Journal of Physics: Conference Series **513** (2014) 032099 doi:10.1088/1742-6596/513/3/032099

Integration of Cloud resources in the LHCb Distributed Computing

Mario Úbeda García[1], Víctor Méndez Muñoz[2 3], Federico Stagni[1], Baptiste Cabarrou[1], Nathalie Rauschmayr[1], Philippe Charpentier[1], Joel Closier[1].

[1]LHCb, Department of Physics - European Organization for Nuclear Research (CERN), Genève, Switzerland.
[2]Port d'Informació Científica (PIC) - Universitat Autònoma de Barcelona, Bellaterra, Spain.
[3]Institut de Física d'Altes Energies (IFAE), Bellaterra, Spain.

E-mail: ubeda@cern.ch

Abstract. This contribution describes how Cloud resources have been integrated in the LHCb Distributed Computing. LHCb is using its specific Dirac extension (LHCbDirac) as an interware for its Distributed Computing. So far, it was seamlessly integrating Grid resources and Computer clusters. The cloud extension of DIRAC (VMDIRAC) allows the integration of Cloud computing infrastructures. It is able to interact with multiple types of infrastructures in commercial and institutional clouds, supported by multiple interfaces (Amazon EC2, OpenNebula, OpenStack and CloudStack) - instantiates, monitors and manages Virtual Machines running on this aggregation of Cloud resources. Moreover, specifications for institutional Cloud resources proposed by Worldwide LHC Computing Grid (WLCG), mainly by the High Energy Physics Unix Information Exchange (HEPiX) group, have been taken into account. Several initiatives and computing resource providers in the eScience environment have already deployed IaaS in production during 2013. Keeping this on mind, pros and cons of a cloud based infrasctructure have been studied in contrast with the current setup. As a result, this work addresses four different use cases which represent a major improvement on several levels of our infrastructure. We describe the solution implemented by LHCb for the contextualisation of the VMs based on the idea of Cloud Site. We report on operational experience of using in production several institutional Cloud resources that are thus becoming integral part of the LHCb Distributed Computing resources. Furthermore, we describe as well the gradual migration of our Service Infrastructure towards a fully distributed architecture following the Service as a Service (SaaS) model.

1. Introduction

Since a few years there is a well established trend in industry moving towards Cloud Computing, through commercial Cloud providers. In addition, there is a noticeable growing number of Cloud providers within the High Energy Physics (HEP) Community, known as institutional Clouds. The HEP Community has to deal with a myriad of implementations, particularities, exceptions, features, interfaces to access Cloud Computing resources, independently of their nature - institutional or private. For those readers familiar with Grid computing, this scenario may sound familiar: the Grid used by HEP Communities has the same problem as it represents an heterogeneous solution. This lack of uniformity regarding Cloud Computing resources represents

at this early stage of adoption its strongest and weakest point: it allows us to compare and choose. In an attempt to not making the same mistakes that were made in the past, we present in this paper the decisions LHCb took to be able to profit from these resources. This paper presents in section 2 the concept of "Cloud Site" followed by the architectural model in section 3. In section 4, we identify use cases for the Cloud Sites and present first results using them. For those interested on a more practical perspective, we devote section 5 to implementation details.

2. Cloud Computing and IaaS

Nowadays, Cloud Computing is a technology that is mature and has been adopted in several environments[7]. The main reasons for its success lie in the ability to cut costs, the reduction of operational complexity, IT headcount, a simple implementation, agility, device and location independence, maintenance, etc... One can find different service models among Cloud Computing providers: Software as a Service (SaaS), Platform as a Service (PaaS), Infrastructure as a Service (IaaS). We will focus on IaaS which is the most basic service and the one that offers the highest levels of customization. The IaaS model allows the deployment of virtual machines (VMs) - of which we as users are the administrators.

A priori it is a technology that represents a logical evolution in response to the growing computational needs and that is made possible thanks to the maturity in fields like virtualization and autonomic computing. However, it represents for us a set of major challenges to be resolved before being able to effectively use it. Out of them, the integration with existing systems will be for us a constant, nagging concern. In order to facilitate this integration, we introduce the concept of Cloud Site.

2.1. Cloud sites

The concept of Cloud Site is not entirely a new concept for us, and certainly not in other worlds, like telephony[11]. A site in Grid Computing represents an administrative division of the resources, where locality (proximity) is one of the factors taken into account for grouping them into "sites". Note that there are no restrictions in size: a "site" can represent from a fraction of a provider's resources to a complete set of resources from a country, for example. Moreover, these divisions, once made, seldom change.

In contrast, a Cloud Site is a virtual container of resources (VMs) where the geographical distance is not necessarily the metric for grouping them, but more the tasks performed on the VMs. We - as resources administrators - have the power to bind, modify and alter them such that they fit our needs. In other words, we become the system administrators of a number of Cloud Sites: more than one Cloud Site may use resources provided by the same IaaS provider, depending on our needs for deploying Cloud Sites with different purposes. At any time we can resize our Cloud Sites: change their shares, shrink their number of resources, etc.

We decided for a uniform approach, where all our VMs are going to be identical - without exceptions - based on CernVM [4] images and distribute our software using the associated file system CVMFS. All our software is released on CVMFS, which represents the nexus between our activities and the OS. In terms of maintainability this has two major consequences:

(i) a tremendous reduction on maintenance work for the Cloud Sites: a shorter learning curve for a particular system, experts can devote their time to improve the system rather than troubleshoot it.

(ii) there is a role shift as the VO becomes the administrator of the resources in contrast to the Grid Sites where the VO is a mere temporary tenant.

One of the positive aspects of having Cloud Sites lies in their spontaneity as well as their ubiquity. Issues like load peaks, regional blackouts, urgent OS patches, need of using deprecated

platforms for running legacy applications are now easily solved. Cloud Sites will serve their purpose, and as soon as they are done, they will be shut down. A Cloud Site represents an entity that provides all means for tracking security issues, monitor itself, allows easy upgrades and that scales well. Aspects like security and monitoring that were managed in background by the Grid Sites partially fall back into the VO administrators who will have to manage them centrally.

3. Architecture

In the previous section, we introduced the concept of Cloud Site, which will be the administrative virtual container in which virtual machines will be bound. It is worth noting the analogy to telephony, as we are approaching the problem in a similar way as for the signaling issue on phone calls. On one hand, the phone call "metadata" which will setup the call (signaling) follows one route and the call itself finds a direct route to its destination. We can see the signaling as the interaction with the IaaS (VM instantiation, termination,etc. . .) and the call as the actual use made of the VM.

We have to provide as a VO some additional services to those provided by the IaaS provider.

3.1. VO services

The VO has to provide 4 key components, namely: Configuration Service, Scheduler, Registry and Monitoring.

Configuration service : it contains static metadata information regarding the various IaaS Providers: authentication protocols, access tokens, endpoint location, contact details, etc. . .

Scheduler : this is the master piece of the complete solution. It deals with the implementation disparities at the IaaS Provider level. In order to minimize the impact on our system, we concentrate all the knowledge regarding the providers in the scheduler. The scheduler's task is to keep the number of VMs between certain thresholds, which can be defined as static (i.e. instantiate / terminate a fixed number of VMs at this Cloud Site) or elastic (i.e. instantiate/terminate VMs based for example on the demand). It also takes into account the agreed constraints on the VM lifetime: based on default limits defined by the resources provider as well as short notice on demand requests for shutdown.

Registry : we need to keep track of the VMs we have running. As we are managing a Hybrid Cloud model composed by multiple IaaS, it is unpractical having to check the Cloud provider's registries one by one. This Registry keeps records of the status of all the existing VMs, their location, functions, etc. . . (recalling the telephony model, one may compare it with the Call Detail Record[8]).

Monitoring : so far, we have described the infrastructure that allows us to start and stop resources on the Cloud Sites, but we have not described how to know what happens within these Virtual Machines (besides their status reported by the IaaS in the Registry). Information like available local disk, instant load, swap, file descriptors opened, network, etc. . . is what we will need to know for troubleshooting a Cloud Site. We need also to pay attention to the monitoring of the particular applications that the VM is executing (i.e. if it is running physics applications, we want to know how many, if it is running a MySQL database, we want to know if it has slow queries, etc. . .).

3.1.1. IaaS provider services On the provider side we can find the usual components of any IaaS: Image store, Registry, Metadata server, Web interface and API, etc.

Image store : database where we will upload our CernVM images. This is a manual operation. Ideally, we would have a common store for all our Cloud providers.

Registry : stores the VM records that belong to this particular provider amd is used in order to update the VO global registry.

Metadata server : while performing the contextualization step (see below) this server dispatches the dynamic content specified by the scheduler (function of the VM, global VM id, etc) to the scripts that will setup the applications on the VM.

Web interface : for LHCb operational purposes is irrelevant.

API : this is what the scheduler will use to interact with the IaaS (for example EC2, nova, occi, etc. . .).

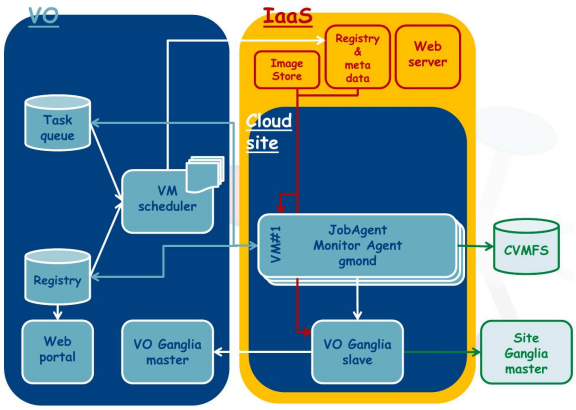

Figure 1. Full architecture with a Cloud site.

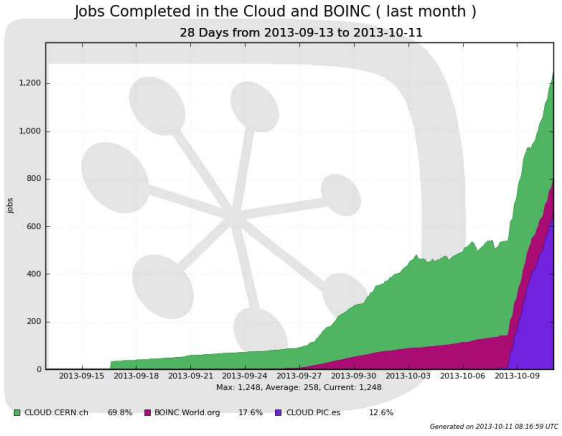

Figure 2. Cloud and BOINC completed jobs (last month).

4. Use cases

For VOs, the main use case for Cloud Computing is of course the integration of Cloud resources within their Distributed Computing resources, where Cloud Sites run VMs acting as Worker Nodes. Within LHCb, we decided to profit from the experience with Cloud Computing and to apply it to other use cases such as the deployment of our Distributed Computing infrastructure, or our testing services.

4.1. Distributed Computing

4.1.1. Pilot-VMs This is the main usage of Cloud Sites: be able to run physics applications anywhere and everywhere. In other words, this requires integrating the Cloud Sites as part of our Distributed Computing resources. LHCbDIRAC is using the "pilot job" paradigm in which a standard piece of software (called pilot) is deployed as an overlay on computing resources. The pilot is then in charge of getting some task (a.k.a job) to be executed on behalf of the VO on the overlaid resource.

First, we have to improve our pilot deployment model, where the VO sends pilots to the Grid sites. We started on our HLT Farm [3], were the PVSS system allows to simply instantiate pilots on the farm nodes. We followed the same approach to run jobs in the Clouds (one may say the VMs are the pilots. . .). Along these lines, there is a third alternative called vac(vacuum) in which the pilot-VMs are instatiated by the site and are not managed by the VO.

The current LHCb Cloud setup is hosted in three Cloud Sites hosted at two different providers: CERN (OpenStack) and PIC (OpenNebula). A third provider, RAL(StratusLab) is already in a testing phase.

Table 1. Comparison among some of our Distributed computing resources for running jobs: T0, T1s, CLOUDs and HLT Farm.

(DIRAC) Site Name	Count	Percentage
DIRAC.ONLINE.ch	14714	49.6%
LCG.CNAF.it	3082	8.6%
LCG.GRIDKA.de	2527	7%
LCG.CERN.ch	2317	6.4%
LCG.RAL.co.uk	1785	5%
LCG.IN2P3.fr	1651	4.6%
LCG.NIKHEF.nl	533	1.4%
LCG.PIC.es	406	1.1%
LCG.SARA.nl	104	0.3%
CLOUD.PIC.es	75	0.2%
CLOUD.CERN.ch	26	0.07%

4.1.2. Production Worker Nodes Two Cloud Sites are dedicated to LHCb production activities, and they are used fore running physics simulation applications.

In a recent snapshop of LHCb jobs presented on Table 1, LHCb was running concurrently 35686 jobs across more than 100 Distributed Computing sites. The table shows the number of jobs running at each of our T0 and T1 sites as well as on our HTL Farm (DIRAC.ONLINE.ch) and the production Cloud Sites. This latter set of resources represents about 3% of the total LHCb computing resources running at that time.

4.1.3. Playground platform We dedicate another Cloud Site at the CERN IaaS to Research and Development activities: some developments require particular applications on the worker node in order to evaluate new features of the physics software or for example a kernel compiled with different flags. Two developments have already benefited from this ad-hoc Cloud Site are x32-ABI[9] binaries and KSM[10] for reducing the memory footprint. Using the GaudiMP extension for the Gaudi framework on which our physics software is based, we have successfully run multicore jobs on that site as well and studied the gain of multicore processing on memory footprint.

4.1.4. "One pilot to fly in all skies" The LHCb plan for running jobs on a variety of Distributed Computing resources can be referred to as "One pilot to fly in all skies". The idea is to use the same overlay mechanism (pilot job) not only Grid and Cloud resources, but also on the High Level Trigger Farm (HLT) as well as Volunteer Computing resources, vac resources, etc... This strategy is based on a simplification of our current distribution model for LHCbDirac, creating identical WNs everywhere. It is achieved modifying the pilots such that we can independently use a push(Grid) or pull model(HLT, Clouds, Volunteer Computing, vac,...) as pilot factory using the same code base. A great spinoff of using this model is the possibility to integrate Volunteer Computing on personal computers, university laboratories during summer periods to the LHCb infrastructure.

4.2. Infrastructure deployment

4.2.1. Continuous Integration testing Running a VO Grid infrastructure represents a major challenge due to the large number of services, agents and databases that are required. In

addition, the physics applications must run under different SW configurations (architecture, Operating System, compilers, version of software,...). LHCb is hosting a master Jenkins server and a set of physical and now also virtual machines that are used to certify the LHCb Grid middleware (LHCbDIRAC [2] and DIRAC [1]) and the physics application stack under a matrix of different OS and python versions, namely: SLC5, SLC6, CernVM 2.6 and Python2.6 and Python2.7. This master Jenkins server and the physical slaves host part of the nightlies infrastructure used for the LHCb core software.

4.2.2. LHCbDIRAC services and agents At the moment, the LHCb Grid infrastructure is hosted by a dozen physical machines located at CERN as well as one machine on each T1 to ensure availability of the services with high requirements on availability. This represents a rather static configuration that meets its purpose. However, it is not flexible when need arises to add services or agents for scalability, and it cannot deal well with site outages like a network intervention at CERN. This represents a major issue when the complete system goes down, jobs keep running on the Grid and need to be followed up. As a consequence, heavy recovery campaigns must be launched to tidy up the leftovers after such an outage. Moving to another major version of the middleware requires a new parallel installation as draining the system prior its full stop may take weeks. As an alternative, deploying our services and agents over Virtual Machines offers plenty of advantages. Currently, if one of our 12 core machines breaks down, about 8.5% of our services will be affected, which potentially can translate into a long recovery time through a sophisticated failover mechanism. The solution lies in service redundancy in order to ensure a constant availability of the services. Using a Cloud Site as a basis for the infrastructure also allows a faster response to scalability issues or outages.

5. Implementation

The LHCb Distributed Computing infrastructure is based on DIRAC and its set of extensions (for example LHCbDIRAC). Among them, VMDIRAC provides the support for a palette of Cloud interfaces[6]. VMDIRAC is the Cloud Site scheduler[5] whose functionality is described in section 3.1. For running our infrastructure components, we can partition them as required on different VMs in order to decouple the physical implementation from the functionality. One of the advantages to use the same mechanism for running our infrastructure site is that it is not restricted to running at CERN. It makes the installation of additional components at other sites much easier. We have based our Cloud Sites on CernVM images using CVMFS as distributed file system. They are configured using the *amiconfig* plugin. We feed the *amiconfig* plugin with a userdata script that configures CVMFS, sets the access rights, writes a HEPiX epilog file that will be run at the end of the boot sequence of the VM. This epilog file sets up the environment for our software from CVMFS and runs the necessary services (specified in the epilog). These components can either be LHCbDirac agents or services in the case of the infrastructure site, or a job agent in case of a worker node site.

For physics simulation production, we currently use small virtual machines: one single CPU core with 2GB of RAM and 10GB of disk is sufficient. However, if we want to run jobs in multicore mode, we can easily increase the number of cores, memory or local disk space.

6. Conclusions

Within the LHCb collaboration there are a set of independent research lines that are slowly converging towards a full integration made possible thanks to Cloud Computing. Cloud Computing can be used for hosting VO services as well as for running production tasks or providing platforms for specific R&D. Either evolution or revolution, Cloud Computing represents a major change in the approach we followed so far administering our resources. Its strongest allies are: simplicity and spontaneity. One interesting aspect is that many of the

developments needed in terms of unification for the Cloud Sites are tremendously effective when applied to the Grid sites (e.g. sourcing the code to run the job from the CVMFS distributed file system instead of making a local software installation). Within few months it has provided us with more than 600 WNs distributed among different providers as well as the resources obtained through volunteer computing with practically no dedicated effort in maintenance.

7. Aknowledgements

The Port d'Informació Científica (PIC) is maintained through a collaboration between the Generalitat de Catalunya, CIEMAT, IFAE and the Universitat Autònoma de Barcelona. This work was supported in part by grants FPA2007-66152-C02-00 and FPA2010-21816-C02-00 from the Ministerio de Ciencia e Innovación, Spain. Additional support was provided by the EU 7th Framework Programme INFRA-2007-1.2.3: e-Science Grid infrastructures Grant Agreement Number 222667, Enabling Grids for e-Science (EGEE) project and INFRA-2010-1.2.1: Distributed computing infrastructure Contract Number RI-261323 (EGI-InSPIRE).

The authours would like to thank Andrew Mcnab whose developments for vac provided us with experience dealing with bottlenecks while running physics applications based on CVMFS. Also, the IaaS provider teams at CERN and PIC for their support, as well as the Test4Theory group at CERN for their guidance while working on the different aspects of the Volunteer Computing project. Lastly, but not less important, we would like to thank the LHCb collaboration for being our guinea pigs testing the Volunteer Computing infrastructure.

References

[1] A Tsaregorodtsev et al. Status of the dirac project. *Journal of Physics: Conference Series*, 396(3):032107, 2012.

[2] F Stagni et al. Lhcbdirac: distributed computing in lhcb. *Journal of Physics: Conference Series*, 396(3):032104, 2012.

[3] L G Cardoso et al. Offline processing in the online computer farm. *Journal of Physics: Conference Series*, 396(3):032052, 2012.

[4] P Buncic et al. Cernvm a virtual software appliance for lhc applications. *Journal of Physics: Conference Series*, 219(4):042003, 2010.

[5] Víctor Méndez et al. Rafhyc: An architecture for constructing resilient services on federated hybrid clouds. *Journal of Grid Computing*, 2013.

[6] Vctor Mndez Muoz et al. The integration of cloudstack and occi/opennebula with dirac. *Journal of Physics: Conference Series*, 396(3):032075, 2012.

[7] Maryline Lengert and Bob Jones. Strategic Plan for a Scientific Cloud Computing infrastructure for Europe. Technical Report CERN-OPEN-2011-036. CERN-OPEN-2011-036, CERN, ESA, Geneva,S Frascati, Aug 2011.

[8] A. Ofrane and L. Harte. *Introduction to Telecom Billing: Usage Events, Call Detail Records, and Bill Cycles*. Althos Publishing, 2004.

[9] N. Rauschmayr and A. Streit. Evaluation of x32-ABI in the Context of LHC Applications. Jun 2013. Procedia Computer Science: Volume 18, 2013, Pages 2233–2240.

[10] N. Rauschmayr and A. Streit. Reducing the memory footprint of parallel applications with ksm. In Rainer Keller, David Kramer, and Jan-Philipp Weiss, editors, *Facing the Multicore-Challenge III*, volume 7686 of *Lecture Notes in Computer Science*, pages 48–59. Springer Berlin Heidelberg, 2013.

[11] Henning Schulzrinne and Jonathan Rosenberg. Signaling for internet telephony. In *in International Conference on Network Protocols (ICNP*, 1998.