLT hardware. The behaviour of Sim@P1 was studied to determine whether the system could be used more broadly. It was found that the system can support 10% event generation in addition to simulation. More study is needed to safely include reconstruction and derivation tasks. A plan to use Cloudscheduler to operate Sim@P1 is proposed to automate error correction.

References

[1] The ATLAS Collaboration, JINST **3** S08003 (2008)

[2] S. Ballestrero *et al.*, J. Phys. Conf. Ser. **664** no.2, 022008 (2015)

[3] Openstack project, ”OpenStack” [software], version Icehouse, available from https://www.openstack.org/software/icehouse/ [accessed 2018-09-24]

[4] S. Ballestrero *et al.*, J. Phys. Conf. Ser. **898**, no. 8, 082012 (2017)

[5] R. J. Sobie *et al.*, arXiv:1302.1939 [cs.DC].

[6] CernVM project, “amiconfig” [software], version cernvm-4, available from https://github.com/cernvm/amiconfig [accessed 2018-10-10]

[7] HTCondor project, “HTCondor” [software], version 8.6.11, available from http://htcondor.org [accessed 2018-09-24]

[8] PanDA project, “Harvester” [software], version 0.0.25, available from https://github.com/PanDAWMS/panda-harvester [accessed 2018-09-24]

[9] CernVM team, “CernVM” [software], version 4.1, available from http://cernvm.cern.ch [accessed 2018-09-24]

[10] Taylor R P *et al* J. Phys.: Conf. Ser. **664** 022038 (2015)

[11] Taylor R P *et al* J. Phys.: Conf. Ser. **898** 052008 (2017)

[12] A McNab *et al* J. Phys.: Conf. Ser. **513** 032065 (2014)

[13] libvirt project, “virsh” [software], version 0.10.2, available from https://libvirt.org [accessed 2018-10-10]