Journal of Physics: Conference Series

PAPER • OPEN ACCESS

Scaling the CERN OpenStack cloud

To cite this article: T Bell *et al* 2015 *J. Phys.: Conf. Ser.* **664** 022003

View the article online for updates and enhancements.

Scaling the CERN OpenStack cloud

T Bell[1], B Bompastor[1], S Bukowiec[1], J Castro Leon[1], M K Denis[1], J van Eldik[1], M Fermin Lobo[3], L Fernandez Alvarez[1], D Fernandez Rodriguez[3], A Marino[4], B Moreira[1], B Noel[1], T Oulevey[1], W Takase[2], A Wiebalck[1] and S Zilli[1]

[1]European Organization for Nuclear Research (CERN), Geneva, Switzerland
[2]High Energy Accelerator Research Organization (KEK), Tsukuba, Ibaraki, Japan
[3]Universidad de Oviedo, Oviedo, Asturias, Spain
[4]University of Manchester, Manchester, England, United Kingdom

Abstract. CERN has been running a production OpenStack cloud since July 2013 to support physics computing and infrastructure services for the site. In the past year, CERN Cloud Infrastructure has seen a constant increase in nodes, virtual machines, users and projects. This paper will present what has been done in order to make the CERN cloud infrastructure scale out.

1. Introduction

The CERN private cloud has been in production since Summer 2013. During its first two years, this OpenStack cloud has grown to run more than 12000 virtual machines, hosted on 5500 compute nodes, across the Meyrin and Wigner data centres. Usage is very dynamic, with an average of 3000 virtual machines being created and deleted every day.

In addition, the CERN private cloud provides a volume service, that allows its users to create and mount volumes on their virtual machines. These volumes are provided by Ceph and NetApp storage services.

The CERN private cloud is used by 1900 registered users, and provide resources to 200 so-called shared projects for experiment communities and service providers. The virtual machines provide a large range of services to their owners, including compute-intensive data processing, service nodes for WLCG and CERN services, development servers.

The constant increase in nodes, virtual machines, users and projects required some improvements on our side to be able to scale without problems. In this paper we will present what has been done to make the CERN cloud infrastructure scale out easily.

2. Nova

The OpenStack compute service, also known as Nova, is responsible for provisioning instances using virtualization technologies. Nova has also the responsibility of managing the compute resources where the instances are provisioned. There are several possible solutions that can be used to scale Nova in order to support thousands of nodes.

To deploy CERN cloud infrastructure we decided to use Cells (figure 3). Cells allow us to divide the deployment into smaller groups, structured as a tree, by creating several Nova instances that can share other OpenStack services like for example Glance, Keystone and Cinder.

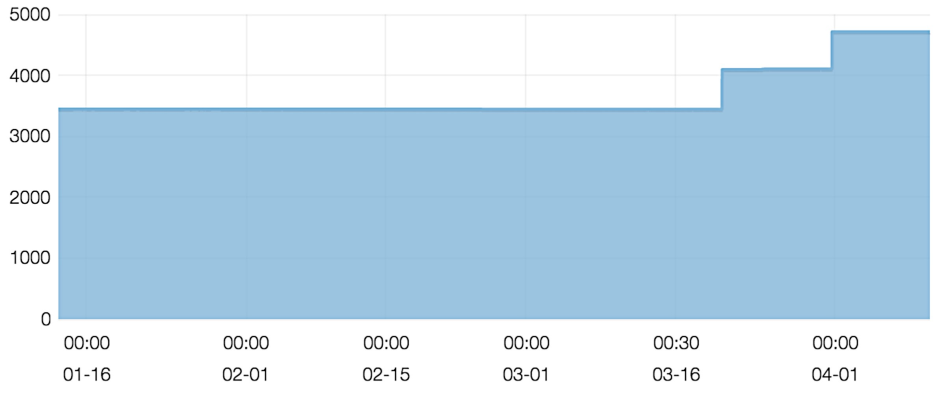

Figure 1. Number of hypervisors

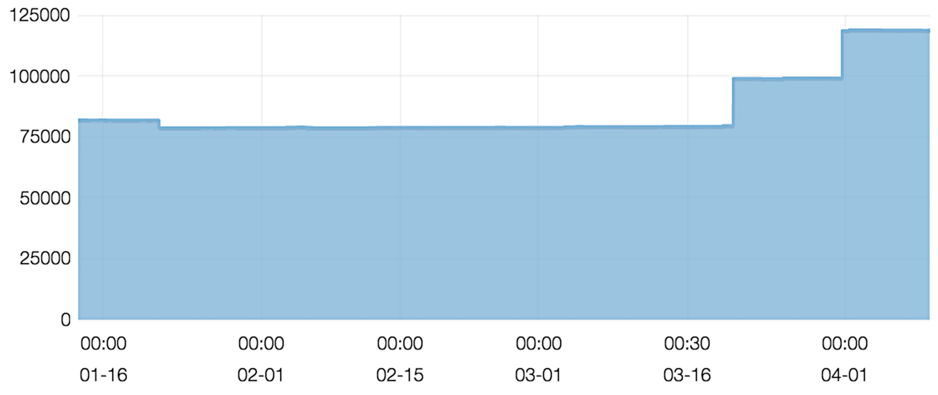

Figure 2. Number of cores

This solution differs from regions since those are independent cloud infrastructures with no share of services between them.

At CERN, the tree structure has two levels: a parent cell, that runs the user facing APIs and the cell scheduler, and children cells, that run all the others Nova services. Child cells are composed of two kind of nodes: nodes dedicated to the cell control plane, responsible for scheduling instances in the available compute nodes and manage their network, and compute nodes, in charge of hosting VMs. Each cell has its own database and message queue broker, allowing the infrastructure to scale in a distributed way, without overloading these components. Adding a new children cell simply means setting up a normal Nova infrastructure and connecting it with the parent cell. All the communication between parent and children cells is established through RPC. In a cloud infrastructure that uses cells, all requests go through the parent cell which schedules and redirects them to the correct child cell.

CERN compute nodes are heterogeneous in terms of hardware, network and operating system. We try to group compute nodes with the same characteristics in the same cell in order to dedicate specific type of resources to projects with specific requirements. In order to improve cell scheduling, we have developed several new filters which allow to create VMs taking into account availability zone, data centre, hypervisor type and project. This helps scaling consistently since we can add cells considering the configuration needs.

As many other cloud offerings, OpenStack Nova has the concept of availability zones. An availability zone can be defined as and isolated location in the scope of network and power

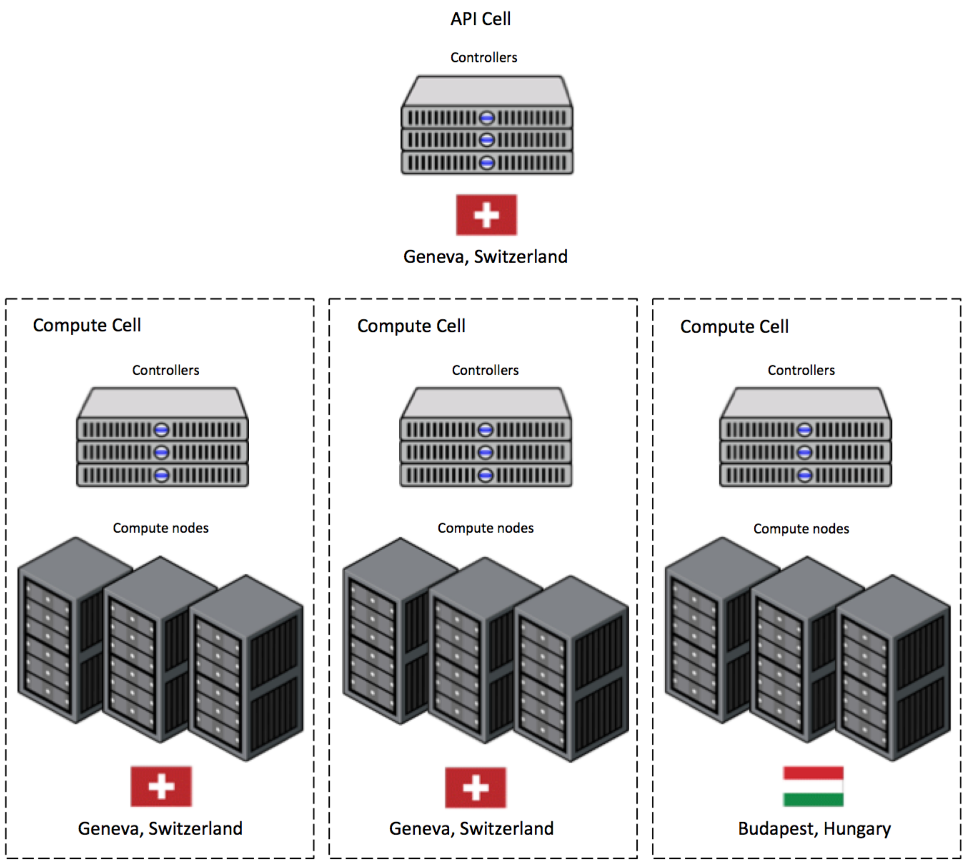

Figure 3. CERN cloud infrastructure cell configuration

infrastructure inside the same region. At CERN we map one availability zone per cell. This gives us the isolation in terms of cell control plane. If a cell control plane is affected, considering the distributed architecture of cells, the others are not affected. Thanks to this we can avoid having to cluster databases and message brokers at cell level. Moreover, the infrastructure is very resilient: if a component fails, others cells should continue to respond not affecting the general availability of the infrastructure.

Even though cells are a great functionality, they are still in testing stage. This means that not all Nova functionalities are available with this type of deployment. During the latest OpenStack development cycle (Kilo) important work was started by the Nova community with the objective of improving cell architecture and making it the default configuration (Cells V2)[1]. One of the biggest problems of cells is that replicates information between the parent and children cells. This database synchronization is a hard problem and it will be removed in the new cells architecture.

3. Keystone

Keystone is an OpenStack project that provides identity, token, catalog and policy services for use specifically by projects in the OpenStack family. During last year, the service went through numerous changes in order to add new functionalities and to make it more scalable. There were mainly two reasons for this change. First we wanted to be able to allow a seamless transition between API versions and to introduce new authentication methods, like Kerberos, X.509 certificates and Single-Sign-On, without having any impact on the users. Second, we wanted to have an architecture that would have allowed us to extend the service to support the load generated by Ceilometer's active polling mechanism. In order to achieve this we moved from a fully physical architecture to a distributed architecture composed of both virtual and physical servers.

User requests go through a first layer of load balancers that inspects and redirects them to the appropriate backend, based on the URL accessed. Right now we have 3 different backends: one for authentication using v2 API, one for v3 API with support to different authentication mechanisms (password, token, Kerberos, X.509 and WebSSO) and a last one specific for Ceilometer (figure 4). Each backend can be scaled up independently to handle the load generated by the users. The information required for the identity service is stored in Active Directory (LDAP) and a SQL database. In order to minimize the accesses to those stores a cache subsystem has been implemented, reducing the load on the LDAP store from 500 connections/sec to 10 connections/sec. Identity services are running mainly on virtual machines, keeping just a frontend and a backend on physical machines in case of a major incident.

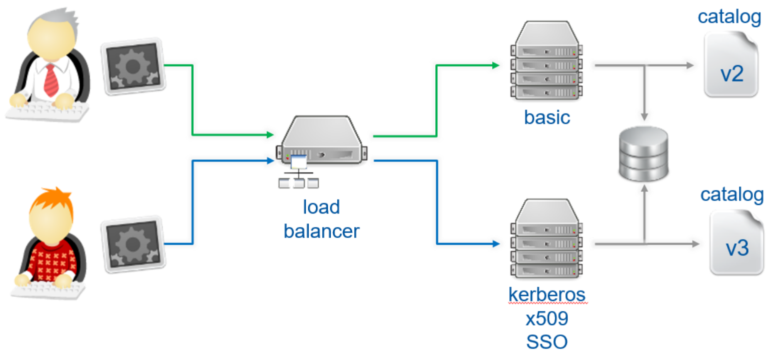

Figure 4. Keystone architecture

Since Icehouse release, the identity service is able to authenticate users and assign them to roles using WebSSO. The user who tries to access the service, typically from the Horizon website, is first redirected to the CERN Identity Provider (https://cern.ch/login). Then, after selecting the credentials he wants to use, the user is forwarded to the identity service with a SAML2 assertion. This assertion contains unique information about the user used to assign him the correct roles, as specified in the federated mapping and the local assignment mapping. After this operation, the user receives a keystone token that is used the Horizon for the consequent OpenStack API calls[2].

In the next months, we will add several new features to further improve the manageability of the service. In particular we will add support to domains, hierarchical projects and endpoint filtering. The first will allow us to delegate the project and user management to the project managers while endpoint filtering will allow us to configure cloud services on demand.

4. Glance

The OpenStack image service, also known by Glance, is responsible to store and maintain the catalogue of virtual machine images used by the OpenStack compute project.

Glance has a REST API interface and relies in a distributed architecture where the components do not keep a state. For this reason Glance is extremely easy to scale out with the simple addition of new Glance services. Nonetheless, when doing so, the back-end store and database access should be carefully considered. For CERN Glance deployment we use a shared Ceph cluster located in Geneva as store. Since Glance is not affected by network latency, we can use this same back-end store also for the Hungary datacentre. However, a service interruption in Geneva will affect the creation of virtual machines in the Hungary as well.

Since we enable the OpenStack Telemetry service, the number of Glance APIs calls greatly increased. A large number of requests on the Glance APIs can affect the service response time. In order to decrease the risk of a degraded user experience and isolate Telemetry traffic we have dedicated some Glance APIs specifically for this use case.

5. Cinder

Cinder is the block storage service in OpenStack: it provides virtual block devices, so-called volumes, to instances. Volumes can be attached to an instance, then detached and re-attached to another instance. In addition to this flexibility, volumes are persistent, i.e. their lifetime is not tied to the lifetime of an instance, and they can come in different flavors, called volume types, which allow to specify backends with different quality of services (e.g. IOPS and bandwidth), technology (e.g. local vs. networked stoarge), or reliability (e.g. level of data replication or provided handling of power outages).

At CERN, we selected Ceph as the backend for Cinder due to its superior design with respect to reliability, scalablity (in terms of size) and future growth (in terms of use cases). After an initial evaluation in the first half of 2013, we procured and deployed a 3 PB Cluster (for Cinder and Glance, actually). While our Ceph cluster's PB-scale capacity is not challenged by our current usage with OpenStack, the combination of Cinder's and Ceph's versatility enable the scaling of the overall OpenStack service: we have introduced several volume types for standard usage, for high performance usage, and for cases in which data availability in case of a power failure is needed. These flavors enabled various services to move from physical hardware to virtualised servers.

6. Horizon

Horizon, the official OpenStack dashboard implementation, has been deployed at CERN since 2013, starting with Grizzly release. Initially, Horizon was configured in a physical replicated architecture. During last year, its structure was changed completely so we were able to run it completely on virtual machines, separating its components between different servers. At the same time, new features have been included in cloud dashboard service.

We configure the OpenStack dashboard in 2 layers: frontend and backend. The backend layer is formed by virtual machines hosting Horizon itself and machines running memcached. The frontend layer is composed of load balancers that distribute the traffic between all the various backend. This architecture is not difficult to implement since Horizon is just a Web client used to call the different OpenStack APIs. For the same reason, OpenStack dashboard is very easy to scale using CERN agile infrastructure tools.

Starting from 2013, we have implemented several new features to allow our customers to work in a more efficient, secure and easy way. The Web Single Sign-On was one of the most important features added to OpenStack workflow at CERN. Single Sign-On is present in most CERN applications providing facilities for B2B collaboration, improving users satisfaction and reducing IT help desk cost. For CERN private cloud users, this is a quality warranty.

We made also some important customization to the Horizon interface. We've decided to use RedHat theme and we've adapted it to CERN requirements. The image list panel was extended, trying to improve the user experience and the images management[3]. Finally, since CERN manages different volume types, we have added more information about the volume type quotas in the create volume workflow.

7. Ceilometer

Ceilometer is the metering component of OpenStack. It collects usage information about the various resources generated by the users, like instances, volumes and images, and stores them in a database.

Ceilometer collects samples in two different ways: notifications, sent by other OpenStack components when an action on a resource is triggered, and active polling of virtual machines usage. While notifications are lightweight, polling generates a lot of stress on the APIs due to the high amount of requests done. Figure 5 shows the increase of Nova API requests made by Ceilometer when we first enabled it in April 2014. The big amount of queries lead to a longer response time on the top level APIs. To avoid any impact on the other services and on the users, we introduced a parallel deployment of Keystone, Nova and Glance APIs, used exclusively by Ceilometer compute agents for active polling. Moreover, all the samples are forwarded to a dedicated RabbitMQ cluster instead of being queued in the central broker.

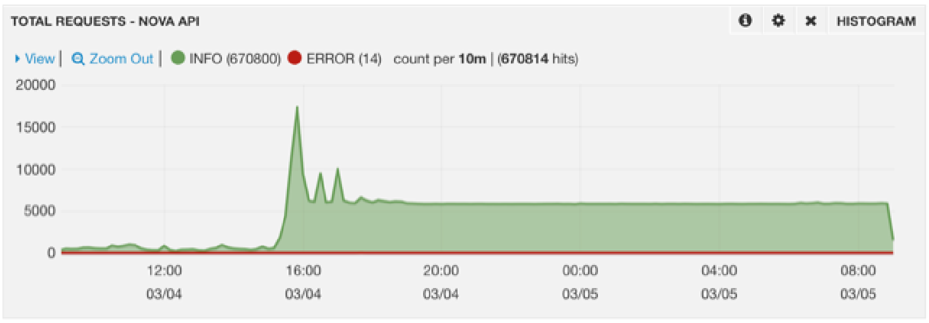

Figure 5. Ceilometer's number of requests on Nova API

We have two different usage scenarios for Ceilometer data: accounting and orchestration. Depending on the use case the samples are stored in different backend. For accounting, the main requirement is that all samples need to be preserved for 3 months. Initially a MongoDB cluster was deployed but, due to the high maintenance cost and poor performance, it was replaced with an HBase cluster. On the other hand, for the orchestration scenario, there is no need of store months of data but it iss important that the queries are executed very fast. For this scenario we deployed a very small MongoDB cluster in which the samples are stored in a capped collection. Since the data retained is very little, the database can live entirely in memory with a consequent increase in query performance.

8. Neutron

Neutron is a virtual network service for OpenStack. It provides an API to request and configure virtual networks that connect network interfaces from other OpenStack services (such as vNICs from Nova VMs). CERN cloud infrastructure is currently using nova-network to provide networking to instances. There is a medium term plan towards its deprecation in favor of Neutron and therefore a migration is desirable.

Extensive investigations have been carried out to define a suitable Neutron deployment model that takes into account our current computer centre network topology, on which Nova VMs are connected to pre-existing non-OpenStack managed virtual networks. Such a topology restricts the choice of the overlay technology for realizing software defined networks to those based on virtual point-to-point links.

CERN strategy is to first deploy Neutron with a minimum set of features and then add more incrementally. This year we have deployed a cluster on which interfaces are directly connected to a single provider network relying on Neutron subnets for IPAM over the above mentioned predefined virtual networks. It is running using the Modular Layer 2 implementation framework which allows us to simultaneously utilize the Linux Bridge and Hyper-V L2 agents on compute nodes. With this deployment we are able to provide the end-users with the security group functionality, which is not possible with nova-network due to the Nova Cells scalability model.

Since we are using a flat networking model and an external DHCP service, we do not need to operate physical replications of dedicated network nodes. This initially simplifies the scaling operations of Neutron and enables us to dynamically scale out Neutron services using a mixed physical / virtual architecture, analogously to how we scale other OpenStack services.

9. Operations

Managing a cloud infrastructure of thousand of nodes requires to automate common tasks and testing.

9.1. Rundeck

To automate tasks we are using Rundeck[4], a tool that lets us create jobs to cover some of our most common operations. Currently we are using Rundeck for interventions on hypervisors, creation of tenants, resource updates, health reports, and general user notifications. We also introduced some jobs that are meant to be used directly by the system administrators in case of hardware problems. Thanks to this, we managed to delegate internal sensitive tasks to other groups without exposing credentials or procedures.

9.2. Rally

A very important point is to ensure that OpenStack APIs are working correctly. For this purpose we selected Rally, a benchmarking tool for OpenStack that generates real workload on the APIs and measures the performance. This software can be used for CI/CD (Continuous Integration/Continuous Delivery) as well as for existing OpenStack cloud verification. We are currently running more than 10 benchmark scenarios in all the cells composing our cloud. The results are sent to Elasticsearch and can be visualized via Kibana dashboards (figure 6). With this tool we get a fast feedback in case of a bug, a mis-configuration of newly added cell, or errors following a component upgrade.

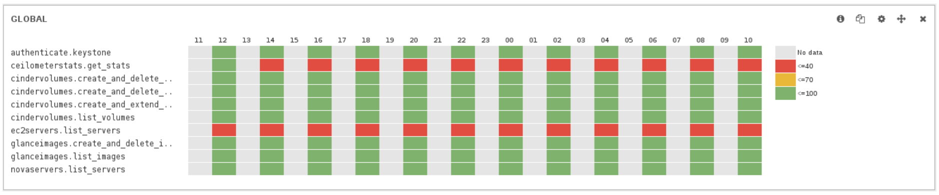

Figure 6. Rally dashboard for EC2, Glance, Cinder and Ceilometer APIs

9.3. Upgrade

One of the biggest challenges we are facing every six months is the upgrade of OpenStack to the latest release, following upstream release cycle. Since Grizzly we are following a component by component upgrade pattern. Every week we upgrade a different component, starting from the common ones, like Oslo, to normally finish with Nova. This patter is not fixed and we slightly changed the order depending on the requirements. The benefits of these approach is that in case of problem we know exactly where is the source. The only strong requirement is to ensure the version of the component we are releasing is compatible with the rest of the services. Also in this case we are using Rally to validate everything is working as expected.

Details about the executions of these upgrades can be found on our Openstack-in-Production blog[5][6][7].

9.4. Image management
The CERN private cloud offers to its users a set of public images that can be used as a base template for their virtual machines. Unfortunately, matching all people's use cases has been proved to be infeasible, leading us to define a standard environment that provides a good compromise between functionality and flexibility. Windows, SLC and CentOS images are provided on different flavours: different architectures, server or desktop oriented images, base installations or CERN-specific customizations. Having these images public to the users help them to speed-up the deployment of new services on the Cloud, but comes with an additional operational cost on our side. These images have to be updated regularly, preventing scenarios where the boot time of obsolete images takes too much time due to security updates. Currently, the lifetime of CERN private cloud provided images is 3 months.

The image life cycle is a process that is being improved continuously. This mainly work is focuses on optimizing the time spent by the cloud operators on the process, refining the image contents and providing new mechanism to the release cycle. This process has been partially automated via Jenkins.

First, a Jenkins job triggers the creation of all image flavours. This creation process relies on Oz[8] to create the guest operating system to produce the cloud image. Oz templates used on the CERN Cloud can be found in the cernops repository[9]. These jobs run on the Jenkins slaves that provide the required virtualization layer. The resulting image is uploaded to Glance and a series of tests are triggered. The tests instantiate VMs based on the newly created images and check different aspects: cloud-init, keypair injection, Kerberos authentication, IPv6 connectivity, and so on.

The image is then made public and released to the users. All these images are published with additional metadata. These extra information can be used by external applications or orchestration mechanisms to ensure the service defined is based on updated images. The metadata exposes architecture, operating system family, distribution, major and minor versions, edition, release date and the support contact. The Horizon dashboard has been extended with new functionality relying on this metadata[3].

Finally, when new images are released, the previous ones reach their end of life. Currently, the old images that are in use should not be deleted due to restrictions on OpenStack. If a used image is removed, operations like resize, migrate are affected. The solution , yet to be adopted, is to mark the old images as private and to make them visible only to users currently using them. This ensures new machines are based on newer images and while old ones can still be managed without problems.

10. Conclusions and outlook
The CERN private cloud provides Infrastructure-as-a-Service to a large variety of user communities. The use-cases of these communities include

- LHC experiment workloads
  - LHCb workloads submitted through Vcycle[10], scheduling CernVM Virtual Machines[11]
  - ATLAS Cloud scheduler[13]
  - CMS Tier-0 reconstruction workloads[14]
- CERN Batch scheduling[?]
- Service Nodes for LHC experiment and CERN services
  - many of these managed using toolset provided by the Agile Infrastructure project
  - License servers, hostel booking and other enterprise services
- Test and development nodes

  - Continuous Integration services of Grid and CERN IT customers
  - Development nodes of the Accelerator sector
  - Software Build nodes for various LHC experiments
- Personal workstations

Since its launch in Summer 2013, the CERN private cloud has grown rapidly in size and functionality. In the coming year, we will continue to add more capacity for physics data processing, and to add functionality to allow more users and more diverse applications to use the resources. In particular, we plan:

- to deploy Keystone Federation with partner clouds
- to roll-out Heat Orchestration to allow autoscaling and self-healing Platform-as-a-Service deployment
- to introduce support for Containers, based on LXC and/or Docker technologies

In addition, we will continue to upgrade our service infrastructure. During 2015 we will upgrade our KVM hypervisors to run the latest CentOS 7 version[12]. Also, we will continue to deploy performance improvements for I/O and CPU intensive workloads.

References

[1] Nova Cells V2, `http://docs.openstack.org/developer/nova/devref/cells.html`
[2] Identity Federation in OpenStack - an introduction to Hybrid Clouds, Marek Kamil Denis, CERN
[3] Choosing the right image, `http://openstack-in-production.blogspot.ch/2015/02/choosing-right-image.html`
[4] Rundeck - Job Scheduler and Runbook Automation, `http://rundeck.org/`
[5] Our cloud in Havana, `http://openstack-in-production.blogspot.ch/2014/02/our-cloud-in-havana.html`
[6] Our cloud in Icehouse, `http://openstack-in-production.blogspot.ch/2014/11/our-cloud-in-icehouse.html`
[7] Our cloud in Juno, `http://openstack-in-production.blogspot.ch/2015/05/our-cloud-in-juno.html`
[8] Oz project, `https://github.com/clalancette/oz/wiki`
[9] CERN Cloud images templates, `https://github.com/cernops/openstack-image-tools`
[10] Vcycle, a VM lifecycle manager for OpenStack etc, `https://github.com/vacproject/vcycle`
[11] The CernVM project, `http://cernvm.cern.ch/portal`
[12] CentOS Project, `http://www.centos.org`
[13] ATLAS Cloud R&D, `http://iopscience.iop.org/1742-6596/513/6/062037`
[14] The CMS Tier0 goes Cloud and Grid for LHC Run 2, `https://indico.cern.ch/event/304944/session/5/contribution/119/.../0.pdf`