The evolution of the CMS@Home project

Federica Fanzago[1*], *Laurence* Field[2], *Ivan* D.Reid[3], *Marco* Mascheroni[4], *Alan* Malta Rodrigues[5], *Daniele* Spiga[6] and *Christoph* Wissing[7] for the CMS collaboration

[1]INFN, Sezione di Padova,Padova, Italy
[2]CERN, Geneva, Switzerland
[3]Brunel University of London, UK
[4]University of California, San Diego, La Jolla, CA, USA
[5]University of Notre Dame, South Bend, IN, USA
[6]INFN, Sezione di Perugia, Perugia, Italy
[7]DESY, Hamburg, Germany

Abstract. Over time, the idea of exploiting voluntary computing resources as additional capacity for experiments at the LHC has given rise to individual initiatives such as the CMS@Home project. With a starting point of R&D prototypes and projects such as "jobs in the Vacuum" and SETI@Home, the experiments have tried integrating these resources into their data production frameworks transparently to the computing infrastructure. Many of these efforts were subsequently rolled into the umbrella LHC@Home project.The use of virtual machines instantiated on volunteer resources, with images created and managed by the experiment according to its needs, provided the opportunity to implement this integration, and virtualization enabled CMS code from a Linux environment to also run on Windows and Macintosh systems, realizing a distributed and heterogeneous computing environment.A prototype of CMS@Home integrated with the CMS workload management CRAB3 was proposed in 2015, demonstrating the possibility of using BOINC as "manager" of volunteer resources and adapting the "vacuum" concept with the HTCondor Glidein system to get CMS pilots and jobs to execute on volunteers' computers. Since then, the integration of volunteer machines with the CMS workload management WMAgent, the official service dedicated to data production, has been seriously considered. The characteristics of volunteer resources regarding bandwidth capacity, connection behavior, and CPU and RAM capacities make them suitable for low-priority workflows with low I/O demands.The poster describes how the configuration of volunteer resources has evolved to keep pace with the development of the CMS computing infrastructure, including using tokens for resource authentication, exploiting regular expressions to accept workflows, manual glideins to initiate pilots, and other implementation details to achieve successful workflows. Currently volunteers are able to execute task chains of multicore jobs and, despite their limitations, are contributing to CMS computing capacity with around 600 cores daily.

[1*] Corresponding author: federica.fanzago@pd.infn.it

1 Introduction

Over several years now, the idea of exploiting voluntary computing resources as additional capacity for experiments at the LHC has given rise to individual initiatives for several projects. Starting from R&D prototypes and projects such as "jobs in the vacuum" [1] and SETI@Home [2] the experiments have tried to integrate these distributed resources into their data production frameworks in a manner transparent to their computing infrastructure.

In the following we discuss how this has been done in the context of the Compact Muon Solenoid (CMS) experiment, and give some insights into how this has evolved over time in the CMS@Home program [3].

2 Volunteer computing: the idea

Volunteer computing is a well-established form of citizen science in which users download scientific software onto their own computer to run simulations or analyses, without significantly interfering with the machine's normal operation. Public participation in scientific research is becoming more and more crucial not only by enhancing the computing power available for projects but also by improving everyone's understanding of science and its benefits to society. People donate their computers' unused resources to a research-oriented project for free, sometimes receiving credit points in return.

Exploiting voluntary computing resources as additional capacity for experiments at the LHC has given rise to individual initiatives, such as the CMS@Home project subsequently rolled into the umbrella LHC@Home project [4].

3 The BOINC system

The Berkeley Open Infrastructure for Network Computing (BOINC) [5] is software developed by the University of California, Berkley that enables personal computers to put their unused CPU and GPU cycles to use for scientific projects that need more computational power. Developed to manage the SETI@Home project launched in 1999 with the aim of analyzing signals from the Arecibo Observatory's ratio telescope, using computing capacity from volunteers, it was then also adopted by projects in other scientific fields, including biology, medicine, climatology and physics. The number of such projects has increased over time.

BOINC is a client/server platform for distributed internet-connected computing designed to support 'high throughput computing', in which there are large numbers of independent compute-intensive jobs. Users download and install the BOINC client on their resources, selecting the operative system, and after the creation of an account on the BOINC web page, they can join one or more projects. The client runs on volunteer computers and the server, dedicated to a specific project, manages scheduling, tasks and users. The server offers various project applications that are executed on the computers of volunteers. Each project defines its own computing requirements and volunteers can customize the way they contribute by configuring the BOINC client through the BOINC manager, deciding when and how to make available their resources for use (fig.1).

To entice people to participate in the project, a "credit" system was implemented in BOINC. There is no payment involved, but a recognition in credits based on the donated CPU usage time. BOINC then makes a ranking of the most deserving users and declares the winner per project every week; several independent web-sites also monitor the statistics.

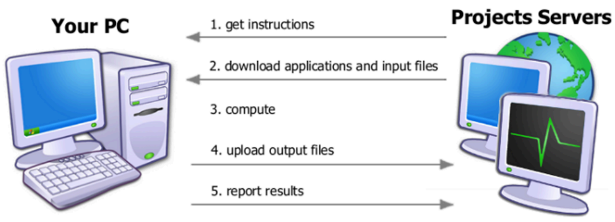

Fig. 1. Connection and info exchange between client and project server

4 LHC@Home

Launched as a BOINC project in 2004, LHC@Home initially focused on the accelerator physics numerical simulation code SixTrack, to help design and improve particle beam dynamics. Many volunteers were attracted to the project, demonstrating the potential of using this type of resources. Since 2011 it has been expanded to a wider range of LHC applications, thanks to the adoption of virtualization which allows software to run on heterogeneous resources without the need to recompile code for the different operating systems. In this way, code developed in Linux systems can also run inside virtual machines (VMs) on MacOS and Windows, widening the pool of potential volunteers. Linux VMs are launched using CernVM [6] as the base for application images, and Oracle VirtualBox as the local hypervisor. The VMs are managed by the BOINC client using the Vboxwrapper software module, maintained by the BOINC development team.

The aim of LHC@Home is to increase the computational power of experiments for simulations and upgrade studies by exploiting idle volunteer resources, thus involving the public in real science. While volunteer resources are provided for free, their integration into an experiment's framework requires some effort from both the experiment and expert teams, not only to create and maintain the application, but also to keep up with changes in the experiment framework and to provide support to volunteers. The same BOINC server, installed at Cern and under IT division responsibility, is used for all the @Home projects.

5 CMS@Home

CMS@Home uses volunteer computing to run simulations of the CMS experiment, and is one of the applications hosted by the LHC@home BOINC project. Born as a proof of concept project in 2015, starting from R&D prototypes and projects such as "jobs in the Vacuum" and SETI@Home, it demonstrated the feasibility of using volunteer resources to run CMS applications for data simulations and reconstruction via CMS Remote Analysis Builder v3 (CRAB3), a tool of the Workload Management group mainly dedicated to user analysis [7]. Over the past years the project has evolved to interface with the official tool for the central production of simulated and reprocessed datasets for CMS, the WMAgent, demonstrating the ability to follow and to adapt to changes of the CMS computing infrastructure, integrating volunteer resources transparently into its data production frameworks.

The CMS workflow management system is based on GlideinWMS [8], with HTCondor [9] managing all the computational resources as a single CMS global pool. The

GlideinWMS coordinates resource allocation in the distributed environment through "pilot" jobs, while HTCondor handles batch processing.

The current method of running jobs on trusted remote resources, such as grid or cloud nodes, uses the pilot approach (glideins), where the pilot job has the responsibility to check if a remote node is correctly configured to execute real jobs (payload). After suitability is confirmed, the pilot allows connection of the node to the HTCondor pool so that it can obtain jobs from the central CMS task queues. In the classical pilot-based pool the submission of pilots are triggered by the CMS Frontend service from the Factory upon detecting workloads in a submit node (schedd). However, volunteer resources are not trusted by the experiment, since anyone can sign up for the project and donate resources, so they can not be added to the pool in the same way. Running code over untrusted resources is a challenge for the experiment. Moreover, volunteer resources have some limitations that need to be considered when deciding which types of productions can be executed.

5.1 CMS@Home challenges

Volunteer resources are not trusted by the experiment, so they cannot be added directly to the CMS global pool. For this reason, a dedicated pool, the volunteer pool (fig.2), has been created. It includes a dedicated WMAgent, which also serves as the HTCondor schedd node for jobs, and the HTCondor central manager (collector and negotiator). The production WMAgents can flock jobs to the volunteer pool and the use of "regular expressions" matches only specific workflows to be executed on volunteer machines.

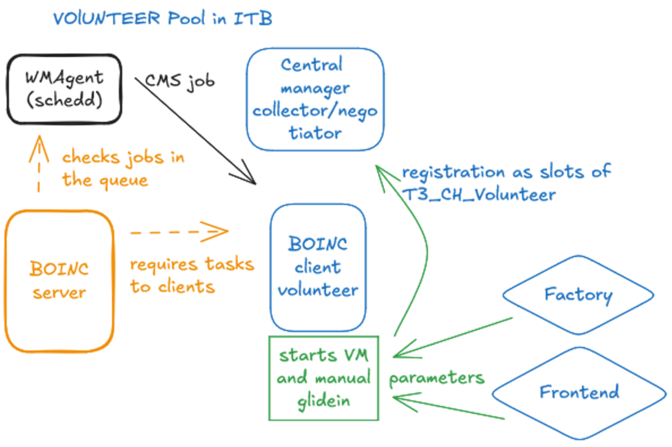
VOlUNTEER Pool in ITB
WMAgent (schedd)
CMS job
Central manager collector/negotiator
checks jobs in the queue
registration as slots of T3_CH_Volunteer
BOINC server
requires tasks to clients
BOINC client volunteer
Factory
starts VM and manual glidein
parameters
Frontend

Fig. 2. The volunteer pool.

The availability of volunteer resources, due to their nature, is not guaranteed; they can appear and disappear as users wish. For this reason, the resource provisioning isn't done through the standard glidein pilot-job approach, as described above, but by adopting the method of "glideins in a vacuum". The pilot starts running in the remote resource as soon as it becomes available and only if all the checks are passed is the node added to the volunteer pool. From the experiment's perspective, these resources appear spontaneously. Dedicated factory and frontend services are used only to download the configuration files required by the pilot.

For dataset production, there are certain limitations of volunteer resources that must be taken into consideration when deciding which workflows to execute on these resources. The potentially limited upload and download bandwidth allows only workflows that produce small outputs (~50 MB)) and do not require input data for execution. Moreover, a resource may disconnect at any time, suspending the execution of the job. As a result, the job will need time either to complete or to be reassigned to another resource. For this reason, the experiment's workflows should have low priority. Additionally, no storage is associated with volunteer machines, so only Monte Carlo task chains can be executed. The produced outputs must be held in a dedicated storage for "quarantine" before being injected into the production system. Finally, limitations in CPU and RAM must also be considered, so single and multi-core legacy workflows with 4 CPUS and 2GB of RAM are currently considered.

5.2 Setup of CMS@home for CMS dataset production

When a request of data production is injected in the system, WMagent prepares jobs that are then queued in the HTCondor schedd. The BOINC server checks the HTCondor queue and triggers the request of BOINC tasks to the available BOINC clients. When a client connects to the server and a task is available, it starts a VM using the dedicated CMS application image provided by the server.

In the bootstrap script of the VM, included in the image prepared by experts and distributed by the BOINC server, the manual glidein wrapper script is launched. It downloads from the factory and the frontend details about the CMS environment that have to be verified on the node, and the parameters for the setup of HTCondor; then it starts the pilot. If checks and setup are successful, the glidein starts the HCondor daemon connecting the resource as an execution node to the pool as slots of the T3_CH_Volunteer site so payloads can arrive from the WMAgent schedd and be executed. During the lifetime of the glidein, the resource can receive more payloads. The usage of the manual glidein was implemented in the last years as the evolution of the "instant glidein" [10] adopted at the beginning of the project. The main difference is in the HTCondor installation and its configuration on nodes. With the instant glidein, HTCondor was required to be already installed, configured and ready to be started in the VM image; instead, with the manual glidein, the HTCondor version and how to configure it is done by the pilot. With the manual glidein, configurations downloaded by factory and fronted can be modified by the experiment according to its needs.

In the HTCondor configuration of volunteers, the value MaxHibernateTime is set to allow disconnection of volunteer resources for some hours without losing the job that was running on them (currently it is set to 7200 sec). The glidein has a lifetime of 48 hours but if it stays running for 30 minutes without receiving a real job, it terminates and triggers the switch off of the VM.

Volunteer services are installed at CERN. The installation and updates of the WMAgent is managed by the CMS WMCore team. The factory, the fronted and the manual glide wrapper are provided and managed by the CMS SI team. The volunteer services are installed and configured like the production services and follow the same updates, although they are usually notified and applied later than in production since it is a low-priority project. Updates usually require service downtime, which is communicated to the volunteers. Sometimes, changes require the creation of a new image for volunteers, which must be tested in the CMS@home 'dev' environment before being released.

The CMS@Home project includes two sites. The first one is the already mentioned T3_CH_Volunteer site composed of the volunteer resources where the production step of the task chain is executed. No storage is associated with this site, outputs of this step are copied to the Data Bridge storage. The second site is T3_CH_CMSAtHome, where the "post-production" steps of the task chain, Merge, LogCollect and Cleanup, are executed. Its VMs are trusted and are under the CERN-IT division's responsibility; the Data Bridge is its associated storage. Outputs of these steps are written into EOS at CERN, the storage associated with the T2_CH_CERN site (fig.3). The effort required for services management and support is less than 0.5 FTE for both CMS and IT teams. It temporarily increases when a new image needs to be prepared, tested and distributed depending on the required changes.

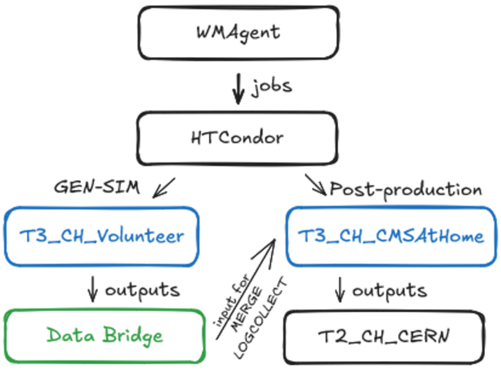

Fig. 3. Workflow's steps

5.3 BOINC tasks

The CMS@home workflow differs from the standard BOINC approach due to the use of glidein jobs. The real jobs are pulled into the volunteer resource by the glidein. The BOINC tasks are therefore pilot jobs that run the CMS glidein. The purpose of the BOINC task is to acquire the resource share, instantiate the VM and execute the CMS glidein. On the BOINC server a cron job checks the state of the CMS global pool and submits BOINC tasks if there are queued jobs. The assimilation and validation steps in a normal BOINC workflow are not needed. Credit is assigned based on the successful run of the glidein rather than the result of the CMS job. The validation and assimilation of the real data produced is handled by the CMS workflow.

5.4 Authentication and authorization

CMS production jobs run on resources using an official production proxy, but volunteer resources are not trusted so this proxy can't be shipped to them. The removal of the official proxy is done at the WMAgent level as a condition in the classad of the submitted job.

Registered users have a username and password and an associated id number; this is how BOINC authenticates users. In order to authenticate the volunteer VMs, the Volunteer Computing Credential Service (VCCS) was developed to generate a short-lived (7 days) X.509 proxy from the BOINC username and password. Thus a usable proxy is made available during the bootstrap of volunteer VMs allowing jobs to run on volunteer hosts and to copy results and logs onto dedicated storage, the Data Bridge.

In order for volunteer resources to connect to the HTcondor pool, they must be authorized. In past years the authorization was done through the VCCS proxy but currently it is based on an idtoken. The idtoken, centrally produced and periodically renewed, is provided during the bootstrap of the VM (fig.4).

5.5 The storage

The storage used by volunteers is the Data Bridge, a S3 Ceph storage system located at CERN under IT division responsibility, where volunteers can only write their produced outputs; deletion or overwriting of files is forbidden to volunteers.

Post-production steps of the task chain (merge, cleanup and LogCollect) running on T3_CH_CMSAtHome have an associated production proxy so they can remove files from Data Bridge. The URL redirector is provided by DynaFed. The redirector authenticates and authorizes connections to allow volunteers to write but not delete files. The storage provides an automatic cleanup policy to remove files older than 60 days.

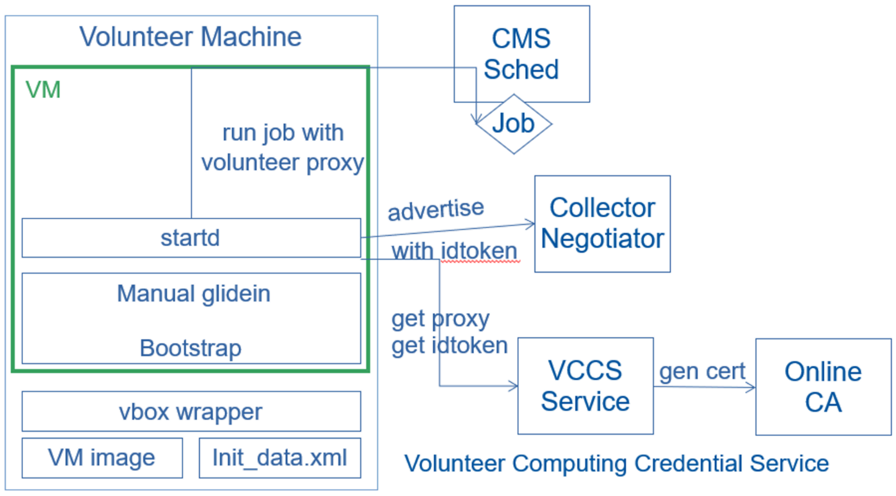

Fig. 4. Volunteer virtual machine initialization

6 Workflows

CMS@Home uses the dedicated WMAgent installed at CERN to inject jobs into the HTCondor pool. It can manage single- or multi-core jobs, but for simplicity the volunteers must adjust their VM preferences to match the workflow. The current setup allows the use of all the CPUs provided by a VM (whole node) and recognised by the pilot, limiting the total memory for VMs to 2GB. This memory value is configured at the factory level and can be changed depending on workflow requirements.

The number of running cores fluctuates as volunteers prioritise their commitments according to their needs; typical values range from 700 to 2,200 cores at any given time.

Most recently we have been submitting 4-core jobs, taking about 10 hours running time, simulating 120,000 proton-proton collisions events and filtering for decays $B0 \rightarrow D^{*\pm}D^{\mp}$. In less than a year this campaign has produced 550 GB of Monte Carlo results, representing 6.1 million filtered events from a total of 42 billion primary collisions.

As is to be expected, the jobs do not run without a certain proportion of errors, usually in the range of 5-10% failures. Many are transient problems but two cases of persistent errors occur: failure of the volunteer VM to access database machines holding the descriptions of the CMS detector's construction; and failure of the stage-out of result data to the Data Bridge. The first of these is characterised by multiple failures from the same volunteer, suggesting a local network misconfiguration, but the second does not present a consistent pattern and is currently not understood.

7 Conclusions

Over the years, the CMS@home project has demonstrated the possibility of integrating volunteer resources into the analysis and production framework of the CMS experiment. However, this integration requires significant effort to stay aligned with changes in the official infrastructure services and to provide daily support to volunteer resources by troubleshooting potential issues. CMS@home contributes around 400 daily volunteer resources, representing about 0.3% of production and about 1% of opportunistic CMS resources. While it is not critical from the resources perspective, it is useful to explore capabilities to offload payload onto opportunistic resources considering their intrinsic limitations. Moreover, the potential number of volunteers who could be immediately added to the project considering former participants of SETI@Home, which is no longer active, and the current ATLAS@Home participation is around 2-3k nodes, a value comparable to a small Tier-2 Grid site. For full production deployment, further improvements are still necessary to manage volunteer resources as an official CMS site. In particular, it is essential to optimize process monitoring to identify the causes of possible errors, simplify their management, and better structure technical support, which should collaborate with the support team for official productions

We would like to thank the Dynamic Resource Provisioning, the Submission Infrastructure and the Workload Management teams for their support, as well as all the volunteers for their precious feedback and contributions.

References

1. A.McNab et al., *Running Jobs in the Vacuum*, J. Phys.: Conf. Ser. **513** 032065 (2014)
2. SETI@Home, http://setiathome.ssl.berkeley.edu/
3. CMS@Home, https://lhcathome.web.cern.ch/projects/cms
4. D.Cameron et al., *All grown-up; 18 years of LHC@home,* EPJ Web Conf **295**, 04004 (2024)
5. D.Anderson et al., *BOINC:A Platform for Volunteer Computing.* J.Grid Computing **18**, 99–122 (2020)
6. P.Buncic et al., *CernVM – a virtual software appliance for LHC applications,* J.Phys.: Conf. Ser. **219** 042003 (2010)
7. L.Field et al., *CMS@home: Enabling Volunteer Computing Usage for CMS*, J.Phys.: Conf. Ser. **664** 022017 (2015)
8. GlideinWMS, https://glideinwms.fnal.gov/doc.prd/index.html
9. M.Litzkow et al., *Condor-a hunter of idle workstations,* Proceedings of the 8th International Conference of Distributed Computing Systems, 104-111 (1988)
10. L.Field et al. *The Instant Glidein; A generic approach for the late binding of jobs to various resource types,* J.Phys.: Conf. Ser. **898** 092009 (2017)