PROCEEDINGS OF SCIENCE

Opportunistic usage of the CMS Online cluster using a cloud overlay

Olivier Chaze*
CERN
CH-1211 Geneva 23, Switzerland
E-mail: olivier.chaze@cern.ch

Jean-Marc Andre (5), Anastasios Andronidis (2), Ulf Behrens (1), James Branson (4), Philipp Brummer (2), Cristian Contescu (5), Sergio Cittolin (4), Benjamin G. Craigs (2), Georgiana-Lavinia Darlea (6), Christian Deldicque (2), Zeynep Demiragli (6), Marc Dobson (2), Nicolas Doualot (5), Samim Erhan (3), Jonathan Richard Fulcher (2), Dominique Gigi (2), Frank Glege (2), Guillelmo Gomez-Ceballos (6), Jeroen Hegeman (2), Andre Holzner (4), Raúl Jimenez-Estupiñán (2), Lorenzo Masetti (2), Frans Meijers (2), Emilio Meschi (2), Remigius K. Mommsen (5), Srecko Morovic (2), Vivian O'Dell (5), Luciano Orsini (2), Christoph Paus (6), Marco Pieri (4), Attila Racz (2), Hannes Sakulin (2), Christoph Schwick (2), Thomas Reis (2), Dainius Simelevicius[a] (2), Petr Zejdl[b] (5)

1. DESY, Hamburg, Germany
2. CERN, Geneva, Switzerland
3. University of California, Los Angeles, Los Angeles, California, USA
4. University of California, San Diego, San Diego, California, USA
5. FNAL, Chicago, Illinois, USA
6. Massachusetts Institute of Technology, Cambridge, Massachusetts, USA

a) Also at Vilnius University, Vilnius, Lithuania
b) Also at CERN, Geneva, Switzerland

PoS(ISGC 2016)022

Abstract. After two years of maintenance and upgrade, the Large Hadron Collider (LHC), the largest and most powerful particle accelerator in the world, has started its second three year run. Around 1500 computers make up the CMS (Compact Muon Solenoid) Online cluster. This cluster is used for Data Acquisition of the CMS experiment at CERN, selecting and sending to storage around 20 TBytes of data per day that are then analysed by the Worldwide LHC Computing Grid (WLCG) infrastructure that links hundreds of data centres worldwide. 3000 CMS physicists can access and process data, and are always seeking more computing power and data. The backbone of the CMS Online cluster is composed of 16000 cores which provide as much computing power as all CMS WLCG Tier1 sites (352K HEP-SPEC-06 score in the CMS cluster versus 300K across CMS Tier1 sites). The computing power available in the CMS cluster can significantly speed up the processing of data, so an effort has been made to allocate the resources of the CMS Online cluster to the grid when it isn't used to its full capacity for data acquisition. This occurs during the maintenance periods when the LHC is non-operational, which corresponded to 117 days in 2015. During 2016, the aim is to increase the availability of the CMS Online cluster for data processing by making the cluster accessible during the time between two physics collisions while the LHC and beams are being prepared. This is usually the case for a few hours every day, which would vastly increase the computing power available for data processing. Work has already been undertaken to provide this functionality, as an OpenStack cloud layer has been deployed as a minimal overlay that leaves the primary role of the cluster untouched. This overlay also abstracts the different hardware and networks that the cluster is composed of. The operation of the cloud (starting and stopping the virtual machines) is another challenge that has been overcome as the cluster has only a few hours spare during the aforementioned beam preparation. By improving the virtual image deployment and integrating the OpenStack services with the core services of the Data Acquisition on the CMS Online cluster it is now possible to start a thousand virtual machines within 10 minutes and to turn them off within seconds. This document will explain the architectural choices that were made to reach a fully redundant and scalable cloud, with a minimal impact on the running cluster configuration while giving a maximal segregation between the services. It will also present how to cold start 1000 virtual machines 25 times faster, using tools commonly utilised in all data centres.

International Symposium on Grids and Clouds 2016
13-18 March 2016
Academia Sinica, Taipei, Taiwan

*Speaker.

1. Introduction

The Compact Muon Solenoid (CMS) experiment [1] is one of the two general purpose physics experiments at the Large Hadron Collider (LHC) [2] at the European Organization for Nuclear Research (CERN [3]). It is a general-purpose experiment covering a wide range of physics at the TeV scale. Recent results obtained by the CMS collaboration include the observation of a new particle with a mass of 125 GeV consistent with the Higgs boson of the Standard Model. Its Data Acquisition (DAQ) system reads data from more than 50 million channels through approximately 650 custom links. At the receiving end, data is fed into commercial networking hardware and subsequent stages of the data flow are implemented using commercial components. A two stage trigger system is used to reduce the amount of data kept for offline processing to match the available storage and computing resources. The first level, implemented in custom hardware, selects beam collisions ("events") at a rate of up to 100 kHz. The second trigger level, also called High Level Trigger (HLT), is implemented in software running on a dedicated computer farm receiving full events from the event-building network. CMS event building (EVB) is networking intensive, with more than 100 Gbyte/s throughput. The HLT is a CPU intensive process that selects 1 out of 1000 events. The selected events are subsequently stored and forwarded to the central data recording center at CERN (Tier0 [9]) by the storage manager infrastructure, which can sustain a throughput of around 11 Gbyte/s (heavy ion runs). About 300 Mbyte/s sustained throughput is sent to Tier0, averaging 20 TBytes per day. In order to implement the aforementioned functionalities and the control of the experiment a computing cluster of more than 1500 computers has been set up: the CMS Online cluster. It has been designed to cope with the peak luminosity requirements of the LHC. In addition, there is service time for the LHC and the detectors during which the cluster remains mostly idle. It means that the cluster could be used for a different purpose, when not fully in use by the data acquisition processes. The part of the cluster that runs the HLT encompasses most of the computing power with approximately 16000 cores. In the past, a cloud-controlling layer has been overlaid onto this part of the cluster to allow it to be used by external processes.

2. The CMS Online cluster

The CMS Online cluster serves the needs of a community of more than 900 users. It is autonomous (i.e. independent from all other networks). The CMS Online cluster has been designed with redundant services to avoid wasting accelerator time and to scale easily for future expansions.

2.1 Computing resources

The CMS Online cluster (Figure 1) is mainly organized into three stages:

- 1344 cores (84 x dual E5-2670 8-core CPUs) read data coming from the detector and the electronics. They act as the first stage event building.
- 16192 cores filter the events. They act as High Level Trigger farm. Three generations of hardware: 288 x dual X5650 6-core + 256 x dual E5-2670 8-core CPUs + 360 x dual E5-2680v3 12-core.

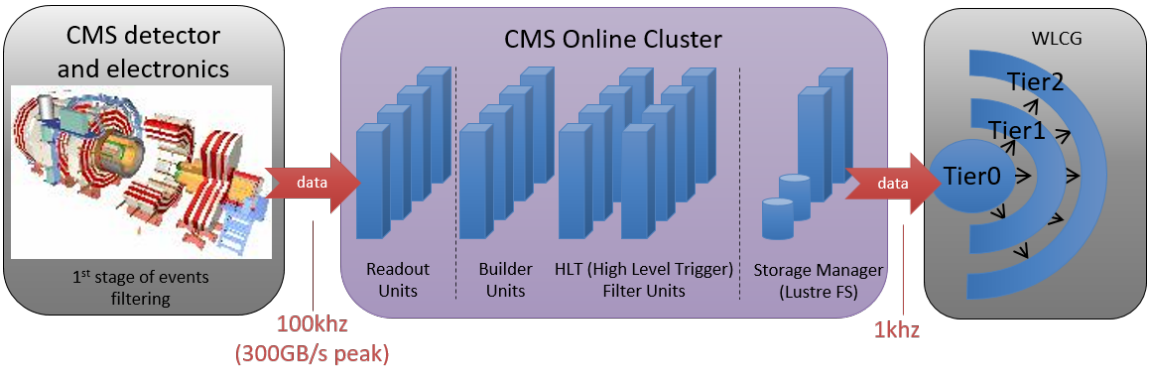

Figure 1: Flow of data in the CMS Online Cluster

- A Lustre file system [8] stores the data before shipping it to Tier0 [9] which is the starting point of the computing grid. It acts as the storage manager [10].

About four hundred computers are used by the sub-detectors to control their electronics, and a hundred more are used for monitoring and testing.

To get an idea of the HLT farm potential, its raw compute power in 2015 (HS06 benchmark score [11] = 352K) is greater than the Tier0 (score: 300K) computing center and greater than all CMS Tier1 sites together (score: 300K). The HLT HS06 benchmark for 2016 is forecast at 480K as the oldest generation of hardware will be replaced.

2.2 Networking in the CMS Online cluster

The CMS Online cluster makes use of the following networks (Figure 2):

- The CMS Control networks. All computers in the CMS cluster are connected to one of these networks, at the level of one network per rack. They all constitute a larger routed network. It is built using over 1000 ports across more than 100 switches and four routers in a redundant configuration.

- The CMS Data networks. These are additional high bandwidth networks used for the event building, connectivity to the High Level Trigger filter farm and to the storage manager. These high bandwidth networks are composed of 40 Gbit/s Ethernet and 56 Gbit/s Infiniband switches.

- The CERN public network (GPN). This network connects the CMS Online cluster to the outside world through the CERN routers and the CMS access servers that act as computer gateways.

- The Central Data Recording private network (CDR). This network allows data transfer from the storage manager computers to the Tier0 at CERN for later analysis and archiving. It is served by four bonded 40 Gbit/s links (160 Gbit/s).

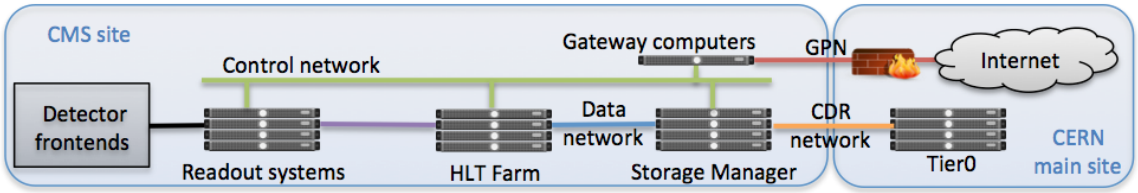

Figure 2: Networking in the CMS Online Cluster

3. Cloud overlay on top of the CMS Online cluster

The HLT farm, with its 16000 cores, concentrates most of the compute power of the CMS Online Cluster and therefore the cloud has been deployed on top of it.

3.1 Requirements and constraints

The most important requirement is to have no impact on data taking. The primary role of the CMS Online Cluster is to read and filter data from the detector, meaning the compute resources can only be allocated to the cloud when they are free, therefore in an opportunistic way. It is also not possible to significantly change the current network infrastructure or the compute nodes configuration. Big changes would imply revalidation of the system, which is something that cannot be afforded.

3.2 The CMS Online cloud with OpenStack

The CMS Online Cloud is an overlay on top of the CMS Online Cluster able to act as a WLCG tier site running data analysis jobs. Therefore the CMS Online Cloud needs access to several resources outside the CMS site, as described in the next sections. The Cloud architecture based on the OpenStack Grizzly release [4] is made of user frontends to control the cloud: Nova and EC2 for the command line API. The management of the virtual machines disk images is done by Glance. Several other tools are used, such as Keystone for the identity service, nova-compute for the compute node virtualization, and nova-network for the network virtualization. Lastly, OpenStack also needs a few more external tools: a message passing system like RabbitMQ [5] and a database backend, in our case a MariaDB/Galera Cluster [6]. All OpenStack services except Glance are managed inside a Corosync/Pacemaker cluster [7] across four servers to ensure their high availability. The Glance service is running on a dedicated server and a second one in passive mode is ready to take over if necessary.

On the compute nodes, nova-compute, nova-network, nova-metadata, Libvirt and KVM have been deployed. The changes done on the compute nodes are mainly confined to Network Address Translation (NAT) rules to redirect the traffic either through the dedicated VLAN of the Data Network to access Tier0 or through the GPN network to access external resources. The architecture is depicted in Figure 3. Additionally several custom scripts were deployed on the compute nodes where open source tools were not available. Some of them configure the networks and check the routing/NAT rules. Another one checks the consistency between the virtual machines running on

the compute nodes and the OpenStack database content, and makes the necessary clean up (delete the VMs, clean the Libvirt files, etc).

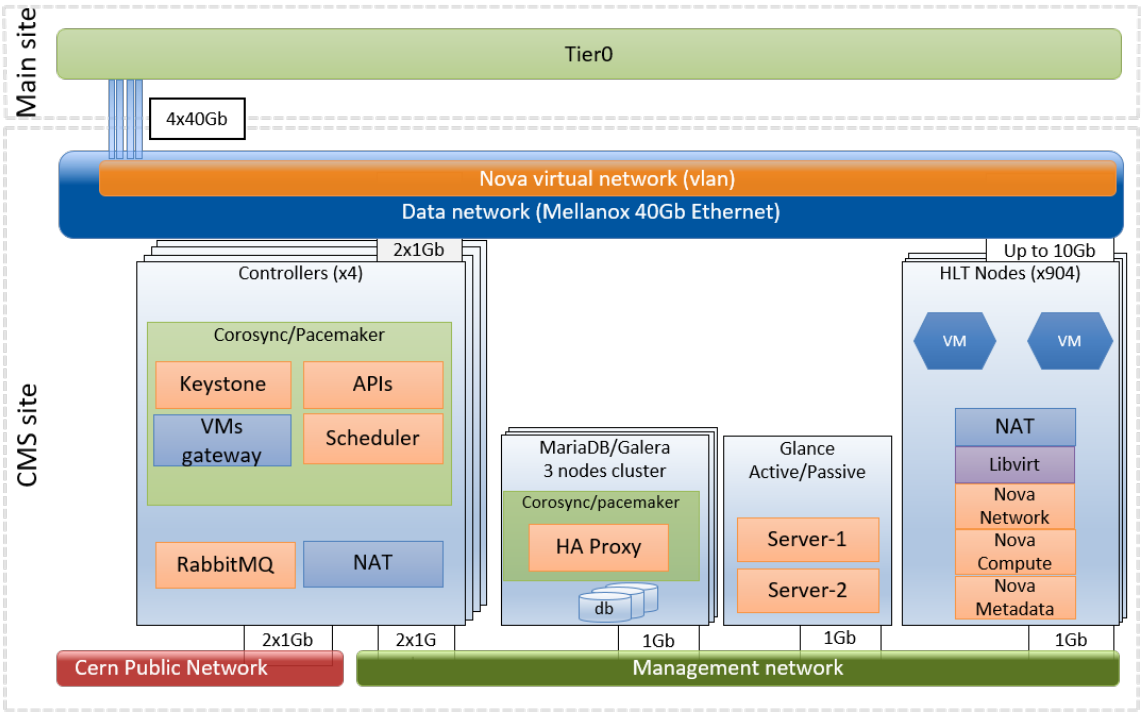

Figure 3: OpenStack architecture used until early 2016

The virtual machines and the controller can communicate using the nova flat network. The virtual machines can also reach the external world through the GPN interface of the controller nodes which use source NAT. A Squid server [13] has been deployed on each compute node and defined as proxy by the virtual machines to speed up the download of the software needed for data analysis.

3.3 Redundancy of OpenStack components using Corosync/Pacemaker

This section describes how Corosync/Pacemaker have been used to make the different software components in the cloud redundant and scalable. Pacemaker provides a high availability cluster resource manager which uses the Corosync cluster engine as a group communication system across the nodes of the Pacemaker cluster (in our case the cloud controller nodes). The Pacemaker software defines the services which need to run and takes care of starting/stopping/checking the services. The Corosync software provides a layer of synchronisation across the different machines used to provide the redundancy and high availability. It has methods to check the correct functioning of the services defined in Pacemaker on the cluster of nodes designed to run this service. Different services can be linked or grouped to be able to move complete functionality between hosts. For example a specific service like the OpenStack Keystone component now becomes virtual as it can run in many physical hosts and needs to have an associated virtual IP which moves with it. When Corosync realises a service, e.g. the OpenStack Keystone component, is no longer

running, it will implement a recovery procedure. In our case, the Corosync STONITH mode (Shoot The Other Node In The Head) is used which reboots the node on which the service was running. At the same time Pacemaker will allocate this service to another one of the redundant nodes and starts the service or services there. In the example, the virtual IP of the Keystone service will be set up on the new node and the keystone software started.

3.4 Virtual machines and WLCG jobs specifications

The virtual machines of the HLT Cloud follow the grid specifications of running virtual machines with a multiple of four cores virtual machines. Depending the generation of hardware the compute nodes host either a single virtual machine with twelve cores and 18 GBytes of memory, two virtual machines with eight cores and 13 GBytes of memory, or three virtual machines with eight cores and 19 GBytes of memory. At full capacity the HLT Cloud can run up to 1880 virtual machines. They all use the same image. The typical WLCG jobs running in the HLT are single core, request 2.5 GBytes of memory each and have a targeted processing time of eight hours. The available memory of the virtual machines avoids to make use of all cores, which is known and concerns many Tier sites. However, multi-core jobs will be available in the future (lower memory foot print).

3.5 The production phase

The architecture has been used for more than two years and gives the required functionality and performance. The CMS HLT Cloud contributes to offline data analysis like any other Tier site. With its high bandwidth connectivity to Tier0 (160 Gbit/s) it is capable of running different kinds of jobs, even network intensive ones, as shown in Figures 4 and 5 below. The contribution of the HLT Cloud to the processing of grid jobs is significant, an example of number of jobs completed with 70 percent of the HLT farm is depicted in Figure 6

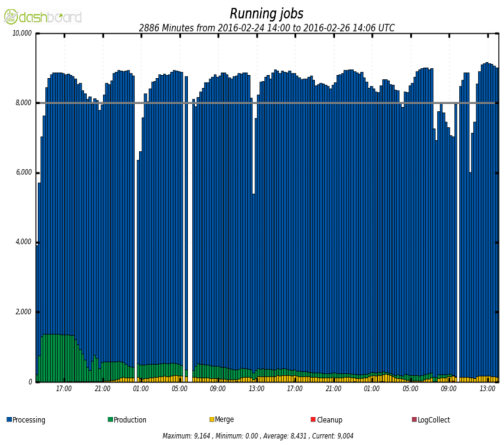

Figure 4: Single core jobs running in the CMS HLT Cloud

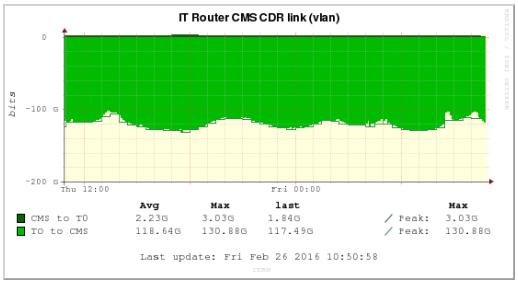

Figure 5: Network usage of the CMS HLT Cloud

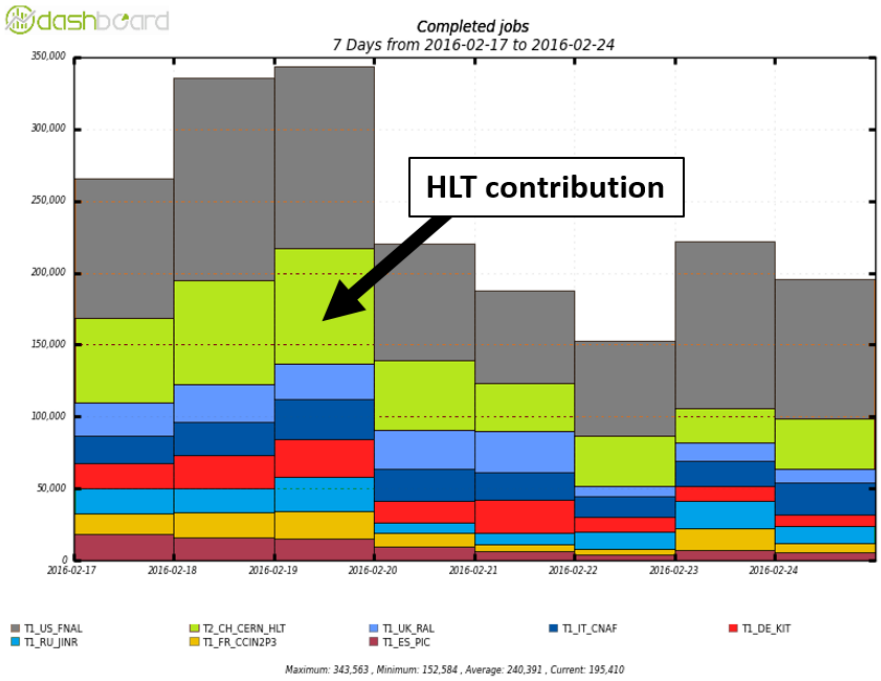

Figure 6: Completed jobs by the HLT Cloud and Tier1 sites during a week

4. The 2016 challenge: Cloud usage during inter-fill periods

So far the CMS HLT Cloud has been used only during maintenance periods. The next challenge is to extend its usage during inter-fill periods which correspond to short periods of a few hours while the beams of particles are being refilled. These inter-fill periods happen at least once a day and during this time, the CMS Online cluster is mostly idle as there is no collisions happening in the detector. The 2015 LHC statistics show that the period of time while there was collisions represented only 20 percent of the time (maintenance periods excluded).

4.1 Requirements and constraints

The inter-fill periods are short (an average of three to six hours expected for 2016) and unpredictable. They can occur at any time of the day or night, even during the weekend, therefore the HLT Cloud operation must be fully automatised and be started and stopped within a few minutes to obtain a satisfactory efficiency. It also requires caution to avoid any impact on data taking as this is the primary role of the CMS Online Cluster. The next sections describe the ongoing and achieved work.

4.2 Speed up the Cloud start

The first challenge was to improve the distribution of the virtual image across the compute nodes. With no tuning it takes almost four hours to cold start 1000 virtual machines. The reason for this is that the Glance server has to serve a 1 GByte image to 900 compute nodes via a 1 Gbit

link. As the inter-fill periods last between three and six hours on average, a significant fraction of this time would be spent on booting and therefore this would result in an extremely poor efficiency. The image distribution has been significantly improved using a combination of common tools and custom scripts as detailed in the next sections.

4.2.1 Increase the bandwidth available to serve the virtual machine image

As the network is the bottleneck, the logical solution is to increase the link speed of the Glance server, which has been done. As a test, a Glance server has been setup with a 10Gbit/s link. From 3.7 hours to start 1000 virtual machines it still takes one hour despite a ten times faster link speed as shown in Figure 7. The reason is that the Nova Glance and the Nova Scheduler induct on overhead (scheduling of the image distribution).

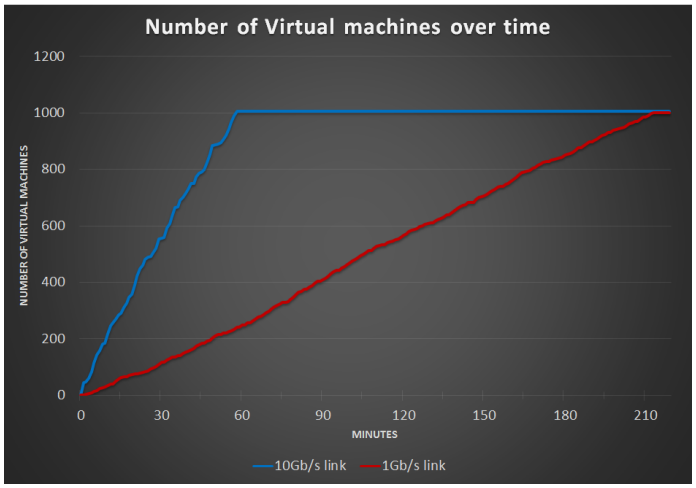

Figure 7: From 3.7 hours to 1 hour to cold start 1000 VMs

4.2.2 Pre-cache the image in Nova cache beforehand

In order to reduce the overhead inducted by the scheduling of the image distribution, an embedded feature of OpenStack has been exploited. Once a virtual machine has been started on a compute node, the virtual machine image is stored locally in the Nova cache. The instantiation of the next virtual machines of the same type on the same compute node will be sped up as they will use the image stored locally in the Nova cache instead of downloading it again from the Glance server. With the help of a custom script, the image is pre-cached on all compute nodes before the virtual machines are actually started. It means that even the first virtual machine instantiated on a compute will use the image stored in the Nova cache.

4.2.3 Compress the virtual machine image

A simple way of reducing the time required to transfer the image is simply by compressing it. For instance a raw image of one gigabyte can be reduced to a few hundred megabytes. Now, the virtual machine image is compressed and exposed to the compute nodes via an Apache server [12] added on the Glance server.

4.2.4 Improve the distribution using proxies

In order to improve the bandwidth available to serve the virtual machine image, we re-use the Frontier Squid servers [13] (mandatory for data analysis) installed on the four OpenStack Controllers. The compute nodes use them as proxies to download the compressed image hosted on the Glance Apache server. The full pre-caching mechanism is depicted in Figure 8.

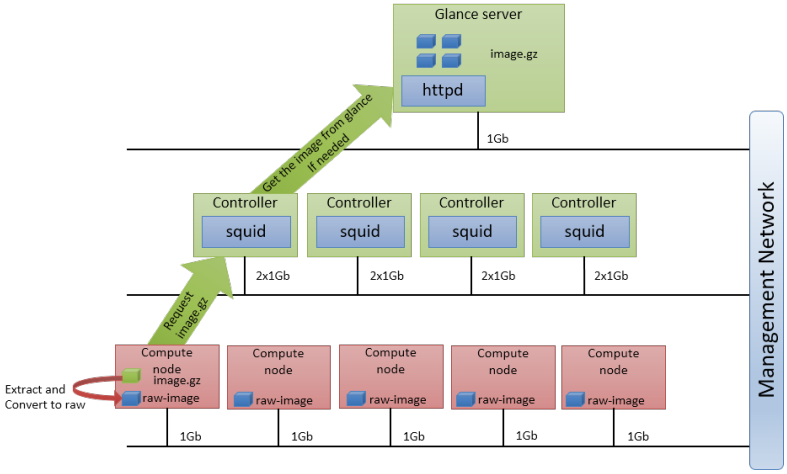

Figure 8: Virtual machine image pre-caching mechanism

This pre-caching mechanism reduces the load on the OpenStack infrastructure and especially on the nova-scheduler because the Glance server is taken out of the startup loop as there's no need to distribute the image anymore at that stage. As a consequence more virtual machines can be started at a given time, reducing from 3,7 hours to 8 minutes the cold start of 1000 virtual machines as depicted in Figure 9.

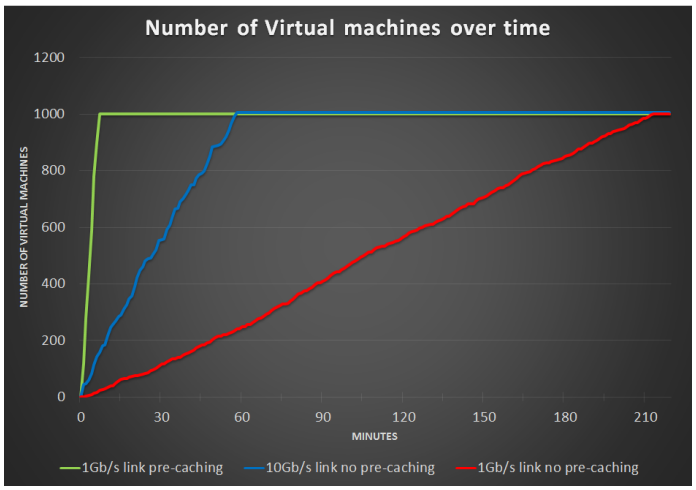

Figure 9: From 3.7 hours to 8 minutes to cold start 1000 VMs

4.3 Select nodes allocated to the Cloud

Even though there are no beam collisions to be processed during the inter-fill periods, test and calibration runs are taken during this time. Therefore only a fraction of the HLT can be allocated to the Cloud in these periods. The machines of the HLT Cluster are organised into smaller clusters called Builder Unit Appliances (Figure 10), where each one is composed of one head node (Builder Unit) and up to 16 workers (Filter Units) depending of the generation of hardware. Filter Units are the compute nodes of the HLT Cloud. In order to preserve the functionality of the CMS Online Cluster during the inter-fill periods, all Builder Unit appliances must be left with a minimum amount of healthy Filter Units. It has been decided that a minimum of four Filter Units will be left per Builder Unit during the inter-fill periods. From the HLT Cluster configuration stored in a database, including the faulty Filter Units under maintenance, the list of compute nodes that can be allocated to the Cloud is automatically generated (Step 2 in Figure 12). The script also ensures, if possible, that the four Filter Units left in the Builder Unit Appliance are in four different physical chassis to avoid losing them all at once if a physical problem occurs on the chassis. If for some reason a Builder Unit Appliance has less than four Filter Units available then no Filter Units from this Builder Unit Appliance will be allocated to the HLT Cloud. The fraction of the HLT Cluster that can be allocated to the cloud is configurable in case inter-fill tests require more computing power.

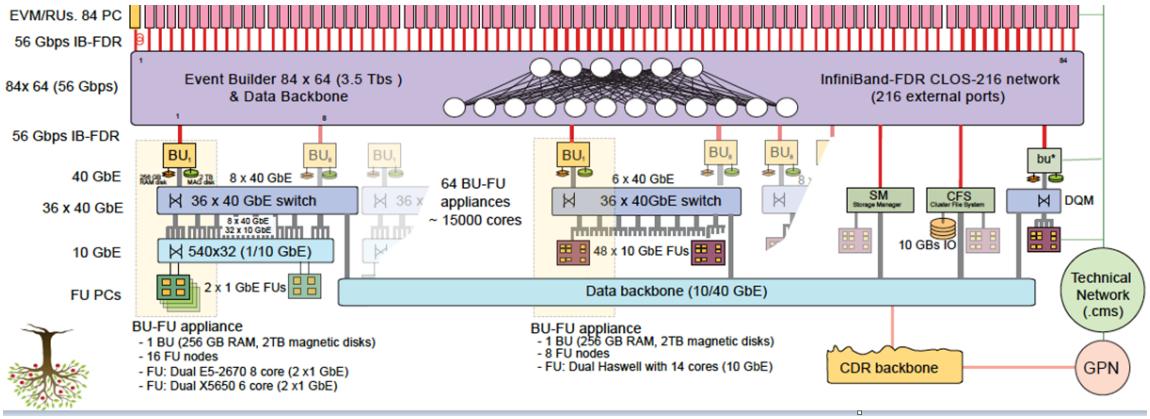

Figure 10: Extract of the CMS Online Cluster

4.4 Automatise Cloud operation

4.4.1 Detect the beginning and end of the inter-fill periods

The inter-fill periods can happen at any time of the day or night. The start and stop of the HLT Cloud must be automatised and therefore require a trigger, which will be the LHC (Large Hadron Collider) machine mode [14]. The CMS Online Cluster is collecting data when the LHC machine mode is “stable beam”, At the end of a fill, the LHC enters the “beam dump” and then “ramp down” modes. The “ramp down” mode will be the trigger to start the HLT Cloud. After re-injecting beams into the LHC, the energy of the beams is increased to the collision energy during the “ramp” mode (happening few minutes before the next ’stable beam’ period) at which point the Cloud is stopped.

An example of inter-fill period is depicted in Figure 11. These LHC machine modes are easily available via a SOAP interface.

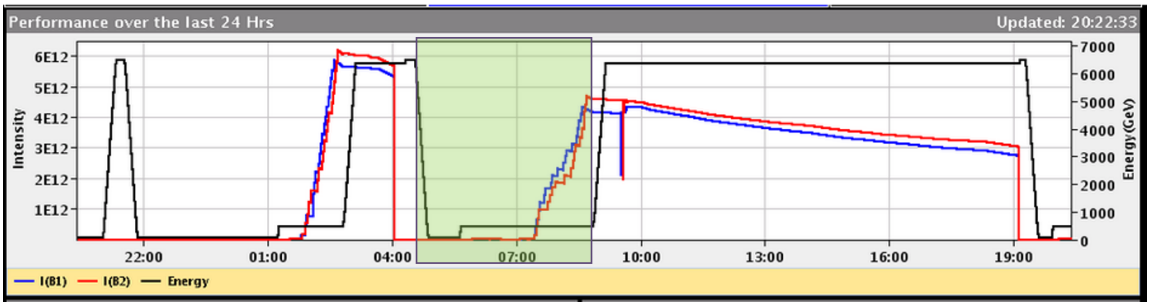

Figure 11: Beam intensity (red and blue lines) and energy (black line) graph for a 24 hour period. The green area shows an example of an inter-fill period.

4.4.2 Control the OpenStack services at the compute node level

The selection of compute nodes that can be used for the HLT Cloud has been described in section 4.3. The next step is to start and stop the OpenStack services on the selected nodes. The OpenStack services are managed by a script deployed locally on each compute node, which simply calls the system V init scripts (Step 4 in Figure 12). On the other hand it ensures no virtual machines are running after the services are stopped. Experience showed that the OpenStack APIs (ec2, nova) aren't reliable when it comes to stopping the virtual machines, as some still run and are impossible to stop using the common OpenStack APIs. It must be added that the APIs are too slow to stop the virtual machines during inter-fill periods. As a consequence the home-made script takes care of stopping the OpenStack services, killing the virtual machines, removing their related files and eventually marking them as deleted in the OpenStack database (Step 3, 4 and 5 in Figure 13).

4.4.3 Control the OpenStack services at the cluster level

Now that the OpenStack services and virtual machines are controlled in a reliable manner at the compute node level they need to be controlled at the cluster level. The compute nodes run a daemon (HLTd) which manages the data taking applications. We have extended this daemon to also manage the OpenStack services through the script described in section 4.4.2. The compute nodes role can be switched between "data taking" and "cloud" on demand and from a central server using the HLTd daemon API (Step 3 in Figure 12 and step 2 in Figure 13).

4.4.4 Automatically instantiate the virtual machines

As soon as the OpenStack services are started, the openstack-compute service running on each compute node will automatically update the OpenStack database with the compute node resources available, namely the number of cores, the memory and disk space. A script regularly checks the database content and detects that new resources (VM slots) are available (Step 7 in Figure 12). The script will then call the ec2 OpenStack API to instantiate the virtual machines (Step 8 in Figure 12), the virtual machine image to be used is defined in advance and the same image runs in the whole

cluster. In addition, to avoid overloading nova-scheduler the script also ensures that a maximun of 50 virtual machines are being scheduled simultaneously. The flavor of the virtual machine (Number of cores, memory and disk) is automatically selected according to the hardware of the compute node. Three flavors exist, one for each generation of hardware that makes up the CMS HLT Cluster. To ensure that the virtual machine will be spawned on the right generation of hardware a custom nova-scheduler filter has been implemented. All the steps described above, the generation of the list of compute nodes that can be allocated to the cloud, the detection of the LHC machine modes, the call to the HLTd API and the call to the ec2 API to instantiate the virtual machines are part of a same daemon running on the OpenStack Controllers called cloud-igniter. The full process is depicted in Figure 12 and Figure 13.

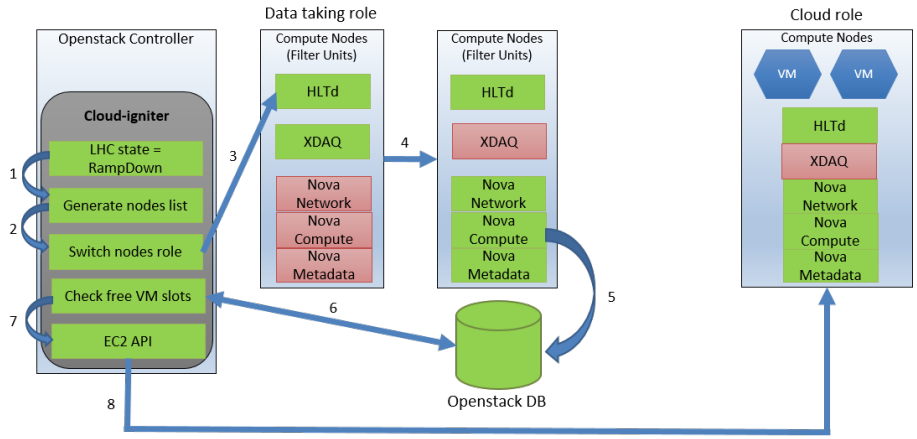

Figure 12: Schematic view of automatic cloud start

1.Entering RampDown mode 2.Generate compute node list 3.Contact nodes 4.Switch role 5.nova-compute automatic DB update 6.Detect available VM slots 7.Call ec2 API 8.VMs started

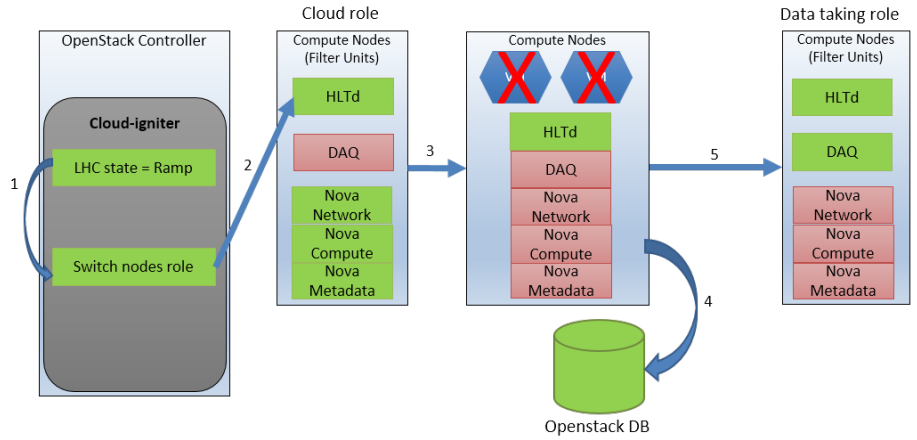

Figure 13: Schematic view of automatic cloud stop

1.Entering Ramp mode 2.Contact nodes 3.Kill VMs 4.Clean OpenStack DB 5.Switch role

5. Proof of concept

Figure 14 demonstrates that the HLT Cloud has been automatically operated using the daemon cloud-igniter detailed in previous sections. Thousands of cores have been automatically allocated to the HLT Cloud during real inter-fill periods and released for data taking as expected when the next run of physics was about to start. These Cloud runs have been triggered only for inter-fill periods happening during the work-hours, which explains why the HLT Cloud hasn't been started over the weekend or if the inter-fill periods occurred outside work-hours periods when the supervision couldn't be guaranteed.

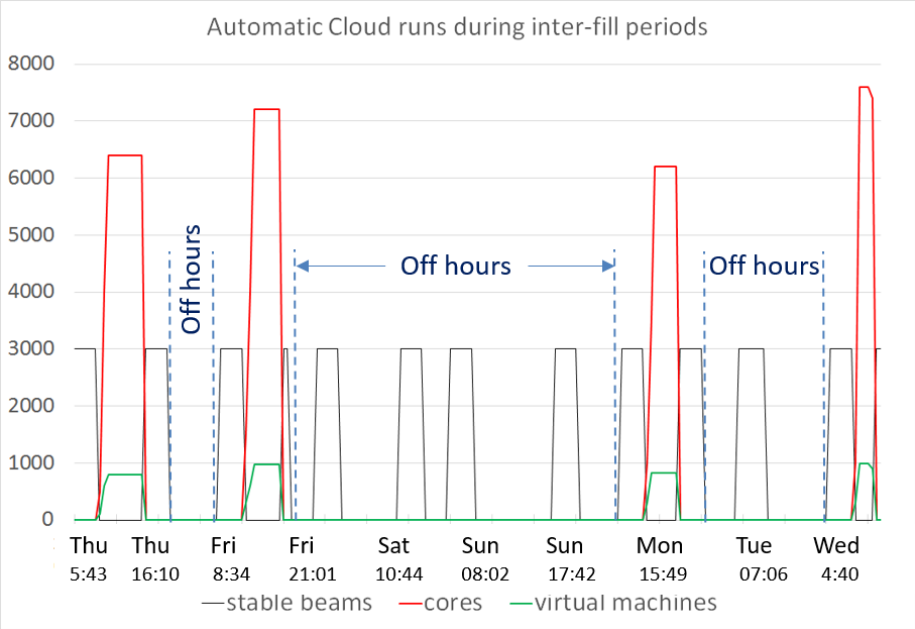

Figure 14: The HLT Cloud is started automatically at "ramp down" LHC mode and stopped at"ramp" mode

6. Efficiency of the HLT Cloud during the inter-fill periods

The typical grid jobs are defined such that they are completed after eight hours of processing. Eight hours is too long for the inter-fill periods, therefore, dedicated jobs with a targeted processing time of two hours will be created for the HLT Cloud. The 2015 statistics of the LHC operation [15] show that the LHC has been in "stable beams" mode for 20 percent of the time (maintenance periods excluded). It means that the HLT Cloud can be used the rest of the time, not exactly 80 percent of time (time between two stable beams) but closer to 75 percent, as the inter-fill periods correspond to the time between "ramp down" and "ramp" modes. It represents an accumulated time of 192 days. The worst case scenario is that 50 percent of the jobs are completed during the inter-fill periods. For instance, during a 3 hours inter-fill period, the first bunch of two hours jobs will be completed but the next one will be killed half way from completion. Thid still represents 96 days of actual job processing time, which is significant as it represents the same processing time than the maintenance periods. In addition to this, the time allowed for the HLT Cloud to run could

be extended in the future by allocating compute nodes to the HLT Cloud depending on LHC beams intensity. The full capacity of the HLT cluster is only required when the beams of particles are at their highest intensity, happening at the beginning of the stable beams period. Quickly, the intensity of the beams decreases and so does the needed online computing power. Therefore, compute nodes could be allocated to the HLT Cloud earlier and would give several extra hours to process grid jobs.

7. Recent upgrades in the CMS OpenStack architecture

7.1 Glance service integration in Corosync/Pacemaker

In February 2016 the OpenStack architecture has been migrated to new hardware. The new OpenStack controllers now have 10 Gbit/s interfaces and the cloud-igniter script has been integrated into the Corosync/Pacemaker cluster. The only architectural change concerns the Glance service. As depicted in Figure 3, the Glance service was hosted on a dedicated server and another in stand by was ready to take over if necessary. In the new architecture depicted in Figure 15, the Glance service has been integrated into the Corosync/Pacemaker cluster and associated with a virtual IP address. The virtual machine images are now hosted on the CMS NFS filer so they are accessible from the four controllers. This change provides a high availability for the Glance service and reduces the number of servers needed.

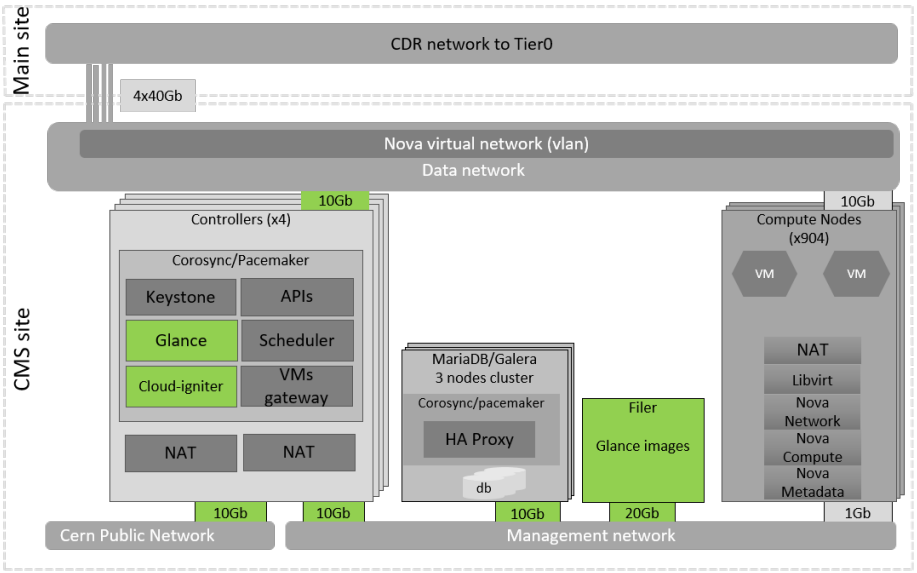

Figure 15: Relevant changes (green) in the 2016 CMS HLT OpenStack architecture

8. Conclusion

An overlay Cloud layer has been deployed on the CMS Online High Level Trigger cluster with zero impact on data taking. The Cloud overlay is able to control 900 compute nodes and deploy thousands of virtual machines within a few minutes in a stable manner. With respect to Grid

resources available for CMS, the HLT Cloud cluster offers significant potential as it represents the same amount of computing power as the combined capacity of all CMS Tier1 sites (2015). The proof of concept that the HLT Cloud can be started during inter-fill periods in an automatic manner is a major achievement and will double its contribution to the WLCG, representing an accumulated processing time of 300 days (Maintenance and inter-fill periods). The ongoing work consists of making the cloud-igniter daemon completely robust by handling all potential error cases that may have an impact on data taking. The cloud-igniter daemon must be fully fault tolerant to hardware issues and loss of communication between the different services involved in the process. In the mean time the inter-fill HLT Cloud runs are being commissioned by experts from the computing grid.

Acknowledgments

This work was supported in part by the DOE and NSF (USA).

References

[1] The CMS Collaboration (Adolphi R et al.) "The CMS Experiment at CERN LHC", JINST 3S08004 361, 2008.

[2] The LHC Study Group, "The Large Hadron Collider Conceptual Design Report", CERN/AC 95-05, 1995.

[3] www.cern.ch

[4] www.openstack.org

[5] www.rabbitmq.com

[6] https://mariadb.com/kb/en/mariadb/what-is-mariadb-galera-cluster

[7] www.clusterlabs.org

[8] www.lustre.org

[9] www.wlcg.web.cern.ch

[10] L. Darlea et al., "Online data handling and storage at the CMS experiment", 21st International Conference on Computing in High Energy and Nuclear Physics (CHEP2015), Okinawa, Japan, 13-17 April 2015, direct link: https://cds.cern.ch/record/2016893/files/082009.pdf

[11] www.spec.org/cpu2006

[12] www.apache.org

[13] http://frontier.cern.ch/

[14] R. Alemany, M. Lamont, S. Page et al., "Functional specification LHC MODES", ref. doc LHC-OP-ES-0005-10-00, 2007.

[15] M. Solfaroli, "LHC operation and efficiency in 2015", LHC Performance Workshop, Chamonix, France, 25-28 January 2016