EPJ Web of Conferences **245**, 03027 (2020) https://doi.org/10.1051/epjconf/202024503027
CHEP 2019

Adapting ATLAS@Home to trusted and semi-trusted resources

David Cameron[1,*], *Vincent* Garonne[1], *Paul* Millar[2], *Shaojun* Sun[3], and *Wenjing* Wu[4,**]

[1]University of Oslo, P.b. 1048 Blindern, 0316 Oslo, Norway
[2]Deutsches Elektronen-Synchrotron (DESY), Notkestrasse 85, 22607 Hamburg, Germany
[3]University of Wisconsin-Madison, Madison, WI 53706, USA
[4]University of Michigan, 500 S State St, Ann Arbor, MI 48109, USA

Abstract. ATLAS@Home is a volunteer computing project which enables members of the public to contribute computing power to run simulations of the ATLAS experiment at CERN's Large Hadron Collider. The computing resources provided to ATLAS@Home increasingly come not only from traditional volunteers, but also from data centres or office computers at institutes associated to ATLAS. The design of ATLAS@Home was built around not giving out sensitive credentials to volunteers, which means that a sandbox is needed to bridge data transfers between trusted and untrusted domains. As the scale of ATLAS@Home increases, this sandbox becomes a potential data management bottleneck. This paper explores solutions to this problem based on relaxing the constraints of sending credentials to trusted volunteers, allowing direct data transfer to grid storage and avoiding the intermediate sandbox. Fully trusted resources such as grid worker nodes can run with full access to grid storage, whereas semi-trusted resources such as student desktops can be provided with "macaroons": time-limited access tokens which can only be used for specific files. The steps towards implementing these solutions as well as initial results with real ATLAS simulation tasks are discussed along with the experience gained so far and the next steps in the project.

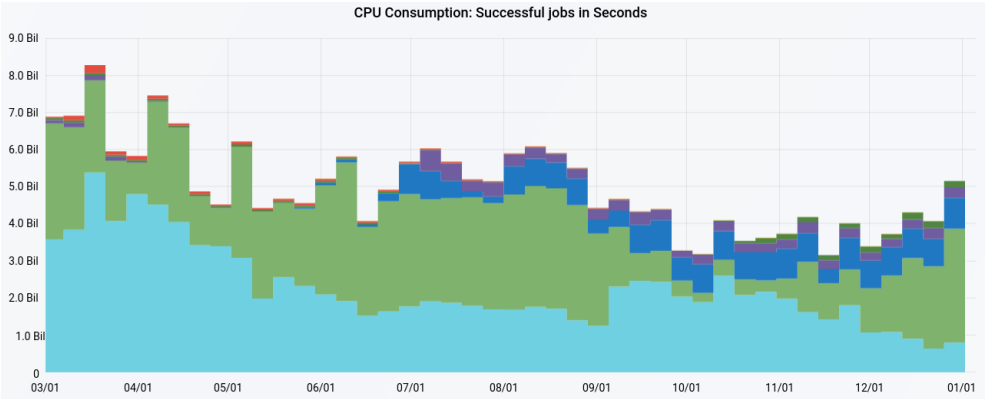
1 Introduction

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], a friendly graphical interface and the ability to run natively on Linux

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

platforms without requiring virtualization [5]. This last feature was key in exploiting backfilling of Grid sites, filling in the gaps in CPU usage left by inefficient jobs and scheduling [6]. Simple installation using a one-line command and the availability of Docker [7] containers have also lowered the barriers to participation. In 2019 ATLAS@Home processed just over 1 billion full simulation events, roughly 10% of the total full simulation events for ATLAS. Figure 1 shows the CPU consumption of ATLAS@Home jobs from March 1st, 2019[1] to December 31st, 2019, broken down by site running the jobs. The "BOINC" site consists of volunteers from the public and the others are ATLAS Grid sites running ATLAS@Home in backfilling mode. It can be seen that around half of these events were simulated using Grid site backfilling, and the other half from public volunteers. The overall size of ATLAS@Home varies over time depending on the types of tasks it is running and the types of tasks that are running on the Grid.



	min	max	avg	total
BOINC	546 Mil	5.379 Bil	2.241 Bil	100.848 Bil
AGLT2	247 Mil	3.734 Bil	1.988 Bil	89.474 Bil
NDGF-T1	0	856 Mil	420 Mil	18.909 Bil
TRIUMF-LCG2	0	571 Mil	181 Mil	8.153 Bil
UAM-LCG2	4 Mil	193 Mil	61 Mil	2.748 Bil
LRZ-LMU	0	215 Mil	34 Mil	1.538 Bil
praguelcg2	0	11 Mil	4 Mil	194 Mil
IN2P3-LAPP	0	2 Mil	541 K	24 Mil

Figure 1. CPU consumption per contributing site from March to December 2019, in CPU seconds per week.

ATLAS@Home was designed to handle untrusted resources, so all data transfers to and from the volunteer computer pass through a central sandbox rather than using Grid storage directly. However, worker nodes on Grid sites can be fully trusted, so it is safe to pass privileged credentials to BOINC jobs running in backfill there, meaning the jobs can interact directly with Grid storage. This allows the system to scale up by avoiding the central sandbox becoming a data transfer bottleneck. "Semi-trusted" resources are defined as computers belonging to ATLAS users, who should have rights to access all ATLAS data, but should not be given privileged credentials which could be used to modify or delete data. A slightly different solution to avoid the central data transfer bottleneck is required in this case.

[1] Grid site backfilling had been running before then but was only properly accounted in official ATLAS monitoring after this point.

In this paper we describe two methods for using trusted and semi-trusted resources in a more optimal way. Section 2 shows how the architecture of ATLAS@Home was modified to address fully trusted resoures and Section 3 presents the changes to integrate semi-trusted resources. Conclusions are drawn in Section 4.

2 A new architecture for trusted resources

The basic architecture of ATLAS@Home has remained unchanged since the beginning, and is depicted on the left-hand side of Figure 2. ATLAS@Home is integrated into the central ATLAS job management system (PanDA [8]) using Harvester/ARC Control Tower [9] and ARC CE [10], such that from the outside it looks like a regular Grid site. In this setup, the ARC CE performs data transfers between Grid storage and the BOINC data staging area, and the jobs running on the volunteer host download input data and upload output data to this data staging area. The proxy credential stays on the ARC CE and is never accessible by the volunteers.

A proposed architecture for using trusted resources is shown on the right-hand side of Figure 2. In this scenario, the proxy credential is passed all the way through to the job and the central data staging area, ARC Control Tower and ARC CE are no longer necessary. The jobs running on the Grid site worker nodes interact directly with Grid storage, as regular Grid jobs do.

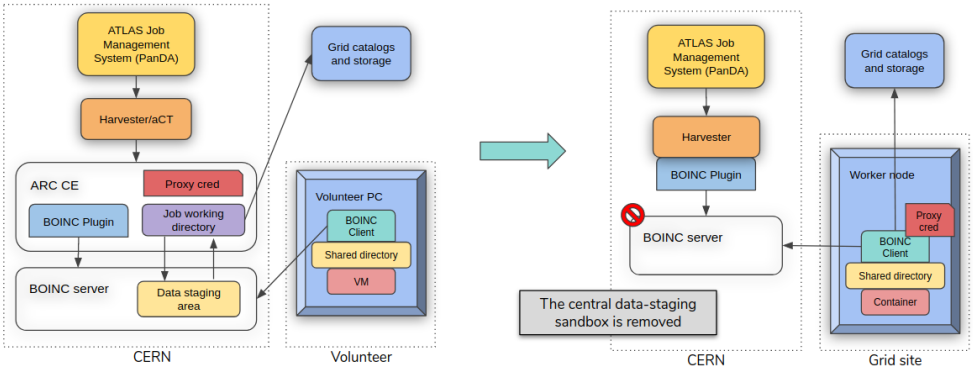


Figure 2. Evolution of the architecture of ATLAS@Home to support trusted resources.

This architecture has been implemented by writing a plugin to Harvester which interacts with the BOINC server using the python bindings of the BOINC WebRPC API [11]. The API allows submission, management and querying of batches of jobs, so the plugin maps the corresponding Harvester methods to the BOINC API:

- **Submit:** define a job which downloads the ATLAS pilot wrapper and runs it with the correct arguments;
- **Query:** query the BOINC server for the status of jobs;
- **Download:** at the end of the job, download the stdout from the BOINC server to Harvester's web area (the data output of the jobs are uploaded directly to Grid storage);
- **Cancel:** if PanDA decides to cancel a job, cancel the job in BOINC.

The proxy certificate is downloaded from the BOINC server when a new job is fetched and therefore this model requires a private locked-down BOINC server which is not open to

the public. For this purpose a special closed BOINC server was created on which accounts can be created on invitation only. This architecture closely resembles a vacuum-like model where pilots appear on worker nodes without being scheduled through a Computing Element or batch system. However, BOINC provides all the features that a vacuum model provides, in addition to the backfilling scheduling features that are intrinsic to the BOINC design, in a single framework.

This implementation exists as a prototype and real jobs have run through the system and completed successfully, however it has not yet been widely deployed. Currently the jobs read and write from and to a single Grid storage, whereas to scale up it is necessary that the jobs interact with their local Grid storage. This requires further work, so that a job knows on which site it starts running and can dynamically adjust its configured input and output locations.

3 Using semi-trusted resources with ATLAS@Work

ATLAS@Work is a project within the University of Oslo to use idle CPU computing resources on the High Energy Physics (HEP) group office desktops for ATLAS data analysis and have the results stored on a shared HEP disk storage space, all using existing ATLAS distributed computing services. The project consists of developing a framework based on ATLAS@Home to connect the desktops together and the provision of 200TB storage for keeping results, connected to the NDGF Tier-1 dCache. The architecture is illustrated in Figure 3.

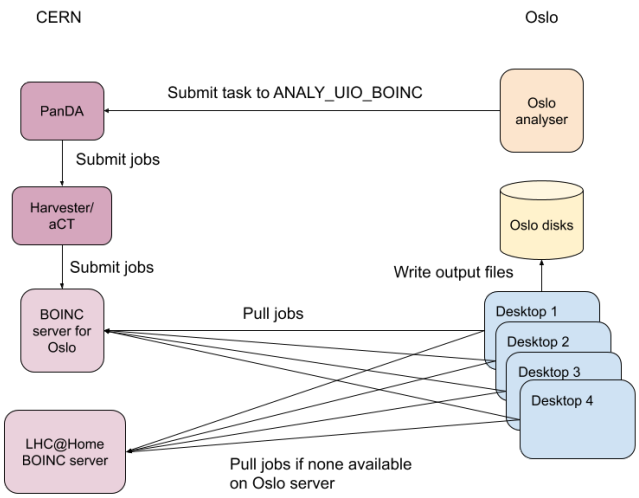


Figure 3. ATLAS@Work architecture.

HEP analysers submit their tasks to PanDA and ask them to be queued in ANALY_UIO_BOINC – a special PanDA "queue" – and that the final results should be stored on local disks. Only HEP analysers are allowed to use this special queue. PanDA prepares jobs for each task which are sent to a dedicated BOINC server through Harvester/aCT.

The desktops are connected to this BOINC server, and ask for tasks to process. The desktops can also be configured so that if no work is available from this server, they get regular ATLAS@Home tasks from the LHC@Home BOINC server. Once a task is finished the results are uploaded directly to the Oslo local storage. PanDA ensures the result is properly registered in the ATLAS data catalog so that it can be found later.

These desktops belong to ATLAS members, and are thus semi-trusted: they should have full access to read ATLAS data on Grid storage but not be given credentials that allow modifying or deleting data. The concept of macaroons [12] is used to allow limited write access to Grid storage. Macaroons are bearer tokens that contain a cryptographically strong list of caveats. Caveats are restrictions that place some limit on the use of a macaroon; for example, that it may only target a specific file, allow only read-only access, is only valid for a limited period, or is only valid from a specific IP address or subnet.

dCache supports macaroons [13] by allowing any authenticated user to request a macaroon. These macaroons have an enforced maximum lifetime and the bearer is only allowed (at most) those operations the macaroon-requesting user is authorised to do. When requesting a macaroon, the client can list additional caveats. Macaroons may also be embedded within an HTTP URL, creating a pre-signed URL.

Figure 4 shows the final implementation of ATLAS@Work. Macaroons are enabled on the NDGF Tier-1 dCache. When ARC CE receives a job, it uses the privileged proxy credential with which the job was submitted to request macaroons for each output file from the storage, asking for write permission for 24 hours on the specific output file path. It then passes the macaroons to the BOINC server along with the job. The macaroons are passed to the BOINC client when it requests a job, and are then used to upload the output files directly to the storage.

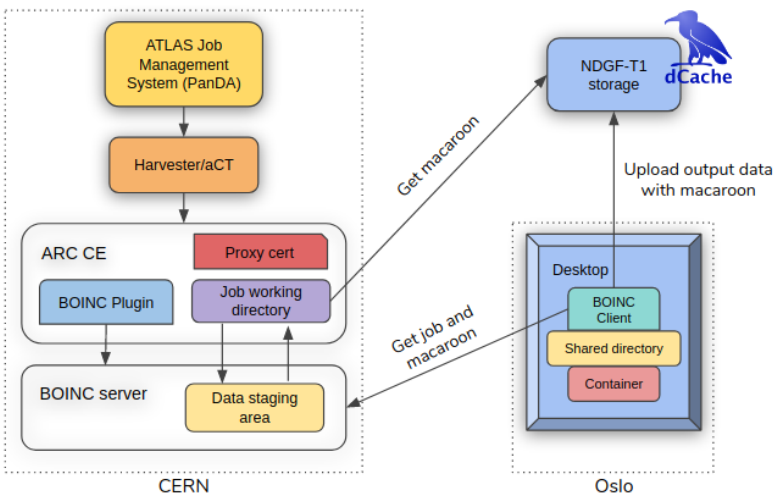


Figure 4. ATLAS@Work implementation.

ATLAS@Work has been used to run real tasks from analysers in the Oslo HEP group. They can submit tasks to PanDA in the same way as they submit regular Grid tasks and the ATLAS@Work system in the background ensures they run swiftly and efficiently on the cluster of office desktops.

4 Conclusion

This proceeding has shown how the ATLAS@Home architecture has been adapted for different trust models. A fully trusted resource can have direct access to Grid storage and thus avoid data transfer bottlenecks from a central sandbox. Semi-trusted resources can make use of macaroons to allow restricted time-limited write access to Grid storage. Future work may introduce macaroons to untrusted resources to allow direct access to storage instead of routing transfers through the central sandbox.

We would like to thank the CERN IT department for hosting the BOINC infrastructure for LHC@Home. We would also like to thank all our volunteers not just for the resources provided over the years but for the 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 and the analysers at the University of Oslo for testing out ATLAS@Work.

References

[1] C. Adam-Bourdarios, D. Cameron, A. Filipčič, E. Lancon, W. Wu, J. Phys.: Conf. Ser. **664**, 022009 (2015)
[2] ATLAS Collaboration, JINST **3**, S08003 (2008)
[3] D. Anderson, *BOINC: a system for public-resource computing and storage*, in *proceedings of 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] D. Cameron, W. Wu, A. Bogdanchikov, R. Bianchi, EPJ Web Conf. **214**, 03011 (2019)
[6] W. Wu, D. Cameron, D. Qing, Comput. Softw. Big. Sci. **3**, 8 (2019)
[7] D. Merkel, Linux J. **2014**, 239 (2014)
[8] T. Maeno, J. Phys.: Conf. Ser. **119**, 062036 (2008)
[9] T. Maeno et al., EPJ Web Conf. **214**, 03030 (2019)
[10] M. Ellert et al., Future Gener. Comput. Syst. **23**, 219 (2007)
[11] BOINC WebRPC API, `https://boinc.berkeley.edu/trac/wiki/RemoteJobs`
[12] A. Birgisson, J.G. Politz, U. Erlingsson, A. Taly, M. Vrable, M. Lentczner, *Macaroons: Cookies with Contextual Caveats for Decentralized Authorization in the Cloud*, in *proceedings of Network and Distributed System Security Symposium* (2014)
[13] A. Ashish et al., J. Phys.: Conf. Ser. **898**, 102009 (2017)