CMS@home: Enabling Volunteer Computing Usage for CMS

CMS@home: Enabling Volunteer Computing Usage for CMS

L Field[1], H Borras[2], D Spiga[1] and H Riahi[1]

[1] CERN, Geneva, Switzerland
[2] University of Heidelberg, Germany

E-mail: laurence.field@cern.ch

Abstract. Volunteer computing remains a largely untapped opportunistic resource for the LHC experiments. The use of virtualization in this domain was pioneered by the Test4Theory project and enabled the running of high energy particle physics simulations on home computers. This paper describes the model for CMS to run workloads using a similar volunteer computing platform. It is shown how the original approach is exploited to map onto the existing CMS workflow and identifies missing functionality along with the components and changes that are required. The final implementation of the prototype is detailed along with the identification of areas that would benefit from further development.

1. Introduction

The Compact Muon Solenoid (CMS) experiment is a large general-purpose particle physics detector that observes proton−proton collisions produced by the the Large Hadron Collider (LHC). It generates large quantities of data in the form of discrete events that represent the detector's response to the very small signals induced by the collision debris. After a reduction of events by dedicated electronics and a large computing farm located close by, the resulting data is sent to the CERN Computer Centre for archiving. The data is then distributed, processed, and analysed using the Worldwide LHC Computing Grid (WLCG) [1], by the thousands of physicists that are participating in the CMS collaboration.

In addition to the data produced by the detector, physics models are used to provide simulations (i.e. a virtual detector) that can be compared to the experiment data. The amount of data generated by such activity is comparable to that generated by the detector itself and is computationally very intensive. Over the lifetime of the LHC, its power and intensity will be increased, which along with detector and data acquisition improvements, will lead to higher data rates [1]. However, with no anticipated increase in the amount of resources provided by the WLCG collaboration other than those offered by technological trends, the exploitation of opportunistic resources is being investigated.

Volunteer computing is hence of interest as an additional source of resources for computational intensive tasks that could help to bridge this shortfall. It is a type of distributed computing where the owner of a computational resource can donate computing capability to a project or cause. Although the resources are provided for free, there is an associated cost in order to exploit those resources for the intended purpose. From a technical perspective the computation task needs to be adapted for such an environment and the resources need to be integrated into the existing workflows.

This paper defines an architecture that can be used by CMS to exploit opportunistic resources provided using volunteer computing and presents the experience from employing this in a prototype project, CMS@home. Section 2 describes the use of volunteer computing for high energy physics applications and outlines the main challenges in particular for CMS. The architecture for the CMS@home project is described in Section 3 along with a description of how each component was realized. Finally the experience of building a prototype project is documented in Section 4 along with some concluding remarks in 5.

2. The Adoption of Volunteer Computing for CMS

Volunteer computing emerged in the mid 1990s, around the same time as Grid computing. Of note is the SETI@home project [2] that was launched in 1999 to analyse signals received by the Arecibo Observatory's radio telescope for signs of extra terrestrial intelligence. The project attracted several hundred thousand volunteers and proved the viability of such an approach even though to date no evidence for extra terrestrial intelligence signals has been found. In 2002, the Berkeley Open Infrastructure for Network Computing (BOINC) project [3] was founded to provide a middleware system for volunteer computing. BOINC provides a client (including a GUI) for the volunteers' machines and a project server (including Web site) that together can be used to run applications on the volunteer's machine.

In 2004 BOINC was used to build the LHC@home project [4] to investigate the stability of particles which experience head-on and long-range beam-beam effects for different optical configurations and machine imperfections. More recently the Test4Theory project [5] pioneered the use of virtualization with BOINC and paved the way for using volunteer computing for high energy physics applications. To achieve this, the CernVMwrapper [6] (and later the vboxwrapper) applications were developed to control a Virtual Machine (VM) using the VirtualBox hypervisor [7]. From the BOINC client's perspective, the wrapper is just another application but from the project's perspective, there is a resource pool available comprised of VMs. This approach follows the Vacuum model [8] which is defined as a scenario in which virtual machines are created and contextualized for experiments by the resource provider itself rather than the resource consumer. The contextualization procedures are supplied in advance by the experiments and launch clients within the virtual machines to obtain (*pull*) work from the experiments' central queue of tasks. From the perspective of the experiments, VMs appear by *spontaneous production in the vacuum* [8]. Like virtual particles in the physical vacuum: they appear, potentially interact, and then disappear [8].

As stated above, to use the Vacuum model the VMs need to be contextualized so that they can obtain work from a central task queue. However, the two workload management systems used in CMS (Crab3 [9] and WMAgent [10]) do not provide such a feature and follow the push model for job submission. In addition existing Grid resources are considered to be trusted while volunteered resources are untrusted as anyone can register to become a volunteer. Together, the absence of a central task queue and the integration of untrusted resources are the main challenges faced in adopting volunteer computing for CMS.

To join the push-based job submission with pull-based VMs, an entity is required in the middle to bridge between the models. The job description and any input files can be pushed onto the bridge and a job agent in the VM can pull them from the bridge. This approach also has the advantage that one credential type can be used to push (write) and another or any can be used to pull (read). As the resources are untrusted, any output files should be sandboxed and validated before being assimilated for further use. Hence a bridge is also needed for the data generated by the volunteered that supports push (write) with the volunteers credential. Once validated the data can then be transferred from the sandbox to another storage system.

In the Test4Theory project, this bridge was provided by the CoPilot framework [11], developed as part of the CernVM project [12].

3. The Architecture and Implementation of the CMS@home project

The architecture of the CMS@home project can be seen in Figure 1. It shows the main components and their interactions.

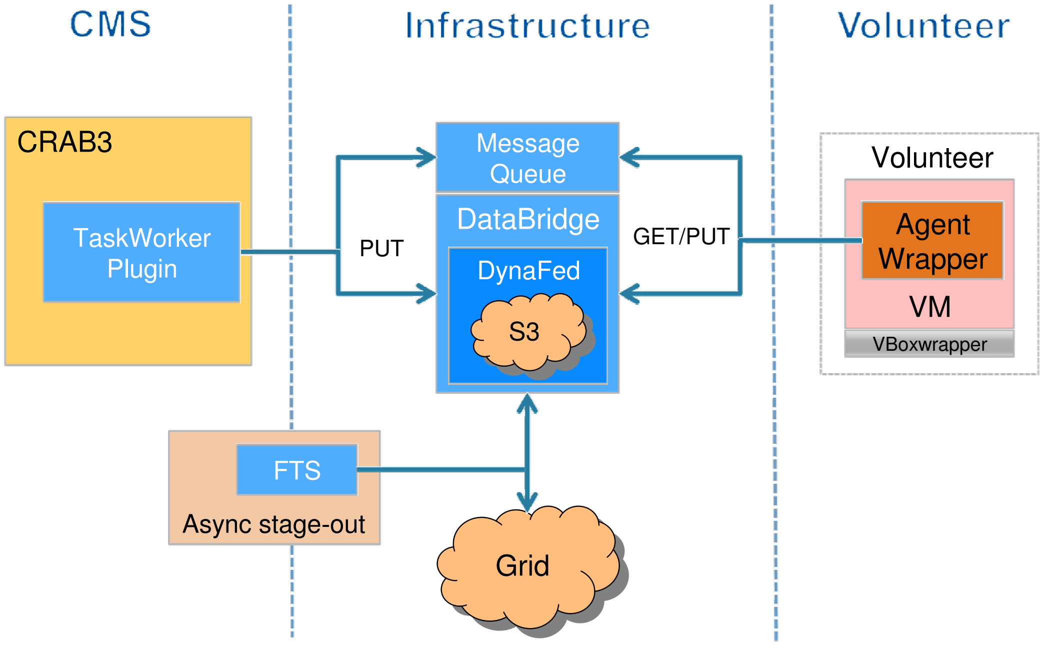

Figure 1. The architecture of the CMS@home project

Central to the architecture is the concept of the bridge in this case called the DataBridge. In fact there are two bridges; one for the job description and another for the job input/output files. For the job description the bridge is implemented using a message system which preserves ordering using a FIFO queue. Job descriptions are small in size O(Kb) but there are many of them O(100K) and many potential clients O(100K). The *DataBridge* is used to stage job input/output files. Here scalability is a concern as each file may be large O(100Mb) and the total data volume that must be supported is at least twice the number of jobsfread O(Tb).

3.1. Authentication

When using the vboxwapper, the volunteer's BOINC ID and an authenticator can be read from /dev/fd0 within the VM. The BOINC server itself is essentially an Identity Provider (IDP) and can be used to validate the volunteers credentials. By using the mod_auth_mysql plugin [13] for Apache [14], HTTPS requests from the VM can be authenticated by mapping the BOINC ID and Authentication key to the username and password. The credentials in this request can then be validated against the user table in the database used by the BOINC server. This method allows

Apache-based components that support HTTPS to be considered to implement components of the architecture.

3.2. DynaFed

The Dynafed [15] federates storage over HTTP and is being evaluated by LHCb and ATLAS. Clients are redirected using the HTTP redirect function to the storage system where a requested file can be found. The redirection can consider smart replica selection criteria such as availability and proximity, and offers great scalability potential as multiple storage systems can be used. It can federate WebDAV or S3 enabled storage systems and has an Apache front-end. As such it was adopted to implement the DataBridge functionality. Two buckets were created in the Ceph [16] storage system at CERN: one for input files and another for output files. The Dynafed was configured so that the x509 credential of the Crab3 server could be used to PUT input files in the input bucket and the credential of the volunteer could be used to GET the input files and PUT the output files in the output bucket.

3.3. Message Queue

Rather than extending an existing messaging system to support the credential of the volunteer, a simple message queue was implemented on the DataBridge using the python-dirq library [17]. Two CGI scripts were created; one to handle a HTTP PUT request to add a job description to the queue and another to handle a HTTP GET request to retrieve a job description from the queue. The PUT CGI script requires authentication using the x509 credential of the Crab3 server and the GET CGI script is authenticated using valid credentials of volunteers.

3.4. DataBridge

Together the Dynafed and message queue provide the functionality for the DataBridge that supports the following workflow. Job descriptions can be added to the message queue on the DataBridge by using HTTPS PUT and a specified x509 credential. Input files can be added to the input bucket in Ceph storage using HTTPS PUT and a specified x509 credential. The job description can be obtained from within the VM using the credential of the volunteer and a HTTPS GET request to the DataBridge. URLs for the input files provided in the job description can be used to obtain the input files from within the VM using the credential of the volunteer and a HTTPS GET request to the DataBridge. After running the job, the output files can be uploaded to the output bucket in Ceph storage using the credential of the volunteer and a HTTPS PUT request to the DataBridge.

3.5. Job Submission Plugin

Crab3 is one of two workload management systems used by CMS and offers a plugin mechanism for job submission methods. It was selected for use with this workflow as the plugin mechanism simplified the integration and local expertise was available. This mechanism relies upon a TaskWorker which defines how tasks should be handled. When a user submits a task the TaskWorker will divide the task into multiple jobs. It then starts submitting those jobs and a static variable is used to select the BOINC plugin for job submission. The plugin then obtains the required input files and job descriptions and pushes them onto the DataBridge using HTTPS PUT. At the moment only production (simulation) jobs have been considered that do not require large input files. In addition, no end user credentials are provided.

3.6. Job Agent

On the volunteers' side a job agent was created to interact with the DataBridge and run jobs. When a volunteer connects to the CMS@home BOINC project, the next time their machine is idle the BOINC client will automatically start to download a pre-baked image of CernVM 3.3 [12]. The VM is started and a cron job is run every minute in the VM to check if the job agent is running and if not starts it. The agent uses HTTP GET to request a job description from the DataBridge and if none is available it receives a HTTP 404 error. Once a job description has been retrieved, the agent will parse it to obtain the URLs for the input files and uses HTTPS GET to download them from the DataBridge. It then runs the job, for which the software can be found in CVMFS [12], and uploads the output files (including meta-data) to the DataBridge using HTTPS PUT. After the job has completed, another job will be started by the cron job. As a result the VM is not shutdown and therefore restarting for each job is avoided. This reduces disk as well as bandwidth usage, as the number of times that CVMFS needs to download the same files is reduced.

3.7. Stage-out

A cron job running on the DataBridge automatically reads the meta-data and upload it into the database of the AsyncStageOut (ASO) system [18] of CMS. ASO will then instruct File Transfer Service (FTS) to transfer the result files from the DataBridge into a user defined folder of a storage system within the Grid infrastructure. Once the transfer is completed, the ASO interacts with the CMS catalogue to make the result files available to the community for further analysis.

4. The Initial Deployment of the CMS@home project

A standard BOINC server was deployed as a service by the CERN IT-PES group as seen in Figure 2. This represents the public-facing side of the project as well as the core infrastructure used by the BOINC client. The BOINC Web site provides features such as account management, server status, message boards and statistics. Within a few days of opening the project, volunteers registered, even in the absence of advertisements or announcements. As of writing, the project has nearly 100 active volunteers providing around 300 machines. The message boards proved valuable in communicating to the community of volunteers, many of whom were long time BOINC users and provided precious feedback.

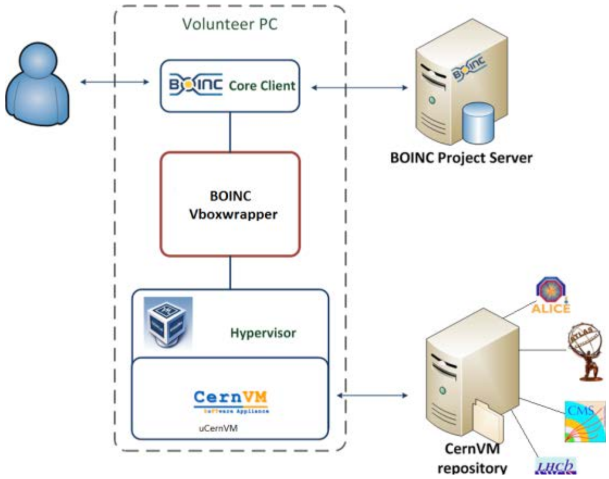

Figure 2. BOINC server infrastructure at CERN

A CMS application was created on the BOINC server that used the vboxwrapper approach which required the creation of a VM image. This application contains the BOINC VirtualBox wrapper, the pre-baked CernVM image, and necessary configuration files. The image was created using the microCernVM 3.3 image and using VirtualBox to contextualize and run one CMS job so that the software is already cached in the image. This image was copied to the BOINC server and a release of the CMS application was made. As credit is only assigned when the task has finished, the VMs can not run indefinitely but must be terminated after a period of time. The duration of a task is configurable and was set to 24 hours, the same as for Test4Theory. As BOINC provides its own task queue, it must be ensured that tasks are always available. The BOINC server provides a mechanism to ensure this and it is by using this feature that the vacuum model is implemented. This means that whenever a BOINC client connects to the BOINC server it will always find a new task and start the VM.

The developers, supporters, and volunteers of the CMS@home project simply configure their BOINC clients to attach to the CMS@home project server. The CMS application is available for the 64bit Windows, Linux and Mac platforms and has been verified to work. On starting the VM the workflow is as follows: job descriptions are requested, input files downloaded, jobs run, results uploaded and after this has been repeated for 24, the BOINC task ends which triggers the assignment of BOINC credit. Monitoring of the activity can be done using the Apache access logs. Additional development effort has provided a feature that supports the integration of activity on the DataBridge based on information from the Apache logs. The server itself shows activity relating to the BOINC tasks; however they represent resource provision activity rather than real CMS job activity.

On the volunteers' side, two features were added to the job agent to support monitoring of the job by the volunteer. The first is multiple VM console support that allows different log files (and the top command) to be seen in the different consoles. The other is graphics support that allows the same log files to be downloaded from a Web server residing within the VM. The ability to download log files so that they can be added to messages in the forums can significantly help debugging activities. Both features are part of the standard BOINC client but require the application to support them. The graphics feature can be extended to show visualizations of the data generated on the volunteers machine, as in Test4Theory.

5. Conclusion

Volunteer computing can and is providing significant additional computing resources O(100K) machines to various projects. In this paper the feasibility of developing a volunteer computing platform for CMS was investigated. This result is that an advanced prototype for a CMS@home project now exists. Furthermore, the project already has an established presence with real volunteers providing resources.

An approach following the Vacuum model where VMs appear, potentially interact, and then disappear was adopted. In order to achieve this the challenge of integrating a job agent residing in a VM on an untrusted resource with the CMS job submission framework was investigated. The solution proposed was the DataBridge which can be used to stage-in and stage-out input and output files between the volunteer and the wider Grid infrastructure. It reuses the dynafed component to access the CERN Ceph storage service, and support for BOINC authentication was added. This has been successfully used for the initial deployment of a volunteer computing platform for CMS that is running a MinBias MC generation workload.

Currently the CMS@home project is being exercised with a MinBias MC generation workload which is manually injected. In order to send useful work to this resource pool, further work is required to fully integrate the volunteer computing infrastructure with the CMS workflows. Once more diverse workloads can be supported, the focus will shift towards scaling the project. This will include making it more attractive to volunteers by creating CMS@home Web pages and improving the internal Webapp. Once production-ready, the project can be added as a Beta to the vLHC@home project and benefit from an existing pool of volunteers and future outreach initiatives.

6. Acknowledgements

We would like to thank Ben Segal (CERN-IT) and Ioannis Charalampidis (CERN-PH) for their consultancy. The BOINC server infrastructure was provided by the CERN IT-PES group (Nils Hoimyr, Tomi Asp and Pete Jones). We would also like to thank all the volunteers, especially those who were active on the message boards and helped us to evolve the project. Without volunteers we can not do volunteer computing.

7. References

[1] Ian Bird. Computing for the Large Hadron Collider. *Annual Review of Nuclear and Particle Science*, 61(1):99–118, November 2011.

[2] E. Korpela, D. Werthimer, D. Anderson, J. Cobb, and M. Leboisky. SETI@home-massively distributed computing for SETI. *Computing in Science & Engineering*, 3(1):78–83, February 2001.

[3] D.P. Anderson. BOINC: A System for Public-Resource Computing and Storage. pages 4–10. IEEE, 2004.

[4] W Herr, D Kaltchev, E McIntosh, and F Schmidt. Large Scale Beam-beam Simulations for the CERN LHC using distributed computing. volume C060626, pages 526–528, Edinburgh, UK, June 2006.

[5] Daniel Lombraa Gonzle, Francois Grey, Jakob Blomer, Predrag Buncic, Artem Harutyunyan, Miguel Marquina, Ben Segal, Peter Skands, and Anton Karneyeu. Virtual Machines & Volunteer Computing: Experience from LHC@Home: Test4theory project. In *PoS(ISGC 2012)036*, Taipei, Taiwan, March 2012. Proceedings of Science.

[6] Cernvm wrapper. `{http://dx.doi.org/10.5281/zenodo.17508}`, author = Daniel Lombraa Gonzlez and Ioannis Charalampidis and VolatileStorm and Julien Nabet.

[7] Oracle VM VirtualBox. `https://www.virtualbox.org/`. Accessed: 2015-05-14.

[8] A McNab, F Stagni, and M Ubeda Garcia. Running Jobs in the Vacuum. *Journal of Physics: Conference Series*, 513(3):032065, June 2014.

[9] M Cinquilli, D Spiga, C Grandi, J M Hernndez, P Konstantinov, M Mascheroni, H Riahi, and E Vaandering. CRAB3: Establishing a new generation of services for distributed analysis at CMS. *Journal of Physics: Conference Series*, 396(3):032026, December 2012.

[10] D. Evans, D. Mason, O. Gutsche, S. Metson, S. Wakefield, D. Hufnagel, A. Hassan, A. Mohapatra, M. Miller, and F. van Lingen. Large Scale Job Management and Experience in Recent Data Challenges within the LHC CMS experiment. In *PoS(ACAT08)*, Erice, Italy, November 2008. Proceedings of Science.

[11] A Harutyunyan, C Aguado Snchez, J Blomer, and P Buncic. CernVM Co-Pilot: a Framework for Orchestrating Virtual Machines Running Applications of LHC Experiments on the Cloud. *Journal of Physics: Conference Series*, 331(6):062013, December 2011.

[12] P Buncic, C Aguado Sanchez, J Blomer, L Franco, A Harutyunian, P Mato, and Y Yao. CernVM a virtual software appliance for LHC applications. *Journal of Physics: Conference Series*, 219(4):042003, April 2010.

[13] mod_auth_mysql. `http://modauthmysql.sourceforge.net/`. Accessed: 2015-05-14.

[14] The Apache HTTP Server Project. `http://httpd.apache.org/`. Accessed: 2015-05-14.

[15] Fabrizio Furano, Ricardo Brito da Rocha, Adrien Devresse, Oliver Keeble, Alejandro lvarez Aylln, and Patrick Fuhrmann. Dynamic federations: storage aggregation using open tools and protocols. *Journal of Physics: Conference Series*, 396(3):032042, December 2012.

[16] Ceph. `http://ceph.com/`. Accessed: 2015-05-14.

[17] python-dirq. `https://github.com/cern-mig/python-dirq`. Accessed: 2015-05-14.

[18] Hassen Riahi, Tony Wildish, and Diego Ciangottini. AsyncStageOut: Distributed user data management for CMS Analysis. In *In these proceedings*.