A scalable infrastructure for CMS data analysis based on OpenStack Cloud and Gluster file system

2014 J. Phys.: Conf. Ser. 513 062047

(http://iopscience.iop.org/1742-6596/513/6/062047)

Journal of Physics: Conference Series **513** (2014) 062047 doi:10.1088/1742-6596/513/6/062047

A scalable infrastructure for CMS data analysis based on OpenStack Cloud and Gluster file system

S. Toor[1], L. Osmani[2], P. Eerola[1], O. Kraemer[1], T. Lindén[1], S. Tarkoma[2] and J. White[3]

[1] Helsinki Institute of Physics (HIP), CMS Program, Helsinki Finland.
[2] Computer Science Department, University of Helsinki, Helsinki Finland.
[3] Helsinki Institute of Physics (HIP), Technology Program, Helsinki Finland.

E-mail: {Salman.Toor, Paula.Eerola, Carl.Kraemer, Tomas.Linden}@helsinki.fi, {Lirim.Osmani, Sasu.Tarkoma}@cs.helsinki.fi, John.White@cern.ch

Abstract. The challenge of providing a resilient and scalable computational and data management solution for massive scale research environments requires continuous exploration of new technologies and techniques. In this project the aim has been to design a scalable and resilient infrastructure for CERN HEP data analysis. The infrastructure is based on OpenStack components for structuring a private Cloud with the Gluster File System. We integrate the state-of-the-art Cloud technologies with the traditional Grid middleware infrastructure. Our test results show that the adopted approach provides a scalable and resilient solution for managing resources without compromising on performance and high availability.

1. Introduction

This paper describes the state and current experience with the virtualized computing environment for CMS data analysis. In order to fulfill this task, this cluster is Cloud-based, Grid-enabled and uses the Gluster [6] file system.

The cluster components are described including the OpenStack Cloud suite and Gluster file system. The Grid software that controls the execution of the physics jobs are components of the Advanced Resource Connector (ARC) [9]. This allows the end-users to submit the jobs with their preferred Grid or cloud submission system and at the same time provides flexibility to maintain the infrastructure. The analysis software and libraries are installed via the CERN Virtual Machine file system, a common component currently on most Grid sites. Cluster performance is measured by running Site Availability Monitoring (SAM) jobs as well as CMS Monte Carlo simulation jobs and CMS physics analysis jobs.

Several test cases have been defined in order to measure:

- Overall system performance;
- Gluster file system I/O performance;
- Performance of live migrations;
- Overall system stability.

These results are presented in the last chapter.

2. System Components

2.1. OpenStack Cloud

OpenStack [4] is a collection of many open source projects: OpenStack Compute (Nova), OpenStack Object Storage (Swift), OpenStack Image Service (Glance), Identity service (Keystone), OpenStack Networking (Quantum) and Dashboard (Horizon). Nova focuses on the compute layer with a set of tools for overseeing virtual instances and managing the virtual pool of compute resources. Swift, the storage component, enables the creation of massively scalable storage clusters with redundancy and failover for storing and retrieval of objects. Glance, the Image Service, is a lookup and retrieval system for virtual images. Quantum provides network orchestration with options of creating networks, subnets, assigning IPs and even plug or unplug ports. The design and architecture of OpenStack are heavily influenced by the emerging concepts of Software Defined Networks (SDN) thus relying heavily on backend plugins translating network abstractions into physical actions. Keystone provides a unified authentication across all OpenStack components and integrates also with other existing authentication systems such as LDAP. Horizon serves as a web portal to facilitate the control of Cloud resources. Although released under the same umbrella, the projects are developed independently.

2.2. Gluster FileSystem

The Gluster file system (GlusterFS) is an open-source solution to the needs for a low cost, highly scalable distributed file system to meet the storage requirements of a range of environments. The configuration changes can be introduced while filesystem is online making it very flexible and responsive to workloads or unpredictable events. As far as GlusterFS is concerned everything is a volume and these volumes are combined together to create a particular file system configuration. A native NFS stack, NFSv3 compliant, is built into the server enabling access to the storage via clients running the GlusterFS native protocol or the old NFS standard. A GlusterFS installation can be logically decomposed into bricks and translators. A brick is a network attached server with a local filesystem (ext3, ext4) which is then aggregated into a single storage pool or namespace with other bricks. Translators are option modules that connect to volumes, and export high level functionalities.

2.3. Advanced Resource Connector

Advanced Resource Connector (ARC) is a set of open source Grid computing middleware components, distributed under the Apache License, introduced by NorduGrid [12]. It provides a common interface for submission of computational tasks to heterogeneous distributed computing systems and thus can enable Grid infrastructures of varying size and complexity. ARC includes data staging and caching functionality, developed in order to support data-intensive Grid computing.

The ARC components used in this test cluster are:

- A-REX, the ARC job execution service. The test and production CMS data analysis jobs are submitted to A-REX (with Condor back-end).
- JURA, job record publishing service for A-REX. This accounting service is forwarding job details to the SweGrid Accounting System (SGAS) used by NDGF [3] and CSC [1].
- GridFTP server for data staging.

2.4. CERN Virtual Machine file system

The CernVM File System [8] (CernVM-FS) provides a scalable software distribution service. It was developed to assist HEP collaborations to deploy their software on VMs and on the WLCG sites. CernVM-FS is deployed on a wide range of computing resources, from powerful worker nodes at Tier 1 Grid sites to virtual appliances running on volunteer computers. It significantly

reduces the complexity with respect to both required capabilities of the master storage as well as installation and maintenance.

Files and directories are hosted on standard web servers and mounted in the universal namespace /cvmfs. File data and meta-data are downloaded on demand and locally cached. CernVM-FS decouples the experiment software from virtual machine hard disk images and is used as a replacement of the shared software area at Grid sites.

CernVM-FS solves the scalability issues of network file systems such as AFS, NFS, or Lustre, which are traditionally used for shared software areas. By now, CernVM-FS is actively used by small and large high energy physics (HEP) collaborations; for some collaborations, such as ATLAS and LHCb, CernVM-FS is a mission critical component of the distributed computing infrastructure.

In this case, CernVM-FS is used in this project to deploy the CMS analysis software [10] to worker nodes in the cluster. The CernVM-FS file system is mounted by each worker node and the required version of the CMS analysis software is cached.

3. CMS Data Analysis

CMS Monte Carlo simulation and physics analysis jobs to the T2_FI_HIP [2] resources can be submitted directly from an ARC UI or from elsewhere in the WLCG [7] through the EMI WMS or the glideinWMS [11].

Computing jobs in HEP data analysis and simulation are relatively simple and sequential. These jobs do require a well-defined and complex software environment. The base requirement for a CMS physics computing job in a Cloud environment is to run the correct Virtual Machine (VM). The VM should contain the correct runtime libraries and be enabled to install the correct analysis software. The method used here is CernVM-FS as described in Section 2.4.

The CMS reconstruction and analysis code are organized as independent modules that are driven by the CMS framework. A job configuration script defines the modules to be loaded (as plugins), configures them with the provided parameter sets, and steers the module execution sequences according to specified paths. It is the combination of a job configuration script and the installed software set (via CernVM-FS) that defines a CMS physics job that is sent to a Grid (in this case Cloud) worker node.

4. System Architecture

Nodes (WN) are running on VM instances inside the OpenStack Cloud. Currently we consider our approach as semi-static, as the instance management is manual. In the near future we are aiming for a comprehensive elastic solution by including the EMI authorization service (Argus) and the Execution Environment Service (Argus-EES).

The infrastructure solution is deployed on the University of Helsinki Computer Science Department HPC "Ukko" cluster. The resources are homogeneous: Dell PowerEdge M610 servers with two quad core Intel Xeon E5540 processors, 32GB (8 x 4GB) 1066 MHz DDR3 ECC of RAM, 4 x 10GbE Broadcom 57718 network interfaces and 80GB SATA 7200rpm HDD. OpenStack Grizzly release is installed on top of Ubuntu 12.04 LTS with the controller and the compute nodes. Figure 1 and Table 1 illustrate the overall system architecture and number of available resources.

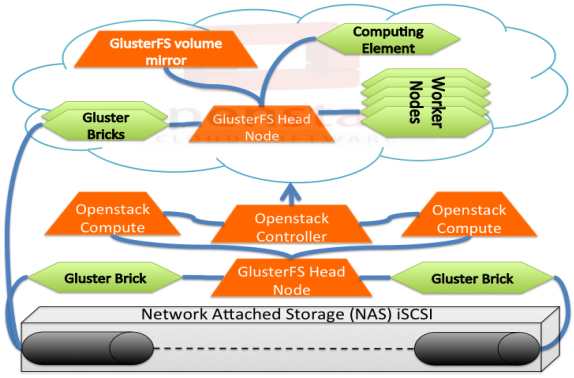

Figure 1: Overall system architecture based on OpenStack and GlusterFS

Type	Nodes	Cores	Memory(GB)
Virtual machines			
WN	25	4	14
CE	1	4	14
Real machines			
Controller	1	8	32
Compute	19	8	32
GlusterFS	6	8	32
Local and GlusterFS based storage			
	/ Local	/External Ephemeral	/hip-dii-Grid Shared MP
WN	10GB	45GB	900GB
CE	10GB	-	900GB

Table 1: Available resources both inside and outside the Cloud

The storage system is deployed on 6 Dell PowerEdge M610 servers. Each of the servers has a 512 GB LUN attached from a storage array over the "Fibre Channel over Ethernet" protocol. Two servers are configured into a shared storage backend with a total of 1 TB for the VMs to boot into, and the remaining 4 servers are configured as volume servers with a total 2 TB required for shared storage for the worker nodes. In order to accommodate the requirements for a high-speed ethernet infrastructure and redundancy in case of a link failure, the interconnect between the storage array and the servers is run with separate VLANs, switches and dedicated network interfaces on each of the servers. Administering Cloud resources and keeping the environment operating at maximum efficiency requires monitoring the physical infrastructure as well as resources they share.

The choice of OpenStack and GlusterFS enables us to create an efficient and scalable infrastructure. Figure 1 and Table 1 together provide a overall picture of the system architecture and the available resources. In Figure 1, orange boxes represent the components that are used to build the setup whereas the green sections represent the actual resources in the system. The selection of software components such as those from ARC and CernVM-FS make the site seamless to end users familiar with Grid systems and also eases the interface to Grid accounting and monitoring systems. Up to 25 Worker Nodes (WN) and 1 Computing Element(CE) are activated as OpenStack VMs. Each one is running Scientific Linux release 6.3. The test setup can provide up to 100 slots for serial jobs and 25 slots for whole node requests.

We have used A-REX from ARC as our meta scheduler and HTCondor for internal resource management. CernVM-FS is mounted on each WN in order to provide the software needed to run the CMS analysis jobs. This infrastructure appears as a normal ARC site in the WLCG, thus no extra work is required to integrate the site. The choice of ARC middleware allows to later accommodate multiple projects using different Run-Time-Environments.

For efficient utilization of resources, the image size of the VMs is kept minimal and we rely on the Gluster file system for storage. We have used QCOW2 format for managing VM image files. QCOW2 supports extended features like small file size, multiple snapshots and copy-to-write. As shown in the last part of the Table 1, the storage of each VM is divided into three parts:

- The local `/root`;
- The `/External`, Ephemeral disk attached to each Worker Node. This area contains the Condor job session directories and the cache area for CernVM-FS;
- The `/hip-dii`, a shared area based on GlusterFS amongst WN(s) and CE. This is used for the ARC based job session and cache directories.

With these settings the original size of each of the VM is confined to 10 GB and rest is managed dynamically with GlusterFS.

5. Test Cases

5.1. System Performance

For the first part of the test we were interested to find the system response and performance penalty due to the underlying virtualization. We used the HEPSPEC 2006 [5] benchmark in order to measure this. According to our results, a 4 core VM provides 12.815 HEPSPEC per core whereas a 8 core real machine without hyperthreading provides 14.11 HEPSPEC per core. The result shows that there is roughly 4 % performance loss incurred by employing virtualization.

The second part of this test was based on the instance response time. In our setup we used 1 TB of GlusterFS-based storage for the Glance repository. The penalty introduced by using GlusterFS instead of local storage was measured by looking at the instance booting time.

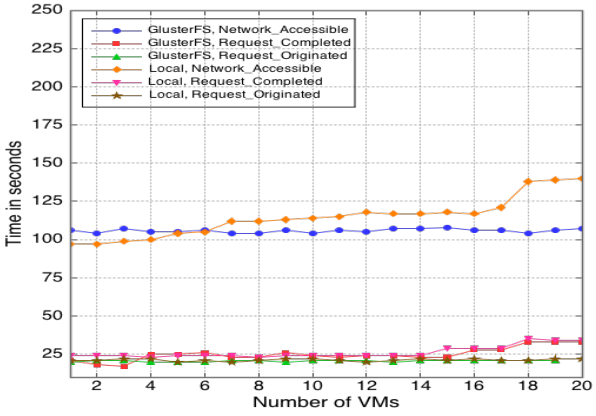

Figure 2: 20 instances

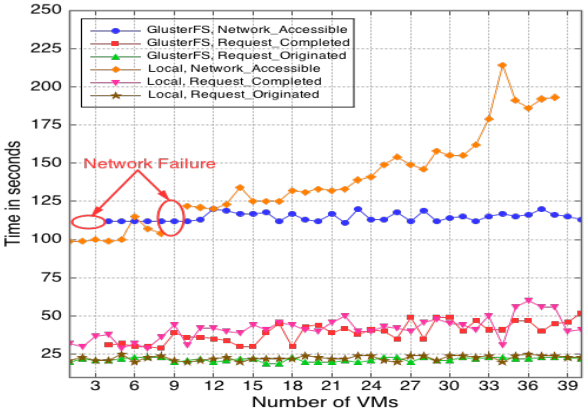

Figure 3: 40 instances

Sequential image booting did not show any problem, while concurrent VM booting worked perfectly up to 20 concurrent VMs. Ramping to 40 concurrent images, we noticed a $\approx$ 10% failure rate.

Figure 2 and 3 illustrate the instance response time while having Glance repository on the local and Gluster file system. We measured three parameters;

- Request Initiation, when the request is placed in queueing system(in our case we used RabbitMQ);
- Request Completion, the scheduler assigns a compute node to boot the VM;
- Network Accessibility, when the VM is available with the network.

The results show that with GlusterFS the worker node VMs took less than 2 minutes to be accessible with all the necessary configurations. The boot process of a VM requires that a separate copy of the image is available in the Glance repository. By having the same file system, the whole process is accelerated. This can be seen in the Figures 2 and 3, with GlusterFS the boot process was flat for the VMs. In contrast, overall time with the local disk started to increase with a higher number of VM requests.

The results presented in this section show that there is no significant penalty by using a virtualized environment and the choice of GlusterFS also gives an acceptable VM startup time.

5.2. GlusterFS Based IO Performance

The choice of GlusterFS is based on its features of scalability, high availability, ease of management and deployment for massive storage solutions. In our setup we have used a distributed GlusterFS for building the Cloud as well as for providing a shared area inside the Cloud resource. With all the advanced features of the GlusterFS, it is important to analyze the performance of the file system. Table 2 shows numbers generated from the GlusterFS profiler while running the infrastructure. Here it is important to note that these numbers are with

	Brick-1	Brick-2	Brick-3	Brick-4
Days	20	20	20	20
Total Read (GB)	40	41	169	831
Total Write (GB)	38	36	364	1017
Average - Maximum Latency (ms) / No. of Calls (in millions)				
Read	0.055-11.6/ 29.8	0.05-10.8/ 31.0	0.29-217.5/0.6	0.380-2386/0.32
Write	0.082-16.8/1.8	0.077-14.0/1.9	1.66-5151.2/10.8	1.062-7281/7.8

Table 2: GlusterFS-based IO performance.

minimum performance tuning. The first half of the table shows the total reads and writes to a certain brick in a period of 20 days. Brick-1 and Brick-2 shared 1 TB between the CE and the WNs. These Bricks contain data files from incoming jobs, ranging from kilobytes to gigabytes. The maximum latencies for the reads and writes of these bricks is less than 20 ms and the average latencies are even less than 0.1 ms. Brick-3 and Brick-4 cumulatively provide 2 TB of storage for the Nova compute and Glance repository. Here it is important to note that the size of the VM images is 10 GB, thus for each new VM a separate file of 10 GB is created. Table 2 also shows higher maximum latencies for brick-3 and brick-4 but the average latencies for these bricks flatten out the effect of these high peaks. The average latencies are less than 0.4 ms and 1.7 ms for reads and writes respectively. In future we expect more improvements with enhanced performance tuning.

5.3. Performance of Live Migrations

In terms of infrastructure management live migration is another key feature. It is extremely useful for overall load balancing and system maintenance. Live migration is one of the features provided by the OpenStack suite. There are two modes of live migration in OpenStack, block-based and shared storage-based. Block-based live migration is slow but but has the advantage of working for almost any setup. Shared storage-based live migration is fast and only requires change in the instance metadata. A number of experiments were performed with a variety of instances with the minimal VM of Ubuntu, m1.small, where the live migration took 6 seconds. On a WN VM with GlusterFS, CernVM-FS and Condor services running took, migration times of 43 seconds were measured. We have also experienced number of failure when migrating more than 10 VMs simultanously. The failure rate was random, with 20 migrations requested where, on average, 2 failed. And for requests of 15 VMs, 5 of them failed to boot successfully.

5.4. Overall System Stability

This test gives a broad overview of the infrastructure. The stability of the infrastructure was monitored using SGAS accounting data and CMS Dashboard data for SAM monitoring jobs. Regular reporting of the status of incoming jobs is one of the essential requirements for each site. Currently for a Cloud-based setup, there is no option of registering the executed jobs. Our choice of ARC middleware allows us to publish all the necessary information required to run a WLCG site. Figure 4 presents a four month window, from the CMS dashboard, of the status of our site. The green color illustrates that services are running OK whereas yellow are the warnings and red are the errors at some of the WNs. The unavailability of the service presented with the brown color. This was mainly due when the whole system was down. The downtime of the site was due to three main reasons, first is the compatibility issues of different releases of OpenStack, second was the size of the VM images and third was hardware malfunctions. According to SGAS data on a CSC server our setup has run more than 65k jobs starting from June 2013 until present. These consist of Monte Carlo simulation jobs, physics analysis jobs and monitoring jobs. In total we have run 55k CMS jobs with 789 walltime days. One example of the performance of

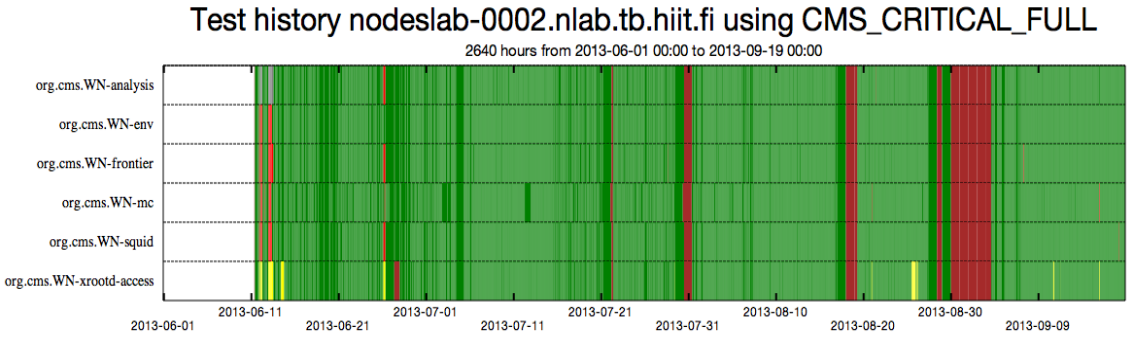

Figure 4: System stability over 4 months from CMS dashboard.

user analysis jobs is a set of 400 jobs that ran for 74 walltime days on this system with a CPU efficiency of 85%. These results show that the technologies we are focusing on have the strength to manage production-ready environments.

6. Conclusion

The results shown in this work describe our experience with the state-of-the-art system components used to build a scalable computational infrastructure. The system components, Nova, Glance and GlusterFS, ease the setup but achieving a configuration that produces the desired results is a non-trivial task. For example, burst mode VM boot requests or simultaneous live migration requires horizontal scaling of the OpenStack services.

In the future the aim will be to further evaluate the system performance and employ or develop components that will enhance the scalability and resilience of the setup. One direction will be to explore the Datacenter Indirection Infrastructure (DII), a comprehensive control plan based on dynamic network strategies, multi-path, computational mobility and infrastructure monitoring.

References

[1] Finnish Centre for Scientific Computing. `http://www.csc.fi`.
[2] Finnish Tier-2 WLCG resource. `http://gstat-prod.cern.ch/gstat/site/FI_HIP_T2/`.
[3] Nordic DataGrid Facility. `http://www.ndgf.org/`.
[4] Openstack, Open source software for building cloud infrastructure. `http://openstack.org/`.
[5] Standard Performance Evaluation Corporation. `http://www.spec.org/cpu2006/`.
[6] The Gluster file system. `http://www.gluster.org/`.
[7] WLCG: Worldwide LHC Computing Grid. `http://lcg.web.cern.ch/lcg/`.
[8] Buncic P. Blomer J., Aguado-Sanchez C. and Harutyunyan A. *Journal of Physics: Conference Series*, page 331, 2011.
[9] M. Ellert et al. Advanced Resource Connector middleware for lightweight computational Grids. *Future Gener. Comput. Syst.*, 23(1):219–240, 2007.
[10] C. D. Jones et al. The new CMS event data model and framework. In *Proceedings of International Conference on Computing in High Energy and Nuclear Physics (CHEP)*, 2006.
[11] E. Edelman et al. Running CMS Remote Analysis Builder jobs on Advanced Resource Connector middleware. In *J. Phys: Conference Series 331 062037*, 2011.
[12] P. Eerola et al. The NorduGrid production Grid infrastructure, status and plans. In *Proceedings of Fourth IEEE International Workshop on Grid Computing*, pages 158–165, 2003.