Computing and Software for Big Science (2018) 2:2
https://doi.org/10.1007/s41781-018-0006-z

ORIGINAL ARTICLE

CMS@home: Integrating the Volunteer Cloud and High-Throughput Computing

L. Field[1] · D. Spiga[2] · I. Reid[4] · H. Riahi[1] · L. Cristella[3]

Received: 28 April 2017 / Accepted: 31 January 2018 / Published online: 19 March 2018

Abstract
Volunteer computing has the potential to provide significant additional computing capacity for the LHC experiments. Initiatives such as the CMS@home project are aiming to integrate volunteer computing resources into the experiment's computational frameworks to support their scientific workloads. This is especially important, as over the next few years the demands on computing capacity will increase beyond what can be supported by general technology trends. This paper describes how a volunteer computing project that uses virtualization to run high energy physics simulations can integrate those resources into their computing infrastructure. The concept of the volunteer cloud is introduced and how this model can simplify the integration is described. An architecture for implementing the volunteer cloud model is presented along with an implementation for the CMS@home project. Finally, the submission of real CMS workloads to this volunteer cloud are compared to identical workloads submitted to the grid.

Keywords Volunteer Computing · High Energy Physics · BOINC · High-Throughput Computing

Introduction

As the LHC's scientific programme advances, its power and intensity will be increased and the future demands on computing capacity will be beyond what can be supported by general technology trends. Volunteer computing is of interest, as it has the potential to provide significant additional computing capacity for the LHC experiments. The CMS@home project therefore aims to provide addition computing capacity for the Compact Muon Solenoid (CMS) experiment using donated computing resources from volunteers. An initial prototype solution for the CMS@home project has been developed and further details can be found in [1]. The adoption of volunteer computing brings an additional operational cost, which can be reduced by building upon the existing operational supports and transparently integrating those resources. The emergence of the cloud computing paradigm had resulted in the LHC experiments revising their computational frameworks to support the direct integration of virtualized resources. By considering volunteer computing projects that exploit virtualization as a volunteer cloud, a similar integration path can be followed.

This paper builds upon the experience gained through operating the initial prototype and suggests optimizations of the model to simplify the integration and lower the operational cost for CMS. Specifically, this involves considering the virtualized volunteer computing resources as a cloud and overlaying a solution for high-throughput computing. As it is envisaged that the high-throughput computing layer is common for all types of resources, the overall operations cost can be reduced and in addition similar provisioning methods can be used for the virtualized resources.

The next section provides an introduction to the volunteer cloud concept and compares with this the traditional view of cloud computing, along with how such resources can be used for high-throughput computing. Section 3 describes the revised architecture and shows how high-throughput computing has been overlaid upon the volunteer computing resources and integrated with the CMS computational framework. The results from using this improved deployment are given in Sect. 4 and are compared with identical submissions to the grid.

✉ L. Field
laurence.field@cern.ch

[1] CERN, Geneva, Switzerland

[2] Universita e INFN, Perugia, Italy

[3] Universita e INFN, Bari, Italy

[4] Brunel University, London, UK

The Volunteer Cloud

The Berkeley Open Infrastructure for Network Computing (BOINC) project [2] is a middleware for volunteer computing. It provides a server for building the project and a client that runs on the volunteers' machines. Projects such as SETI@home [3], which was launched in 1999 to search for signs of extra-terrestrial intelligence by analysing signals received by the Arecibo Observatory's radio telescope, have attracted several hundred thousand volunteers. A community has been formed around these projects based on BOINC with volunteers participating in multiple projects. It is therefore advantageous to build upon BOINC and its community, as an existing resource pool of volunteers has already been established. More importantly, BOINC has gained the trust of these voluteers so that they accept that the applications are executed on their computers. Hence, the LHC@home project [4] was built upon BOINC to investigate the stability of particles in the accelartor for different optical configurations and machine imperfections.

However, as high energy physics applications are only available for Linux, this reduces the potential resources pool as Windows and Mac machines cannot be used. The use of virtualization with BOINC by the Test4Theory project [5] attempted to solve this issue and paved the way for using volunteer computing for high energy physics applications. This approach provides an application, the vboxwrapper (previously the CernVMwrapper [6]), which runs a virtual machine for a fixed period of time. The virtual machine image run is under the control of the project and can be contexutlized similarly to how VMs would be configured in a cloud resource. From the volunteer's perspective, the wrapper is just another application, but from the project's perspective, this is just another VM in a resources pool. This approach follows the vacuum model [7], whereby virtual machines are created by the resource ower and the virtual machine image is provided by the consumer. For CMS, the virtual machine image launches agents (pilots) within the virtual machines to obtain (pull) work from a central queue of tasks. From the perspective of the resource consumer, VMs appear by spontaneous and, like particles in the vacuum, appear, potentially interact, and then disappear [7]. The CMS@home project has implemented this model whereby BOINC is used to provision VMs which then connect to CMS's computing infrastructure to retrieve jobs.

Architecture and Implementation

The CMS@home Architecture

The overall architecture has been designed based on three guiding principles that emerged during the initial proof of concept [1] as well as experience from other similar projects [5]. These three principles are:

- The use of volunteer computing resources must appear to be seamless from the perspective of the project scientist[1]. It must therefore be fully compliant with the existing computing infrastructure and at most only a few configuration parameters should need changing in the existing workload management system.
- As the volunteer's resources are considered untrusted, since anyone can register, and grid resources are trusted, it is crucial to carefully manage the resulting hybrid ecosystem to ensure security is not compromised.
- Output data produced by volunteers resources must be quarantined and validated before being migrated to the trusted domain.

Furthermore, the solution for exploiting donated resources must result in a sustainable system. This means that the ongoing operation cost must mostly be met by the existing operational support structures. The challenge is to provide this in an experiment agnostic way that can potentially be a generic solution for other experiments. The prerequisite is to decouple resource management from job/workload management. Achieving this is possible when following the vacuum model as already stated in Sect. 2. The result of this strategy is that the job definition, payload preparation and project scientist interface as well as all the application dependencies are completely within the experiment's domain and can control this system regardless of the type of computing resource used.

A queue can be used to implement an asynchronous mechanism for volunteer computing resources to pull a job. It can be used as an interface to bridge the two independent systems: workload management and resource management. Using such an approach can satisfy the first principle whereby a project scientist can continue to use the very same system to submit workloads. However, resources should be authenticated so that only registered volunteers can pull workloads from the queue and the result returned can be traced to a specific volunteer. This

[1] The term project scientist is used in the BOINC nomenclature to describe the function of the scientist who is submitting the computational workloads.

step must be agile enough to handle the dynamic and chaotic participation of the volunteers.

As the grid authentication mechanism requires a number of checks to verify the identity and authenticity of an individual, it cannot be used in the context of volunteer computing. However, the second principle can be met using the same technology but with a different policy, i.e. just check that the volunteer is registered. The policy used is that of the BOINC project and hence the term registered is therefore defined by the implementation of that policy within the BOINC project. Having such an authentication mechanism allows access to be restricted to only registered users, the blocking of individual registrations and attribution of operations to specific registrations. The result is that the resources are authenticated and authorized, but remain untrusted in the wider context of the grid infrastructure.

Finally, as the outputs are produced on untrusted resources, they should be quarantined and validated before being migrated to trusted storage where further process can occur. This has implications on both the technologies used and output validation policies. From a technology perspective, the solution must be interoperable with grid storage systems to satisfy the first principle. The validation must assume that the output may contain malicious files and hence they must be temporarily staged before being validated and migrated to storage, satisfying principle number three. For the workload used, this validation is achieved during the *merge* step. The merge jobs are submitted to combine the results into fewer, larger files before further analysis is carried out using the grid. Any corrupted results should be rejected at this stage and further physics validation can be done. Files corrupted during transfer to the data bridge are already rejected via checksum validation and play no further part. Maliciously corrupted files are unlikely as the malefactor would need both an intimate knowledge of the CMS software and the ability to compromise the VM. Hence corrupted files are unlikely, and currently the merge jobs show a zero failure rate. Inspection of the merge logs of a recent batch of jobs showed 1,773 merge jobs, combining 104,800 result files (representing 262 million generated events giving 6.91 million processesd events) with no evidence seen of a file error.

As a result of these considerations, three pillars of the architecture were identified and developed:

(1) A job queue to collect job descriptions and possibly payloads as they are generated by the experiments.
(2) A method to translate the volunteer's credential from a BOINC username and password that is available on the resource to one that is compatible with the grid authentication technology.
(3) A hybrid storage system to which files can be uploaded from volunteer's resources and also interact with the standard grid infrastructure and tooling.

The Implementation and Integration for CMS@home

The first goal is to contextualize VMs so that they can pull work from a central task queue. Within CMS, there are two workload management systems: CRAB3 [8] and WMAgent [9], which have distinct roles and responsibilities. This paper mainly discusses CRAB3; however as both systems rely on HTCondor [10] for the underlying technology used to manage both the resources and jobs, the same discussion holds for the WMAgent. The CMS uses the Condor Glidein [11] mechanism to implement a pilot-based system for resource management. These glideins are placed on the resource by GlideinWMS. In the context of volunteer computing, the GlideinWMS cannot be directly used, as it submits (pushes) the glideins to the batch system that is managing the resources. However, the condor_schedd, that is the job queue used by the glideins, can provide the first pillar of the architecture. Hence, the condor_schedd was adopted to provide implementation of the queue where the jobs are pushed by CRAB3 as usual. The VMs were then contextualized to pull from the queue via the direct instantiation of the HTCondor glidein. From the perspective of the project scientist, the volunteer resources appear as a regular batch system and only a small configuration change is required to direct the jobs to that site. All the specific technical details for accessing volunteer resources are therefore hidden from the project scientist and hence it is seamless, while at the same time there is nothing additional for the experiment to maintain and hence it is sustainable.

For the glideins to connect to the condor_schedd, a credential is required. This is usually a trusted grid credential that is furnished by the GlideinWMS. To provide an untrusted grid credential, the Volunteer Computing Credential Service (VCCS) was developed to generate a short-lived (7 days) X.509 proxy from the BOINC username and password. Given that the BOINC username and password are available from within the VM, an HTTPS request is made to the VCCS for the proxy. The VCCS is a Web service that verifies the incoming request against the BOINC DB using Apache's mod_dbd and, once validated, OpenSSL generates a certificate request. This request is used to contact an online certification authority (CA), specifically set up for the purpose of signing the X.509 certificate that was generated. The signed certificate is then used to generate a short-lived X.509 proxy for the volunteer. After the condor_schedd has been updated to recognize this volunteer computing CA, this untrusted (in the grid sense) X.509 proxy can be used to pull jobs. The VCCS has a configurable cache for the generated proxies to reduce the number of calls to the online CA.

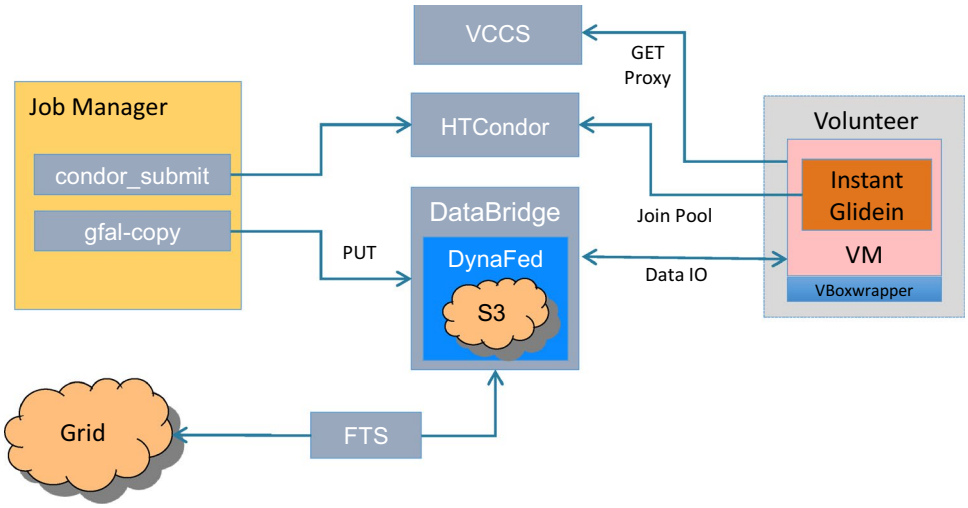

Fig. 1 The CMS@home Architecture

Furthermore, the lifetime of the short-lived X.509 proxy is also configurable and can be tuned to take into consideration the lifetime of the VM.

Finally, the produced output must be temporarily stored before being validated and migrated to a trusted storage for further processing. This has been implemented using the Data Bridge, a storage system that supports multiple authentication methods with associated policies. The Data Bridge is implemented using the Dynafed component which was developed in the context of HTTP data federations. Here, it enables volunteers to be authenticated with one type of credential and write to a storage system that uses another credential without the volunteer having access to that credential. This is achieved by first authenticating the volunteer with their X.509 proxy using Apache's mod_ssl. Dynafed then returns a signed HTTP redirect and from contacting the underlying S3 (Ceph) storage system. By providing different authorization policies depending on the DN structure of the X.509 proxy, this approach also has the advantage that the volunteer credential can only be used to push (upload) a new file and only the trusted grid credential can be used to pull (read) and delete a file. Hence, the Data Bridge can be used to quarantine the data generated by the volunteers and once validated the data can then be copied to another storage system using the standard data management tools of CMS such as the AsyncStageOut Service (ASO).

A complete schema of the CMS@home architecture is represented in Fig. 1.

The Initial Results

During the initial development of the project, CRAB3 was used to submit batches of jobs running CMS software to generate Monte Carlo events using the proposed geometry for the Phase 2 upgrade to the CMS silicon outer tracker detector designed to operate with the high-luminosity LHC [12].

Two types of collision events were generated, minimum bias (i.e. background) and top–antitop (ttbar) creation. A limitation on volunteer computing is that many home networks are still connected to the Internet via ADSL—asymmetric digital subscriber line—where typically the download bandwidth is of the order of 5–10 Mbps, but the upload speed is significantly less, often 1 Mbps. This means that jobs need to be tailored towards the lowest common denominator, especially with regard to uploading result files. The job parameters were adjusted to give average running times in the order of 1 h, and output files about 50 MB in size; this led to jobs of 250 events for minimum bias and 50 events for ttbar. Such jobs require, on average, nearly 150 kbps of upload bandwidth per concurrent job.

For a statistical comparison with standard grid jobs, batches of 2000 25-event ttbar jobs were submitted to both CMS@home and the grid. The number of result files received as a function of time from the batch submission is shown in Fig. 2. On the grid, with a large number of high-performance hosts available, return times were relatively short, with the first results coming in after 30 min and 90% (1800) of the expected results being received in about 6 h. Unusually, a large number of result files (142, or 7.1%) were never received.

The CMS@home jobs, on the other hand, started returning results after about 80 min, but because of the small number of available volunteer hosts (about 100 at the time) only a limited number were run simultaneously and the return graph (Fig. 2b) shows an even slope for much of its duration as results were returned at a constant rate. 90% of the results were received in 29.5 h and in all 1980 files (99%) were received in 38 h.

The results of both batches were analysed to produce information about Level-1 tracking-trigger objects [12]:

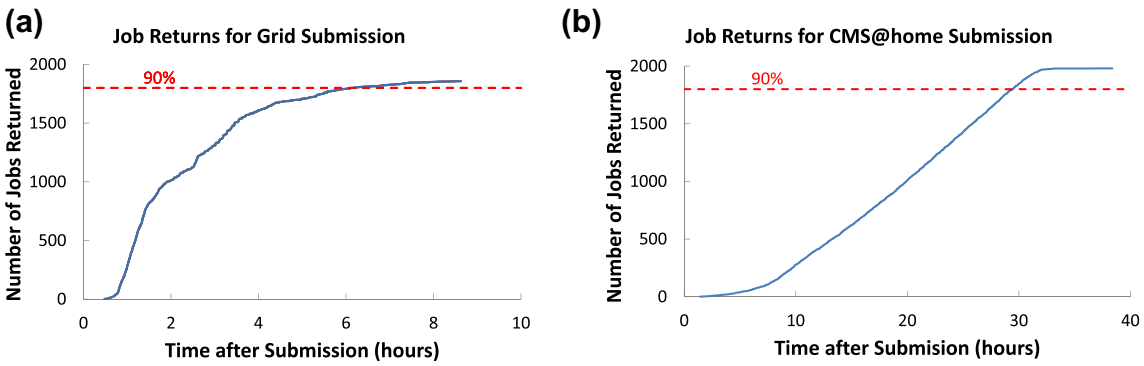

Fig. 2 The distribution of result files received for 2000 25-event ttbar simulation jobs, as a function of time from submission: **a** results from the normal grid; **b** results from ~100 CMS@home volunteers

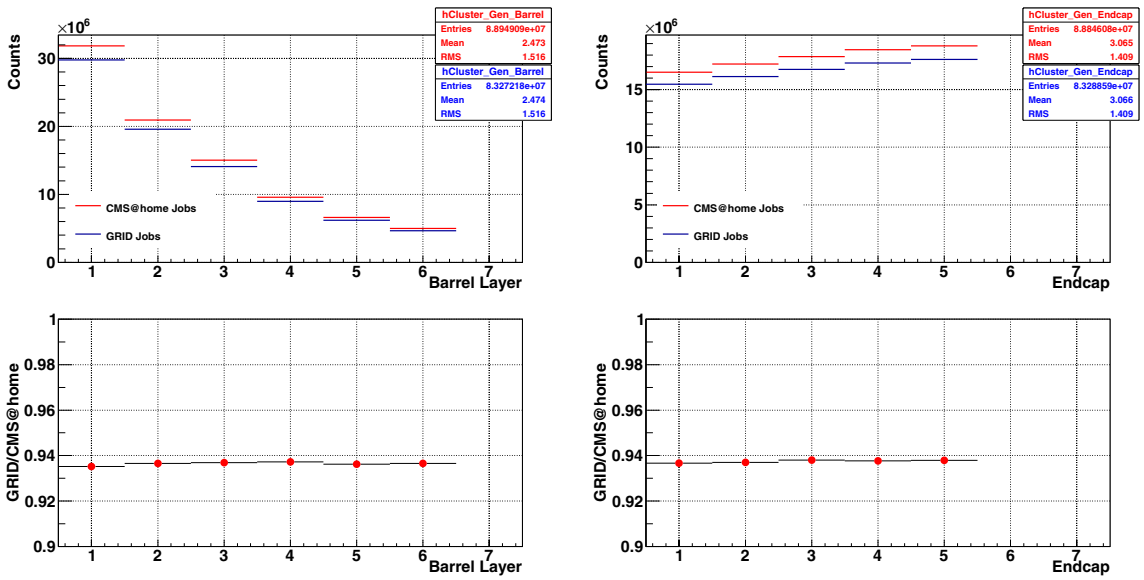

Fig. 3 The total number of simulated clusters in the CMS HL-LHC tracker from successful CMS@home (red) and grid (blue) ttbar jobs (1980 and 1858, respectively) in 2000-job batches: left, by barrel layer; right, by endcap layer; below, ratio of grid to CMS@home results

clusters (pixel or strip signals in individual detectors gathered into possible particle hits), and stubs (pairs of clusters from two closely adjacent detectors, possibly produced in the same particle track). Figures 3, 4 and 5 compare results from the two different processing streams. From the plots it can be seen that the ratio of results closely mirrors the ratio of successful jobs (1858/1980 = 0.938). However, there is a slight increase in this ratio moving out through the endcaps—also seen as a small trend at higher pseodorapidity in Fig. 5. This may be due to the unusually high failure rate of the grid jobs—many grid sites put strict limits on the running time and memory usage of jobs, while there are no such limits *per se* for CMS@home jobs. Individual events may exceed the site limit, particularly memory limits for events with high occupancy rates, removing those events from the results and potentially skewing the hit distributions.

At this proof-of-concept stage, it was considered that there was no reason to suspect any physics difference between results from normal grid jobs and those from CMS@home, although this will need to be revisited when the project moves to production simulations.

As the project matured, a new workflow was sought that better matched the characteristics of the volunteers' computers, and especially their connectivity. One problem that had been encountered was when enthusiastic volunteers were committing six or eight, or even more, computers to the project, and the resultant upload burden often led to transmissions timing out due to insufficient bandwidth.

A workflow was found that both matched the project's capabilities and also was of scientific interest, the rare decay of a Λ_b^0 to a proton, a muon, and a neutrino. This process is a background in investigations of a B_s decaying to two muons,

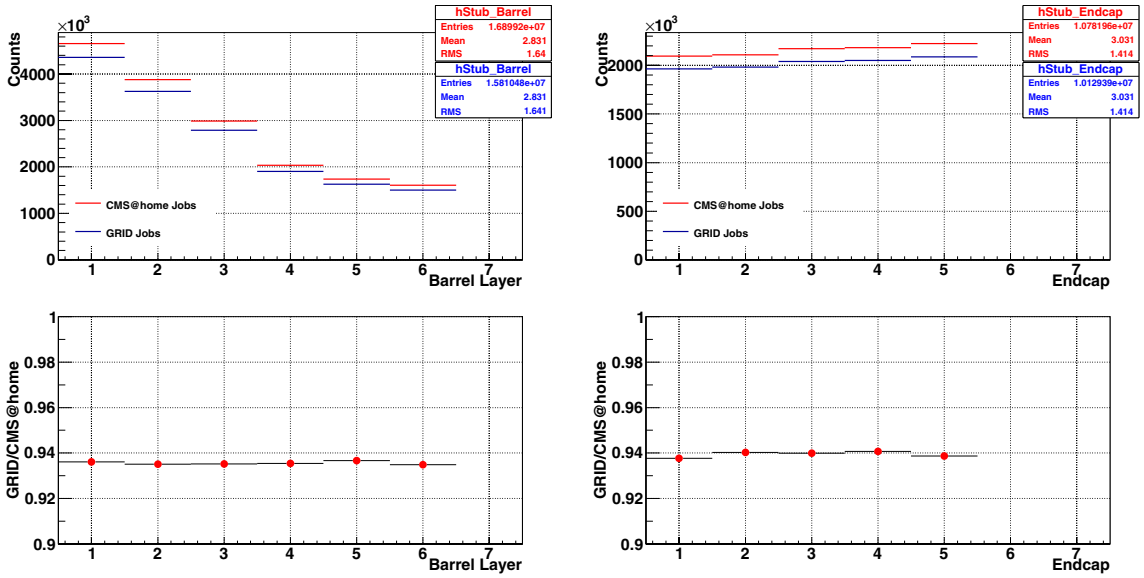

Fig. 4 The total number of simulated stubs in the CMS HL-LHC tracker from successful CMS@home (red) and grid (blue) ttbar jobs (1980 and 1858, respectively) in 2000-job batches: left, by barrel layer; right, by endcap layer; below, ratio of grid to CMS@home results

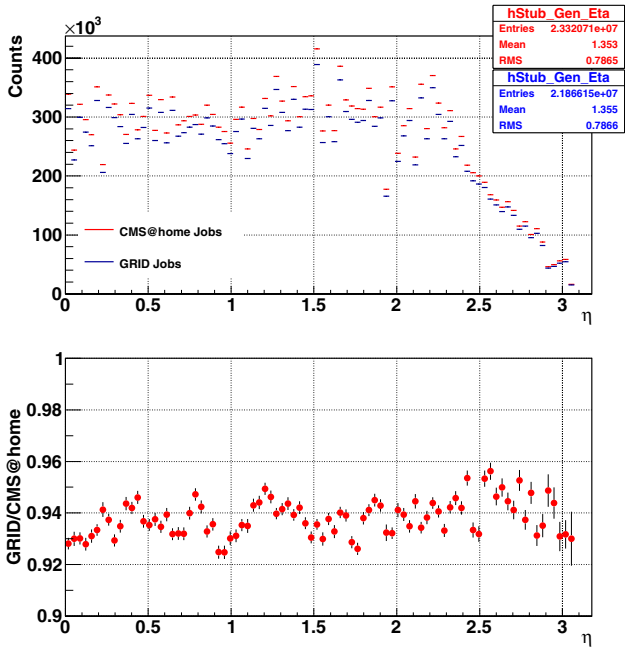

Fig. 5 The distribution of stubs as a function of pseudorapidity η from successful CMS@home (red) and grid (blue) ttbar jobs (1980 and 1858, respectively) in 2000-job batches; below, ratio of grid to CMS@home results

as the proton can be misidentified as a muon. Because the production ratio is small, around $3{\times}10^{-5}$, and many proton–proton collisions need to be simulated to provide a significant number of desired events, the request had been sidelined from official simulation production.

Jobs simulating 200,000 collisions were found to take a median time of 2 h 20m, and return around 16 MB (~6 events), so this workflow was submitted to the project and jobs successfully run on volunteers' machines. From the end of June 2016 to early September, tasks constituting around 22×10^9 collisions had been submitted and returned, with a success rate approaching 98% (see Table 1 for statistics of a representative batch of jobs). At this point, the production was nearing the requested production of 10^6 events, and it eventually considerably surpassed this number after CMS@home was made available as a subproject on the "production" project vLHCathome, opening it up to a wider range of volunteers. As well, the restriction of one job per host was lifted to allow volunteers to run as many jobs as their processors, memory and connectivity would support. By December 2016, when job submission was moved to WMAgent rather than CRAB3, the project had reached occasional peaks of over 800 jobs being run simultaneously, with over 250 results being returned per hour.

An analysis of one batch of 10,000 jobs found 8822 result files (lower than typical) containing 58,196 $\Lambda_b^0 \rightarrow p + \mu + \nu$ events, a ratio of $58196/(8822 * 2 \times 10^5) = 3.3 \times 10^{-5}$, in line with expectations from the Monte Carlo input parameters. Figure 6 shows the distribution of the sum of the reconstructed mass of the protons and muons in these events, showing the overlap with the B_s mass of 5.37 GeV/c^2.

Table 1 Statistics from a representative batch of 10,000 jobs, each of 200,000 proton–proton collisions, searching for $\Lambda_b^0 \to p + \mu + \nu$ events, comparing final job states reported by the CMS Dashboard [13] with result files actually found on the Data Bridge. Dashboard occasionally misses change-of-state reports; here, jobs reported as "unknown" and "postprocess" (the clean-up phase after the job finishes) have apparently actually succeeded, while results from 242 CMS@home jobs never sucessfully transferred to the Data Bridge

Number	Job states reported by Dashboard				Data bridge files	
of jobs	Success	Failure	Unknown	Postprocess	Good	Failed
10,000	9731	249	5	15	9751	7

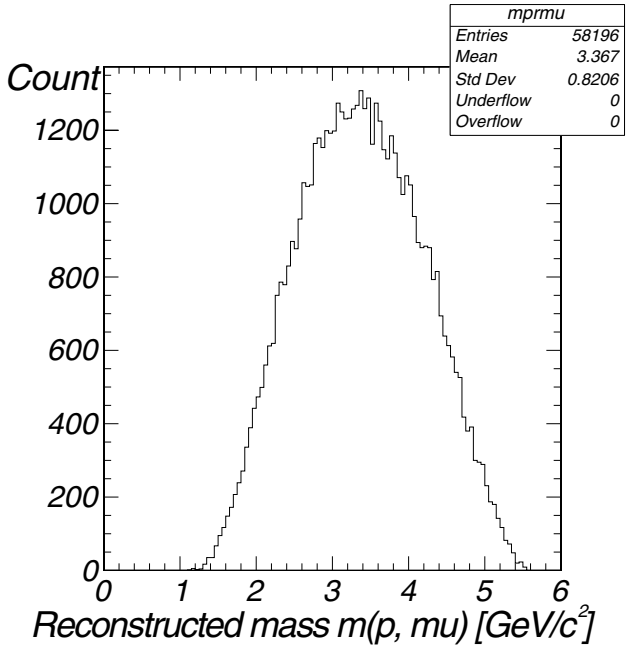

Fig. 6 The distribution of the reconstructed mass of the protons and muons in 58,196 $\Lambda_b^0 \to p + \mu + \nu$ decays from 1.76×10^9 proton–proton collisions, as simulated by CMS@home volunteers. Since the Λ_b^0 has a higher mass (5.62 GeV/c^2) than B_s (5.37 GeV/c^2), the reconstructed mass of the $p + \mu$ overlaps the B_s mass, due to the undetectable ν carrying away a variable amount of energy

Conclusions

By building upon the vacuum model and HTCondor, a volunteer computing platform has been provided that integrates resources managed by BOINC into the computing infrastructure of the CMS experiment. A key feature of this approach is that it is experiment agnostic, i.e. completely independent of the workload management system, and hence may be suitable for other experiments. To realize this, a few supporting services were developed including the Volunteer Computing Credential Service and the Data Bridge. Together they form computing infrastructure for the CMS@home project, which enables the experiment to harness the opportunistic capacity delivered by volunteer computing. This has been used to run real CMS workflows and the results have been compared to results from identical workflows that we submitted to the grid. Because the comparisons showed no obvious defects in the results returned from CMS@home, we conclude that there is no reason not to expand the project to run other scientifically interesting workflows. Indeed, we are in the process of doing this and CMS@home is becoming a significant CMS data-producing resource. Hence, we have demonstrated that CMS@home can provide additional computing capacity for the CMS experiment and that this approach could also easily be adopted by other experiments.

Acknowledgements We would like to thank all the volunteers who have provided computer time to this project, and also assisted in finding and fixing problems associated with its development. We also thank Urs Langenegger for providing the plot of Λ_b^0 decays.

References

1. Field L, Borras H, Spiga D, Riahi H (2015) CMS@home: enabling volunteer computing usage for CMS. J Phys Conf Ser 664(2):022017
2. Anderson DP (2004) BOINC: a system for public-resource computing and storage. IEEE 4–10
3. Korpela E, Werthimer D, Anderson D, Cobb J, Leboisky M (2001) SETI@home-massively distributed computing for SETI. Comput Sci Eng 3(1):78–83
4. Herr W, Kaltchev D, McIntosh E, Schmidt F (2006) Large scale beam–beam simulations for the CERN LHC using distributed computing. C060626: 526–528, Edinburgh, UK
5. Lombraa Gonzle D, Grey F, Blomer J, Buncic P, Harutyunyan A, Marquina M, Segal B, Skands P, Karneyeu A (2012) Virtual machines & volunteer computing: experience from LHC@Home: Test4theory project. In: *PoS(ISGC 2012)036*, Taipei, Taiwan, March 2012. Proceedings of Science
6. Lombraa Gonzlez D, Charalampidis I, VolatileStorm, Nabet J. Cernvmwrapper. http://dx.doi.org/10.5281/zenodo.17508
7. McNab A, Stagni F, Ubeda M (2014) Garcia. Running Jobs in the Vacuum. J Phys Conf Ser 513(3):032065

8. Cinquilli M, Spiga D, Grandi C, Hernndez JM, Konstantinov P, Mascheroni M, Riahi H, Vaandering E (2012) CRAB3: establishing a new generation of services for distributed analysis at CMS. J Phys Conf Ser 396(3):032026
9. Evans D, Mason, D Gutsche O, Metson S, Wakefield S, Hufnagel D, Hassan A, Mohapatra A, Miller M, van Lingen F (2008) 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
10. Litzkow MJ, Livny M, Mutka MW (1988) Condor-a hunter of idle workstations. In: *[1988] Proceedings. The 8th international conference on distributed computing systems*, San Jose, CA, USA, pp 104–111
11. Sfiligoi I (2008) glideinWMSa generic pilot-based workload management system. J Phys Conf Ser 119(6):062044
12. Giacomo S (2016) Upgrades of the CMS outer tracker for HL-LHC. Accelerators, spectrometers, detectors and associated equipment, nuclear instruments and methods in physics research section A
13. Andreeva Julia, Boehm Max, Gaidioz Benjamin, Karavakis Edward, Kokoszkiewicz Lukasz, Lanciotti Elisa, Maier Gerhild, Ollivier William, Rocha Ricardo, Saiz Pablo, Sidorova Irina (2010) Experiment dashboard for monitoring computing activities of the lhc virtual organizations. J Grid Comput 8(2):323–339