EPJ Web of Conferences **214**, 03011 (2019) https://doi.org/10.1051/epjconf/201921403011
CHEP 2018

Advances in ATLAS@Home towards a major ATLAS computing resource

David Cameron[1,*], *Wenjing* Wu[2], *Alexander* Bogdanchikov[3], and *Riccardo* Bianchi[4] on behalf of the ATLAS Collaboration

[1]University of Oslo, P.b. 1048 Blindern, 0316 Oslo, Norway
[2]IHEP, 19B Yuquan Road, Beijing, China 100049
[3]Budker Institute of Nuclear Physics, SB RAS, Novosibirsk 630090, Russia
[4]University of Pittsburgh, Department of Physics and Astronomy, 100 Allen Hall, 3941 O'Hara St, Pittsburgh PA 15260, USA

Abstract. The volunteer computing project ATLAS@Home has been providing a stable computing resource for the ATLAS experiment since 2013. It has recently undergone some significant developments and as a result has become one of the largest resources contributing to ATLAS computing, by expanding its scope beyond traditional volunteers and into exploitation of idle computing power in ATLAS data centres. Removing the need for virtualization on Linux and instead using container technology has made the entry barrier significantly lower for data centre participation and in this paper, we describe the implementation and results of this change. We also present other recent changes and improvements in the project. In early 2017 the ATLAS@Home project was merged into a combined LHC@Home platform, providing a unified gateway to all CERN-related volunteer computing projects. The ATLAS Event Service shifts data processing from file-level to event-level and we describe how ATLAS@Home was incorporated into this new paradigm.

1 Introduction

The ATLAS@Home [1] project began in 2013 with the aim of encouraging members of the public to contribute spare computing cycles to run Monte Carlo simulation processes for the ATLAS experiment [2] at CERN's Large Hadron Collider. Like many volunteer computing projects it uses the BOINC [3] software to manage the distribution of tasks to volunteers. A BOINC server hosts tasks or *work units* to be processed and volunteers run a client which is configured to pull and run work units from specified projects. Once a work unit is processed the client sends the result back to the server which validates the result, and if the result is good awards credit to the volunteer. Credit is simply a measure of how much computation has been done and has no monetary value, however it provides motivation for many volunteers.

The size of ATLAS@Home has been growing constantly, helped by new features such as multi-core tasks [4] and a friendly graphical interface. In this paper we describe recent additions to the project to expand it even further. The consolidation of all LHC-related projects to LHC@Home is presented in Sect. 2. Sect. 3 describes how the move from virtualisation

---

[*] e-mail: david.cameron@cern.ch

to containers significantly decreased the entry barrier to running ATLAS@Home on Linux hosts in data centres and Sect. 4 shows results of running in these centres. In Sect. 5 work on integrating ATLAS@Home with the ATLAS Event Service is presented and conclusions are drawn in Sect. 6.

2 LHC@Home Consolidation

Volunteer computing has a long history at CERN, beginning in 2004 with the orignal LHC@Home project Sixtrack [5], which allowed volunteers to run LHC beam simulations. In the proceeding years the theoretical physics community along with most of the LHC experiments also started their own volunteer computing activity using BOINC. In early 2017 all the CERN-related volunteer computing projects were consolidated into a unified LHC@Home platform [6]. This provides volunteers a single entry point (i.e. a single BOINC server) with the ability to choose which experiments' tasks to run with a single click. It also gives a unified credit system and message boards for communication.

The unification of activities under LHC@Home provided ATLAS@Home with a much larger volunteer base, as several hundred thousands of people had signed up to Sixtrack compared to a few thousand in ATLAS@Home. The Sixtrack project however is more accessible to the average volunteer because it runs natively in Windows, Mac and Linux operating systems. All the other LHC@Home projects require the use of virtualisation to emulate a Linux environment on those operating systems, i.e. installing virtualisation software and having the appropriate hardware requirements correctly configured. This significantly raises the entry barrier to participation from the general public. Despite this, ATLAS@Home benefited from an increase in exposure to a wider community and many volunteers signed up to run ATLAS tasks, either directly or as a failover in case tasks for another project were not available.

Figure 1 shows the CPU consumption in CPU-days for each LHC@Home project in May 2018. ATLAS and Sixtrack are by far the most popular projects but ATLAS always has a steady supply of tasks to run whereas Sixtrack tasks come in bursts. It can be seen that when Sixtrack is low in tasks, ATLAS slightly increases, due to volunteers configuring Sixtrack as their main project but ATLAS as a backup in case no Sixtrack tasks are available.

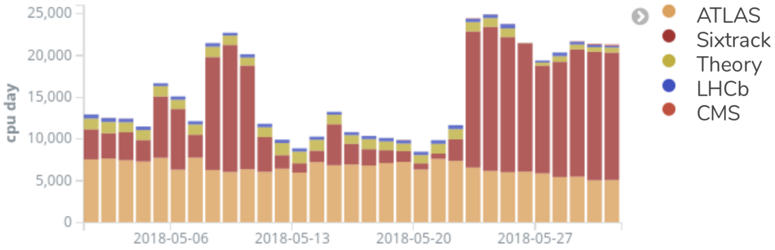

Figure 1. CPU consumption in CPU days per app on LHC@Home in May 2018

3 From Virtualisation to Containers

As mentioned above, ATLAS@Home was built upon virtualisation because ATLAS simulation software requires a specific Linux environment and most volunteers run non-Linux operating systems. It is still true today that the majority of the general public run Microsoft

Windows so virtualisation is still very important for many volunteer computing projects. However, while exploring new avenues for ATLAS@Home it was found that there was a large untapped resource of Linux hosts which potentially could be exploited.

Many data centres have Linux hosts which do not use their CPU to full capacity, for example worker nodes on a batch farm which have no jobs to run, newly purchased machines awaiting commissioning for their intended purpose or service hosts which are over-provisioned and lightly used. Many administrators of these hosts would be reluctant to install virtualisation software simply for the purpose of running ATLAS@Home, and if the data centre already used a virtualised infrastructure it is not easily possible to run a virtual environment inside another virtual environment. In addition there is significant CPU overhead involved in virtualisation.

These issues led to the development of a native Linux version of ATLAS@Home based around container technology, specifically Singularity [7]. Singularity, like other container implementations, provides operating-system level virtualisation, allowing isolated user-space instances which look like a full operating system from inside but are contained in what they can do from the outside. However, because the container shares the host operating system kernel as well as libraries and binaries, there is no need to reproduce an independent copy of the operating system in the container and hence they are much more light-weight than virtualisation both in terms of start-up time and running overhead. A container starts in a fraction of a second compared to minutes for a virtual machine and has almost zero overhead over native performance. Another reason for choosing Singularity is that it has become very popular in the HEP community, meaning that many grid site worker nodes already have Singularity installed or could potentially install it if requested by the community.

The software required to run ATLAS simulations is hosted on the CERN Virtual Machine File System (CVMFS) [8], a read-only file system with multiple caching layers based on the HTTP protocol. In the virtualised ATLAS@Home, CVMFS is mounted inside the virtual machine and the local CVMFS cache is pre-populated with the particular software version required for ATLAS@Home so that each time the virtual machine is started it does not have to download the software over the network. However there are certain other data required for the simulation tasks that varies per task and must be downloaded and cached each time. The native Linux version of ATLAS@Home requires that CVMFS is mounted on the host and configured to use ATLAS repositories. The tools which bootstrap ATLAS@Home tasks then take care of bind-mounting CVMFS inside the container so that it is fully available to the task. An added advantage of this approach is that a single CVMFS cache can be shared among multiple tasks running on the same host, reducing disk space and network traffic.

The native Linux version was released in August 2017 as a test application on LHC@Home, meaning that volunteers had to explicitly opt-in to running it. This was mainly to avoid sending native tasks to Linux hosts that did not have the required extra software installed (CVMFS and Singularity). This software is easy to install and is available in the standard repositories of major Linux distributions, however there are some small configuration steps to follow which require some basic Linux knowledge.

Figure 2 shows the CPU consumption of ATLAS@Home in CPU-days per day for each app version in May 2018. It can be clearly seen that the Linux native version dominates with up to 85% of the CPU consumption. Windows accounts for around 12% and virtualised Linux and Apple Mac making up the remaining 3%. One reason for the success of the native Linux app is that volunteers running it tend to only run ATLAS@Home on their hosts, as opposed to Windows users who tend to run many different volunteer projects. Another reason is the expansion of ATLAS@Home into data centres as will be described in the next section.

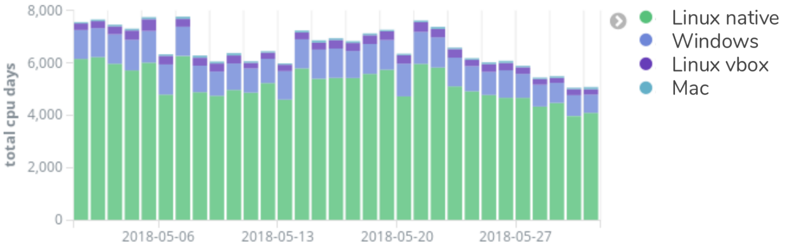

Figure 2. CPU day consumption per version of the ATLAS app during May 2018

4 Running ATLAS@Home in Data Centres

The introduction of the native Linux ATLAS@Home app opened up the possibility of running on resources that the previous requirement of virtualisation made difficult. One such resource is grid worker nodes which for many reasons cannot be used to 100% of their CPU capacity [9]. ATLAS@Home provides a useful way of "backfilling" these nodes, i.e. using the CPU cycles that cannot be used by grid jobs, by running jobs in parallel to the grid jobs on the same machines. Because BOINC is designed to be non-intrusive it only uses CPU that is not being used by other processes, and it can suspend tasks if other processes are using 100% of the CPU. This means that the parallel ATLAS@Home jobs do not disrupt the regular grid jobs. In addition they are not dependent on any of the site's grid services, so if for example the site's grid storage is in downtime and grid jobs cannot run, ATLAS@Home jobs can take over and keep the CPUs busy.

The native app makes setting this up easy, because all ATLAS grid sites already have CVMFS available for running the ATLAS grid jobs, and many sites already use Singularity. Thus the only extra steps for the site administrator are to create an account in LHC@Home, install BOINC client on the nodes and configure it with the account details. Then the BOINC client will automatically download ATLAS@Home tasks and run them. Some additional configuration is also possible for example to set the number of cores used by each task.

Another major contribution to ATLAS@Home from the native app is machines in data centres that are running services which use little CPU, machines procured but not yet commissioned for their final purpose or otherwise idle machines. The CERN data centre in particular has a steady flow of hardware coming in and going out, and running ATLAS@Home on new machines being commissioned or old machines being decommissioned is an effective way of using otherwise wasted CPU cycles.

As shown in Figure 3, the CERN data centre machines (username Agile_Boincers) were by far the largest contributor to ATLAS@Home over the period March to September 2018, with around two-thirds of all CPU consumption. The next three largest contributors were from WLCG [10] sites running ATLAS@Home in backfilling mode: Triumf (a Tier-1 centre in Canada), Beijing (a Tier-2 centre in China) and LRZ-LMU (a Tier-2 centre in Germany). Combined, the data centre machines make up around 85% of all ATLAS@Home resources with the remaining 15% coming from traditional volunteers.

The impact of the native app and its use in data centres can be clearly seen in Figure 4 which shows the number of events simulated by ATLAS@Home per week from January 2017 to September 2018. There was a steady increase starting after the native app was introduced in August 2017 up to a peak in June 2018 which is more than an order of magnitude higher than one year earlier. Some technical issues with the BOINC server deployment for LHC@Home in July/August forced a reduction in the number of tasks running in ATLAS@Home but

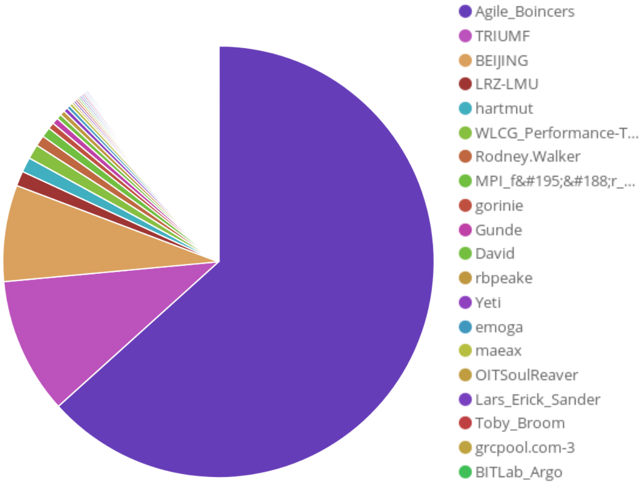

Figure 3. CPU consumption per volunteer from 1 March to 1 September 2018

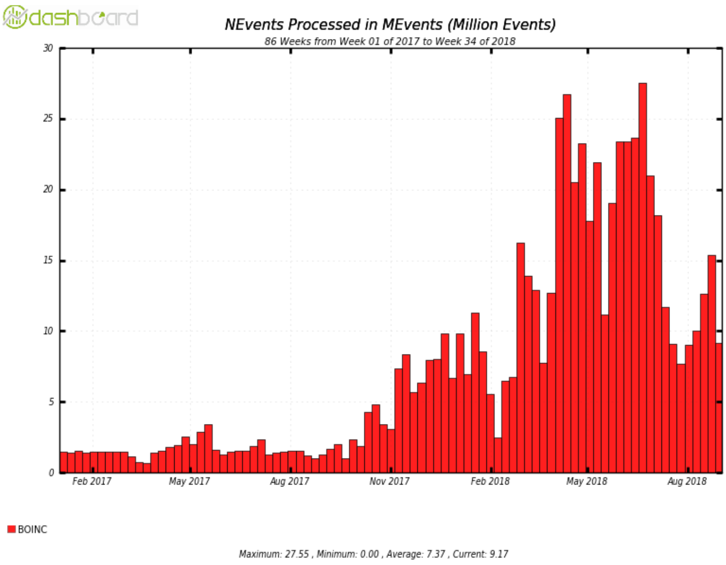

Figure 4. Simulation events processed per week from January 2017 to September 2018

those have since been resolved and the number of events is again increasing. On average ATLAS@Home is processing 4.5% of all ATLAS simulation events, making it equivalent to some of the largest grid sites that ATLAS uses.

5 Integration of ATLAS Event Service

The ATLAS Event Service (AES) [11] represents a paradigm shift from job-level to event-level data processing. A regular Monte Carlo simulation job consists of processing a fixed number of events (for example 200 events on ATLAS@Home or 1000 on grid sites). If the job fails midway during processing then all events are lost and must be reprocessed, which makes it hard to effectively exploit opportunistic resources where jobs may be terminated without warning. AES answers this problem by reducing the level of granularity from jobs to events. As an AES job runs it periodically uploads events that it has processed to grid storage so that in the event of the job ending prematurely, only the events currently being processed are lost. The events not processed by the job are automatically reassigned to another job.

This approach is interesting for ATLAS@Home because often jobs are started and suspended many times before completing, and some jobs do not complete before the one week deadline that volunteers have to process each job. Using AES means that any processing done before the deadline is still useful and is not wasted. However the way AES is used on grid sites poses some challenges to the ATLAS@Home environment. From the beginning ATLAS@Home was designed such that volunteers never have access to sensitive credentials or require contact with any grid services. AES assumes that a job is able both to communicate with the central PanDA service and to write files to remote grid storage. The proposed architecture to address these issues is shown in Figure 5.

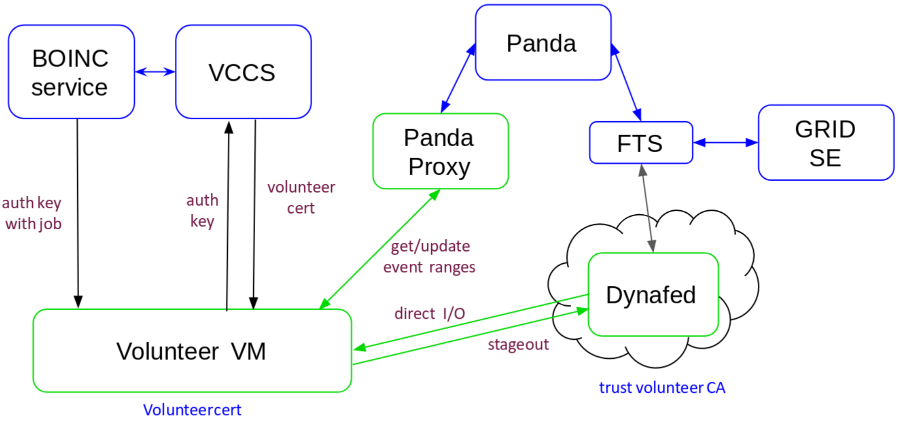

Figure 5. Integration of ATLAS Event Service into ATLAS@Home

The principle behind this architecture is to have a layer of protection or "sandbox" between volunteers and grid services. In regular ATLAS@Home the BOINC server provides such a sandbox: volunteers communicate with and transfer files to/from the BOINC server, and ATLAS services take care of the interface between this sandbox and the grid services. In Figure 5 the green boxes show the sandboxes used in the AES implementation of ATLAS@Home.

To avoid distributing credentials that allow direct access to grid services, credentials are obtained from a special Volunteer Computing Credential Service (VCCS) [6]. The BOINC client uses the BOINC user credentials (auth key) to access VCCS and obtain an X509 certificate which can then be used for grid services which trust the VCCS (i.e. the certificate can only be used for the special sandbox services and not the rest of the grid). The certificates are unique to each volunteer's BOINC account and thus any activities using it can be traced back to the volunteer.

Regular grid jobs use their X509 certificate to communicate directly with PanDA, but AES ATLAS@Home jobs are configured instead to contact Panda Proxy. Panda Proxy is a proxy for PanDA which trusts VCCS certificates and can forward requests on behalf of volunteers to PanDA. These operations are typically to request new events to process and to inform when events are finished being processed. To ensure that volunteers are restricted in the operations they can do in PanDA via Panda Proxy, they must use a token which is contained within each job which only allows operations on that specific job.

Instead of allowing volunteers direct contact with grid storage elements (GRID SEs), Dynafed [12] is used as a storage sandbox. Dynafed is a service which federates multiple distributed storage endpoints into a single unique namespace accessible through HTTP or WebDAV and is already in use by another LHC volunteer computing project, CMS@Home [13]. Dynafed trusts the credentials issued by VCCS and is used to host the input data required for the jobs and the output data produced by the jobs. The transfer from Dynafed to the final destination grid storage is done by the same File Transfer Service (FTS) [14] which replicates data between grid storage elements.

A prototype of AES for ATLAS@Home has been tested and has shown that the concept can work, however further testing will be needed in the future before AES is fully used in ATLAS@Home.

6 Conclusion

This paper has shown how recent improvements in ATLAS@Home have led to an order of magnitude increase in the amount of simulation that it is able to perform. Most of the increase comes from machines in data centres taking advantage of the native Linux app which is easier to use and more lightweight compared to using virtualisation. Using ATLAS@Home to backfill grid worker nodes can also help ATLAS exploit a previously untapped resource of CPU cycles without disrupting regular grid processing. In addition the work on integrating the ATLAS Event Service prepares ATLAS@Home for future directions of ATLAS computing.

We would like to thank the CERN IT department for hosting the BOINC infrastructure and help integrating ATLAS@Home into LHC@Home. We would also like to thank all our volunteers not just for resources provided over the years but for support in helping out others with problems. In addition we express gratitude to grid site admins willing to risk running ATLAS@Home alongside their grid jobs.

References

[1] C. Adam-Bourdarios, D. Cameron, A. Filipčič, E. Lancon, W. Wu, J. Phys.: Conf. Ser. **664**, 022009 (2015)
[2] ATLAS Collaboration, J. Inst. **3**, S08003 (2008)
[3] D. Anderson, *BOINC: a system for public-resource computing and storage*, in *Proc. 5th IEEE/ACM Int. Workshop on Grid Computing, GRID 04* (2004), pp. 4–10
[4] C. Adam-Bourdarios, R. Bianchi, D. Cameron, A. Filipčič, G. Isacchini, E. Lançon, W. Wu, J. Phys.: Conf. Ser. **898**, 052009 (2017)
[5] W. Herr, E. McIntosh, F. Schmidt, D. Kalchev, *Large scale beam-beam simulations for the CERN LHC using distributed computing resources*, in *Proc. 10th European Particle Accelerator Conference* (2006)
[6] L. Field, N. Giannakis, N. Hoimyr, D. Cameron, *Extending CERN computing to volunteers - LHC@home consolidation and outlook*, in *Proc. 23rd Conference on Computing in High Energy Physics (CHEP 2018)* (2019)
[7] G.M. Kurtzer, V. Sochat, M.W. Bauer, PLoS ONE **12**, e0177459 (2017)

[8] J. Blomer, C. Aguado-Sanchez, P. Buncic, A. Harutyunyan, J. Phys.: Conf. Ser. **331**, 042003 (2011)

[9] W. Wu, D. Cameron, A. Filipčič, *Backfilling the Grid with Containerized BOINC in the ATLAS computing*, in *Proc. 23rd Conference on Computing in High Energy Physics (CHEP 2018)* (2019)

[10] I. Bird, Annual Review of Nuclear and Particle Science **61**, 99 (2011)

[11] P. Calafiura, K. De, W. Guan, T. Maeno, P. Nilsson, D. Oleynik, S. Panitkin, V. Tsulaia, P.V. Gemmeren, T. Wenaus, J. Phys.: Conf. Ser. **664**, 062065 (2015)

[12] F. Furano, O. Keeble, L. Field, Phys. Part. Nuclei Lett. **13**, 629–633 (2016)

[13] L. Field, H. Borras, D. Spiga, H. Riahi, J. Phys.: Conf. Ser. **664**, 022017 (2015)

[14] A.A. Ayllon, M. Salichos, M.K. Simon, O. Keeble, J. Phys.: Conf. Ser. **513**, 032081 (2014)