PROCEEDINGS OF SCIENCE

# The “Cloud Area Padovana”: an OpenStack based IaaS for the INFN User Community

**Cristina Aiftimiei**
*INFN CNAF*
*Viale Berti Pichat 6/2, I-40127 Bologna, Italy*
*IFIN-HH*
*Str. Reactorului 30, Magurele, Romania*
*E-mail:* Cristina.Aiftimiei@cnaf.infn.it

**Paolo Andreetto**
*INFN, Sezione di Padova*
*Via Marzolo 8, I-35131 Padova, Italy*
*E-mail:* Paolo.Andreetto@pd.infn.it

**Sara Bertocco**
*INFN, Sezione di Padova*
*Via Marzolo 8, I-35131 Padova, Italy*
*E-mail:* Sara.Bertocco@pd.infn.it

**Massimo Biasotto**
*INFN, Lab. Naz. di Legnaro*
*Via Romea 4, 35020 Legnaro, Italy*
*E-mail:* Massimo.Biasotto@lnl.infn.it

**Fulvia Costa**
*INFN, Sezione di Padova*
*Via Marzolo 8, I-35131 Padova, Italy*
*E-mail:* Fulvia.Costa@pd.infn.it

**Alberto Crescente**
*INFN, Sezione di Padova*
*Via Marzolo 8, I-35131 Padova, Italy*
*E-mail:* Alberto.Crescente@pd.infn.it

**Alvise Dorigo**
*INFN, Sezione di Padova*
*Via Marzolo 8, I-35131 Padova, Italy*
*E-mail:* Alvise.Dorigo@pd.infn.it

**Sergio Fantinel**
*INFN, Lab. Naz. di Legnaro*
*Via Romea 4, 35020 Legnaro, Italy*
*E-mail:* Sergio.Fantinel@lnl.infn.it

**Federica Fanzago**
*INFN, Sezione di Padova*
*Via Marzolo 8, I-35131 Padova, Italy*
*E-mail:* Federica.Fanzago@pd.infn.it

**Eric Frizziero**
*INFN, Sezione di Padova*
*Via Marzolo 8, I-35131 Padova, Italy*
*E-mail:* Eric.Frizziero@pd.infn.it

**Michele Gulmini**
*INFN, Lab. Naz. di Legnaro*
*Via Romea 4, 35020 Legnaro, Italy*
*E-mail:* Michele.Gulmini@lnl.infn.it

**Michele Michelotto**
*INFN, Sezione di Padova*
*Via Marzolo 8, I-35131 Padova, Italy*
*E-mail:* Michele.Michelotto@pd.infn.it

**Massimo Sgaravatto**
*INFN, Sezione di Padova*
*Via Marzolo 8, I-35131 Padova, Italy*
*E-mail:* Massimo.Sgaravatto@pd.infn.it

**Sergio Traldi**
*INFN, Sezione di Padova*
*Via Marzolo 8, I-35131 Padova, Italy*
*E-mail:* Sergio.Traldi@pd.infn.it

PoS(ISGC2015)003

**Massimo Venaruzzo**

*INFN, Lab. Naz. di Legnaro*
*Via Romea 4, 35020 Legnaro, Italy*
*E-mail:* `Massimo.Venaruzzo@lnl.infn.it`

**Marco Verlato**[1]

*INFN, Sezione di Padova*
*Via Marzolo 8, I-35131 Padova, Italy*
*E-mail:* `Marco.Verlato@pd.infn.it`

**Lisa Zangrando**

*INFN, Sezione di Padova*
*Via Marzolo 8, I-35131 Padova, Italy*
*E-mail:* `Lisa.Zangrando@pd.infn.it`

At the end of 2013 INFN-Padova division and Legnaro National Laboratories (LNL) jointly started a new project aiming at expanding their grid-based computing and storage facility, mainly targeted to the needs of the big LHC experiments, with a cloud-based offering best suited to address the needs of the smaller sized physics experiments carried out by the local teams. Despite the great success of the grid model in supporting large scale HEP experiments, its adoption within smaller experiments has in fact been quite limited due to the well known lack of flexibility of the grid, e.g. in terms of: authentication/authorisation mechanisms; few OS and execution environments supported; batch-like only access to resources (no interactivity); deep learning curve for deploying/using services. This leads to a proliferation of several small computing clusters, disconnected from the grid infrastructure, owned by each experiment and fully dedicated to it, often underutilised but not powerful enough to satisfy peak usage needs concentrated in short periods (typically close to a scientific conference deadline). This scenario clearly implies low efficiency and large waste of both human and hardware resources for the data centre. The new cloud-based infrastructure is aimed at merging these scattered computing and storage resources in a unique facility that can serve the different experimental teams on-demand with the maximum of flexibility and elasticity made possible by the cloud paradigm. Leveraging the long-standing experience and collaboration as LHC Tier-2 of the Padova and LNL data centres, separated by 10 km but connected with a dedicated 10 Gbps optical link, the "Cloud Area Padovana" was built and put into production at the end of October 2014, after six months of pre-production operations while a couple of pilot experiments tested the capabilities of the infrastructure with real use-cases. OpenStack was chosen as Cloud Management Framework for implementing a IaaS where computing and storage resources were shared between the two data centres. However, several customisations and innovative services were added to the standard OpenStack deployment in order to address the users' needs, ensure system reliability and implement an efficient resource allocation. These concern the integration with OpenStack of authentication protocols like SAML and OpenID in order to enable user registration and access at first via INFN-AAI and later via Italian (IDEM) and possibly other international identity federations; the implementation of advanced functionalities for the management of users and projects by the IaaS administrators; the High Availability solution adopted to implement fault-tolerance of the cloud services; the development of a fair-share resource allocation mechanism analogous to the ones available in the batch system schedulers for maximizing the usage of shared resources among concurrent users/projects. An overall description of the cloud infrastructure and its operations will be given, together with the perspective of the main scientific applications running on it.

*International Symposium on Grids and Clouds (ISGC) 2015*
*15 -20 March 2015,*
*Academia Sinica, Taipei, Taiwan*

[1] Speaker

## 1. Introduction

The National Institute for Nuclear Physics (INFN) is the Italian research agency dedicated to the study of the fundamental constituents of matter and the laws that govern them, under the supervision of the Ministry of Education, Universities and Research (MIUR). It conducts theoretical and experimental research in the fields of sub-nuclear, nuclear and astroparticle physics, and has been developing all along in house open ICT innovative solutions for its own advanced needs of distributed computing and software applications. INFN carries out research activities at two complementary types of facilities: divisions and national laboratories. The four national laboratories, based in Catania, Frascati, Legnaro and Gran Sasso, house large equipment and infrastructures available for use by the national and international scientific community. Each of the 20 divisions and the 11 groups linked to the divisions or laboratories are based at different university physics departments and guarantee close collaboration between the INFN and the academic world.

In particular, the INFN-Padova division and Legnaro National Laboratories (LNL) are located 10 km from each other, but their data centres, connected with a dedicated 10 Gbps optical data link, have operated for many years as an LHC Tier-2 grid facility [1] for both the ALICE and CMS experiments. Thus, leveraging their long-standing collaborations, they recently started a new project aimed at providing a cloud-based offering of computing and storage resources with the goal of addressing not only the needs of the already supported LHC experiments, but also the needs of the numerous smaller sized physics experiments carried out by the local teams. The latter in fact could have until now only a limited support from the grid facility due to the well known lack of flexibility of the grid technology in terms of authentication/authorisation mechanisms, execution environments, access to resources and deep learning curve for deploying/using services. The project, called "Cloud Area Padovana", chose as Cloud Management Framework one of the most popular and industry supported open source solutions, OpenStack, that is also widely adopted in the scientific reference domain of INFN.

The paper describes the technical choices and the solutions adopted to provide a highly available production ready Infrastructure as a Service (IaaS) platform across the Padova and LNL data centres for the benefit of the INFN user community.

The paper is organized as follows. Section 2 puts the present work in the context of the scientific cloud infrastructures, comparing Cloud Area Padovana with the order of magnitudes larger size CERN Private Cloud. Section 3 shows the overall network architecture and services deployment layout, focusing on the High Availability solutions adopted to implement the fault-tolerance of the cloud services. Section 4 presents the advanced functionalities implemented for the management of users and projects to ensure an appropriate authorization work-flow after user authentication based on standard SAML protocol and federated identity management systems. Section 5 describes the original development of a fair-share cloud resource allocation mechanism analogous to the ones available in the batch system schedulers for maximizing the usage of shared resources among concurrent users and projects. Section 6 shows some examples of applications already running in the production infrastructure, and finally Section 7 provides a summary and the future perspectives.

## 2. Related Work

Cloud Area Padovana is clearly not the first and only project offering a cloud-based access to network, computing and storage resources geographically distributed in different data centres. The most relevant project for the scientific reference domain of INFN is certainly the CERN OpenStack Private Cloud [2]. In 2011 CERN's IT team decided to build a private cloud that would need to integrate well with their very heterogeneous environment. They investigated potential components for new infrastructure tools and processes during 2011, and reviewed a number of candidates. CERN's IT department selected OpenStack as Cloud Management Framework, and started working on it toward the end of 2011, building test clouds for physicists to explore cloud technologies and test integration with CERN specific customizations. Using Scientific Linux, developed by CERN and Fermilab based on the Red Hat distribution, a cloud was rapidly built with Compute, Image, Identity and Dashboard services. OpenStack was chosen since it was recognised to be the fastest growing open cloud community, working to build software that powers public and private clouds for a growing number of organizations, including Cisco WebEx, Comcast, eBay, HP, Intel, MercadoLibre, NeCTAR and Rackspace. Having a vibrant technology and developer open ecosystem around OpenStack allows to benefit from the work of the active contributors but also to use the local engineering skills to enhance the product for others.

At the time of writing, CERN Private Cloud spans two data centres (CERN and Wigner, in Hungary) connected by two 100 Gbps links, hosts more than 4700 hypervisors (120000 cores), 11000 VMs and provides services to 1500 users and 1800 projects.

While CERN's experience was of great inspiration for the building of the Cloud Area Padovana, some architectural design and some technical solutions choices were different:

**Resource management:**

- CERN Cloud uses OpenStack Nova Cells in their architecture to easily manage the expansion of the cloud while keeping a single entry point for all the resources. The Cloud Area Padovana is much smaller, and therefore it was chosen not to partition resources in Nova Cells, because this was considered not needed.

- CERN Cloud does not support live migration of instances between different compute nodes, since there is not a common file system among compute nodes and because of some constraints in the CERN network infrastructure. This is not the case for the Cloud Area Padovana, where the live migration of instances was considered an important feature to be implemented.

- CERN Cloud uses Ceph as storage backend for Image service (Glance) and Block Storage service (Cinder). Cloud Area Padova chose instead GlusterFS as storage backend, given the greater experience of the local engineering skills with this product. GlusterFS is also used for the file system hosting the instances.

- CERN Cloud uses Flume, Elastic Search and Kibana for monitoring the OpenStack status in all nodes. Cloud Area Padova chose instead to use Nagios and Ganglia as monitoring tools, given the long-standing in house experience with these products, inherited from the grid era.

**Networking:**

- CERN Cloud still uses the deprecated Nova Network to integrate OpenStack with the network infrastructure at CERN. Instances created in the CERN OpenStack are created in CERN public networks. Cloud Area Padovana chose instead to use the Neutron service with Open vSwitch and GRE tunneling for managing the networking of the whole cloud infrastructure, as described in section 3.2.

**Authorization/Authentication**

- In CERN Cloud the Keystone Identity service is integrated with the CERN SSO system by using the LDAP backend. Cloud Area Padovana chose instead to develop some extensions to both Keystone and Horizon Dashboard to allow OpenStack to interact with the SAML-based INFN-AAI Identity Provider, and then with the IDEM Italian Federation. Moreover, complete separation of the authentication phase, relying on external IdPs, from authorization phase taking place inside Keystone was implemented, as described in section 4.

On the other hand, some other choices were quite similar, e.g. the use of a configuration infrastructure based on Puppet, the use of EPEL/RDO – RPM packages to install OpenStack, the use of HAProxy as load balancer in a High Availability configuration and of a RabbitMQ cluster for the messaging queue.

## 3. Architecture of the Cloud Area Padovana

Figure 1 below describes the layout of the Cloud Area Padovana. OpenStack services were deployed on hosts located in Padova, while the Compute nodes (where cloud virtual machines are hosted) as well as storage servers were deployed both in Padova and in Legnaro. The two sites are connected through a 10 Gbps fibre link. The details will be discussed in the next subsections.

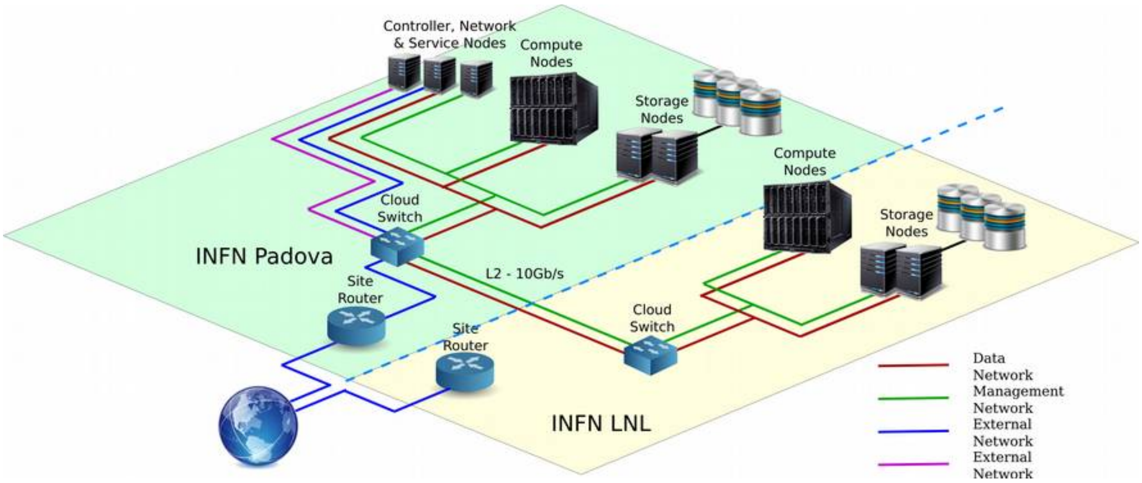


**Figure 1: Cloud Area Padovana layout**

### 3.1. Hardware capacity

Concerning the hardware, in Padova a DELL Blade based solution (DELL M1000e enclosure with 2 switches DELL Force 10 MXL) was chosen. The compute nodes in Legnaro are instead 6 servers Fujitsu Primergy RX300S7. The hardware specifications and the OpenStack role of the servers are listed in Table 1 below.

| Location | # of servers | CPU Model | # of CPUs | RAM (GB) | # of CPU-cores | OpenStack role |
|---|---|---|---|---|---|---|
| PD | 4 | E5-2609 | 2 | 32 | 8 | Controller, Network nodes |
| PD | 5 | E5-2670v2 | 2 | 96 | 40 | Compute nodes |
| PD | 3 | E5-2650v3 | 2 | 96 | 40 | Compute nodes |
| LNL | 6 | E5-2650v2 | 2 | 96 | 32 | Compute nodes |
| **Total** | **18** | | **36** | **1472** | **544** | |

**Table 1: Cloud Area Padovana initial hardware specifications**

Concerning the storage, in Padova a iSCSI DELL MD3620i server solution was chosen. At the moment it includes 23x900GB SAS disks. A storage expansion MD1200 with 12x4TB disks is being integrated, reaching a total of 68.7 TB. This storage system has been configured using GlusterFS, and is used for the cloud images (Glance service), for the storage of the virtual machines (Nova service) and for the block storage available to the cloud instances (Cinder service). In Legnaro a Fibre Channel based system (DELL PowerVault MD3600F plus MD1200 expansion) with 24x2TB disks was acquired instead. It will be used as general-purpose user storage, also configured using GlusterFS.

### 3.2. Network layout

The cloud network configuration is based on three VLANs, each one associated to a class C network. The VLANs are partially shared between the local INFN-Padova division network and the local Legnaro National Laboratories network. The two sites are connected through a dedicated 10 Gbps Ethernet optical data link. The three networks are associated, in the OpenStack terminology, to the management, data/tunnel and public/external networks. The management network connects all the servers hosting OpenStack services, plus a number of ancillary services that will be described in the next sections and are needed to manage, control and monitor the infrastructure and ensure its high availability. The management network uses private IP addresses, but can access the Internet through a dedicated NAT server. The data network is used for virtual machines communications and connects the OpenStack Compute and Network nodes. It supports a 10 Gbps link and MTU > 1500 to allow GRE (Generic Routing Encapsulation) [3] tunnels usage. Lastly, the public network connects the OpenStack services that need to be reached from outside the LAN, i.e. the Horizon Dashboard, the Nova and EC2 APIs and the Network nodes. In order to manage the internal cloud networks the use of OpenStack Neutron service with Open vSwitch and GRE was chosen. Two Neutron provider routers were defined: a first one for virtual machines that need to be reachable from outside; a second one for virtual machines that only need SSH access from internal INFN-Padova or LNL LANs. The first router has the external gateway on the public network, and connects the class C subnets of the 10.63 network. Each OpenStack project uses a separate subnet. Virtual machines obtain their private IP addresses from internal DHCP servers, and can access the Internet through the NAT provided by Neutron, or obtain a floating IP from the public network in case they need to offer services to the external world. The second router connects the class C subnets of the 10.64 network, but has the external gateway towards the internal LANs. In this way the NAT is not used so that the virtual machines can be reached from the INFN-Padova and LNL internal LANs directly by using their private IP addresses. For security reasons we chose to allow only connections from the LAN towards the virtual machines, and not vice-versa. These virtual machines are at any rate allowed to access Internet through an external NAT, even if they cannot obtain a floating IP address. A special configuration was designed in order to provide virtual machines with high performance access to data located in storage systems external to the cloud. The L2 Neutron Agent software was installed on the storage servers, which were equipped with two network interfaces: one on the management network and the other one on the data network. Once all services start, the network bridges and GRE tunnels towards all the components of the cloud infrastructure are automatically created. The only manual operation needed is to add the new traffic flow to the local OpenFlow table. As this table is restored when a new Compute node is added to the cloud, a cron job may be necessary to execute a script to keep it updated by adding the storage flow, if needed. The virtual machine access is possible by creating an interface on the interconnection bridge, and by assigning to it the proper VLAN and an IP address on the same subnet of the virtual machine that need to be reached. Access to data can be obtained e.g. via NFS or other protocols.

### 3.3. High Availability service deployment layout

The business continuity is a very sensitive topic when services are exposed to users and determines their quality evaluation of the provider. In the case of OpenStack, the operation continuity is based on the redundancy of each physical node dedicated to the execution of the stack's components. The OpenStack guide suggests two ways of setting up its services in a highly available mode:

- **active/passive** mode: the services are running on only one physical node of the cluster at any given time, but they can be moved from one node to another
- **active/active** mode: the services run at the same time on all nodes of the redundant cluster

Even if there are cases in which distributed services can keep their internal states synchronized and can be set up in high availability mode with both models, the high availability of stateful services is more achievable with the active/passive mode, in order to prevent data corruption due to race condition. On the other hand OpenStack components are stateless (they store their data on an external SQL database), as a consequence the active/active mode is also suitable for it. In our cloud farm we also considered the benefit of load balancing offered by an active/active model using the lightweight, high performance and easy configurable HAProxy [4] daemon for the balancing of the incoming connections among the physical nodes, and Keepalived [5] for the Virtual IP movement between the HAProxy servers.

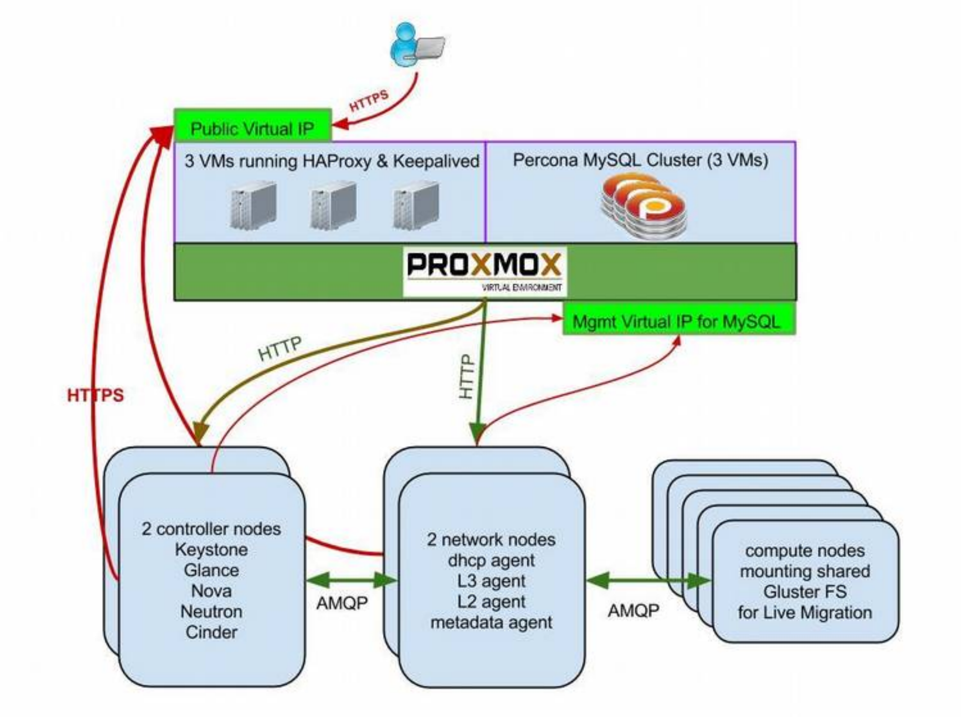


**Figure 2: High Availability configuration**

The three nodes running HAProxy and Keepalived are virtual machines running Linux CentOS 6.6. HAProxy and Keepalived are not "stock" packages from EPEL; they were packaged from recent stable releases that support respectively SSL front-end and unicast protocol. The three virtual nodes are backed by the Proxmox virtualization software facility and are running on different bare metal machines, in order to avoid a dumb scenario in which a single hardware failure would bring down the entire cluster. However, thanks to the live migration, the Proxmox cluster would be able to automatically evacuate virtual machines from failing hardware, and this represents an additional level of safety. The choice of three virtual nodes is dictated by the need of a quorum with a minimum resource usage, which results in at least three machines running the Keepalived daemon. Keepalived is configured to migrate the Virtual IP upon any virtual node failure (networking loss, operating system crash, intentional reboot) or upon HAProxy failure, due to a crash or a planned switch off for reconfiguration. The HAProxy/Keepalived virtual cluster balances the incoming connections towards the MySQL Percona XtraDB cluster that is in turn composed of three virtual instances. These instances are backed by the same Proxmox-controlled physical infrastructure, and reside on different bare metal machines. The configuration of the database cluster is in multi-master (active/active): all three Percona instances can receive queries, and keep the three local databases synchronized by means of the Galera libraries. Figure 2 above shows the High Availability configuration described here. The OpenStack components running on the controller nodes, and made redundant and balanced by means of HAProxy are: Keystone, Glance, Nova-API, Nova-NoVNCProxy, Neutron server and Cinder-API. Note that there are other components not balanced by HAProxy: Nova conductor, consoleauth, cert, scheduler and Cinder scheduler. Even if not controlled by HAProxy, their redundancy is also critical in order to make the cloud IaaS working properly. They communicate with each other and all the other OpenStack components through the AMQP daemon running in the controller nodes. For the AMQP implementation we chose RabbitMQ that was, more than one year ago, the only one offering a robust and easy configurable high availability mode. HAProxy exposes SSL interfaces; in fact it is configured to act like an SSL-terminator. Then, the incoming connections (from clients and OpenStack services themselves) are encrypted and directed to HTTPS URIs. HAProxy is responsible for extracting the plain payload and forwarding it to the OpenStack APIs which listen on a plain HTTP protocol. Each OpenStack component "believes" that all the others listen on HTTPS because all the endpoint's URIs contain the public Virtual IP (which sits on one HAProxy node) and specify "https://" as communication protocol. Then, each component actually listens in plain HTTP as server, but talks in HTTPS as client via the SSL termination. Thanks to this tricky configuration, we avoided a more complex one having the API running inside the Apache daemon, and we encapsulated all the SSL business logic in one place only: the HAProxy's configuration file. This method also improves the performance because at least the decrypting overhead is performed by a dedicated node, which is not part of the OpenStack IaaS. Concerning the networking, there are the Neutron's agents (dhcp, L3, metadata, openvswitch) redundantly running on two physical network nodes. All of them, with the exception of L3, can run in active/active mode because they are stateless and communicate by means of the highly available AMQP daemon. If L3 becomes unavailable, a monitor daemon, running on a Nagios server, can migrate the virtual Neutron routers it was handling to the other L3 instance running on the other network node.

In addition to the Cinder API and scheduler, there are two instances of Cinder Volume running on two physical hosts which manages the shared storage space aggregated by GlusterFS. This space is not only dedicated to the Volume service, but also to the Glance's images and the Compute's instances. The Cinder volume daemons are, in this case also, redundant and can both be addressed by the Cinder scheduler. Finally, all the Compute nodes (hypervisors) run their local instance of Nova compute process, which communicates with the Controller's services by means of the AMQP. All the hypervisors mount the shared GlusterFS file system in their /var/lib/nova/instances and allow the system to perform a live migration of the virtual machines. In this way the system is tolerant to scheduled outages of the hypervisors for maintenance or upgrade of the software.

### 3.4. Infrastructure management and monitoring

#### 3.4.1. Infrastructure management

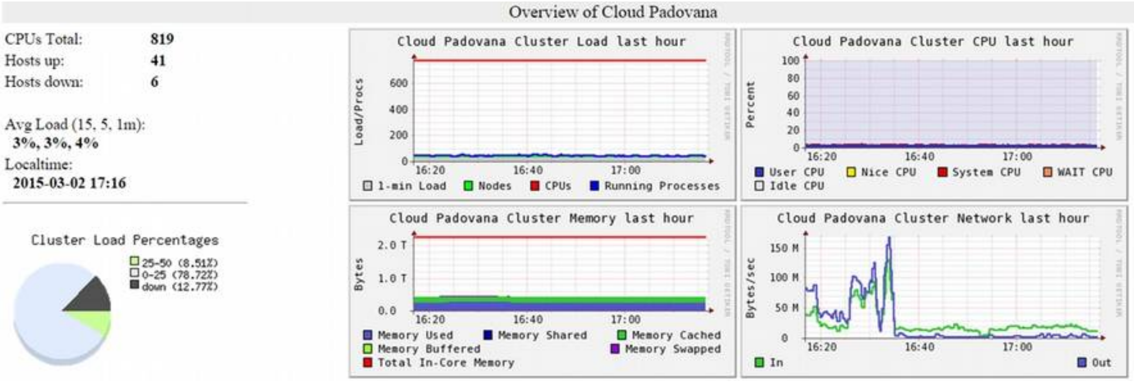
A critical part of a cloud's scalability is the amount of effort that it takes to run the cloud infrastructure. To minimize the operational cost of running our infrastructure, we analysed, set up and used an automated deployment and configuration infrastructure with a configuration management system: Foreman together with Puppet. Combined, these systems greatly reduce manual effort and the chance of operator mistakes. Foreman is a system that automatically installs the operating system’s initial configuration and later coordinates centrally the configuration of all services. The hardware used for the cloud infrastructure is equipped with iDRAC (Integrated Dell™ Remote Access Controller Firmware), iLO (HP Integrated Lights-Out) or OpenManage interface so we can remotely operate on it and easily check the hardware status. The Foreman instance is responsible for provisioning hosts via PXE and also acts as Puppet Server managing all nodes configurations, deployed on a single host. In the Foreman host the following services are installed:

- the TFTP boot server
- the DHCP Server for the cloud management and data subnets
- the Puppet Master

Using Foreman different hosts can be built by installing them with different operating systems choosing the provisioning template and setting the partition table as needed. Using customized Puppet classes, from the Web GUI a set of Puppet modules can be associated to each host. Our infrastructure is mostly based on Red Hat Enterprise Linux derivatives like Scientific Linux and CentOS operating systems, but the possibility of deploying services using Ubuntu is also guaranteed. For each operating system we set up a custom set of kickstart or preseed files which manage the packages installation, the partition table, the network interfaces, linux base configurations like selinux and iptables, some customizations to Puppet auth and conf files. We defined hostgroups by assigning certain Puppet modules responsible with the configuration of: ntp, ssl-key access, Ganglia and Nagios monitoring agents and plugins. All hosts are associated to respective hostgroups according to their roles in the infrastructure. The OpenStack Controller and the OpenStack Network node are installed with a manual procedure, while all the OpenStack Compute nodes have been installed using a home-made, custom Puppet module. In this way we manage all the hosts of our cloud infrastructure from one point: the Foreman Web GUI.

#### 3.4.2. Monitoring

The servers hosting the storage system, the OpenStack services and the MySQL DB, HAProxy and RabbitMQ clusters are monitored by Ganglia and Nagios. Using custom Puppet modules the gmond clients are deployed on each host, and configured to send data to gmetad server in order to monitor: the CPU and memory usage, disk space, GlusterFS volume allocation, network performance.



Overview of Cloud Padovana

CPUs Total: 819
Hosts up: 41
Hosts down: 6

Avg Load (15, 5, 1m):
3%, 3%, 4%
Localtime:
2015-03-02 17:16

**Figure 3: Ganglia monitoring service**

A Nagios server, which collects data from most of OpenStack nodes, is also operating. We created a simple Puppet module to deploy the Nagios client together with the Nagios server SSH key on the nodes and also developed some Nagios plugins to check not only the OpenStack services like Keystone, Nova, Glance, Neutron and Cinder, but also the other useful services that allow the infrastructure to work like GlusterFS, OpenVSwitch, HAProxy, Keepalived, MySQL, RabbitMQ. If there is a problem with these services an e-mail is sent to the cloud support team. In some particular cases the Nagios plugins try to restart the service and check the sanity of the processes afterwards. Some specific OpenStack Nagios checks have also been implemented, e.g.:

- a check that periodically flushes the Keystone tokens from the database;
- a check that creates each hour a virtual machine on each hypervisor checking its network connectivity.

## 4. Identity and Access Management

In the recent years the need for technologies supporting features like single sign on, identity federation and attribute publishing led to a wide range of identity management systems. The Security Assertion Markup Language, SAML, is one of the most relevant technologies in this field. It has gained popularity among many research institutes and universities all over the world. The OpenId project represents another example of a standard for identity management, widely adopted in the area of the social networking. One of the goals of the Cloud Area Padovana project consists on integrating the most common identity management systems into the OpenStack infrastructure. The current version of the delivered software provides extensions to the OpenStack Identity service, Keystone, and web portal, Horizon, for dealing with both SAML and OpenId. The first identity management system working with the Cloud Area Padovana is the "INFN Authentication and Authorization Infrastructure", INFN-AAI. INFN-AAI makes available to various INFN services a SAML compliant identity provider. The identity provider is able to authenticate all the users registered in the centralized accounting system, GODIVA, using different credentials such as simple usernames and passwords, Kerberos tickets or X509 certificates. The result of authentication phase is stored into a SAML authentication assertion which is returned to the service. The service uses the authentication assertion to identify the user. For any authenticated user the identity provider can publish a set of attributes retrieved from the accounting system. The composition of the attribute set depends on the privileges granted to the service that contacts the identity provider. The published attributes can feed both the authorization phase, carrying out an attribute-based access control strategy, and the business logic of the service. In the context of Cloud Area Padovana the service interacting with the identity provider of INFN-AAI is the OpenStack web portal: the Horizon Dashboard. The following subsections describe in more detail how the authentication and authorization phases take place in Horizon.

### 4.1. User authentication and authorization

Inside the OpenStack infrastructure the key role for the identity management is carried out by the identity service: Keystone. All the services that compose the infrastructure, such as Nova or Neutron, rely on the token-based Keystone protocol for identifying the user and controlling the access to any resource. In this situation a gateway is necessary in order to transfer the outcome of the authentication phase occurred in the web portal to the identity service. This is made possible by a couple of extensions that exchange a special custom token, carrying all the information required to identify the user. The first extension is deployed into the web portal while the second one is placed in the identity service. The custom token is forged using a secret shared between the identity service and the web portal and its internal structure is not bound to any specific standard. In this way the gateway can interact with any identity management system supported, in this case both SAML and OpenId.

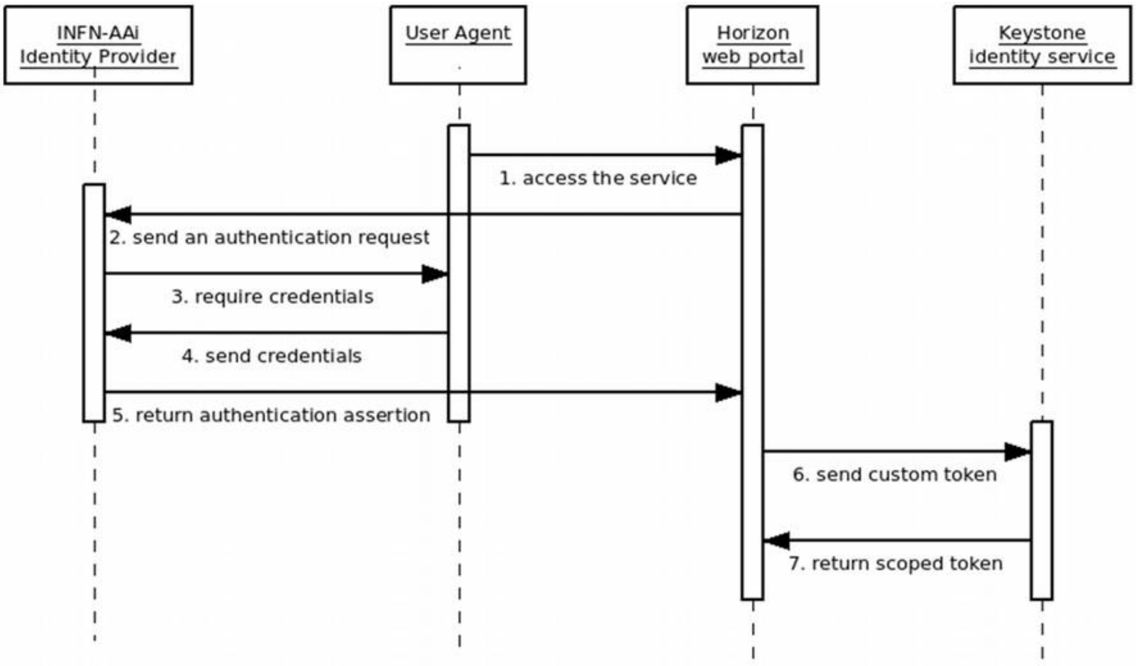


**Figure 4: Authentication and authorisation flow**

Figure 4 above summarizes the operations that occur during the complete authentication phase. The steps from 1 to 5 are the standard ones specified by the Web Browser Single Sign On profile of SAML. The step 6 represents the custom token exchange between the two extensions developed in the Cloud Area Padovana project. With step 7 the identity service, Keystone, once the extension has processed the custom token, returns the Keystone scoped token. The scoped token, according to the Keystone protocol, encompasses any information for accessing an OpenStack service as a member of a project. The key point of the solution developed in Cloud Area Padovana is the complete separation of the authentication phase, which resorts to external identity management systems, from the authorization phase which takes place inside the OpenStack Identity service.

### 4.2. User registration

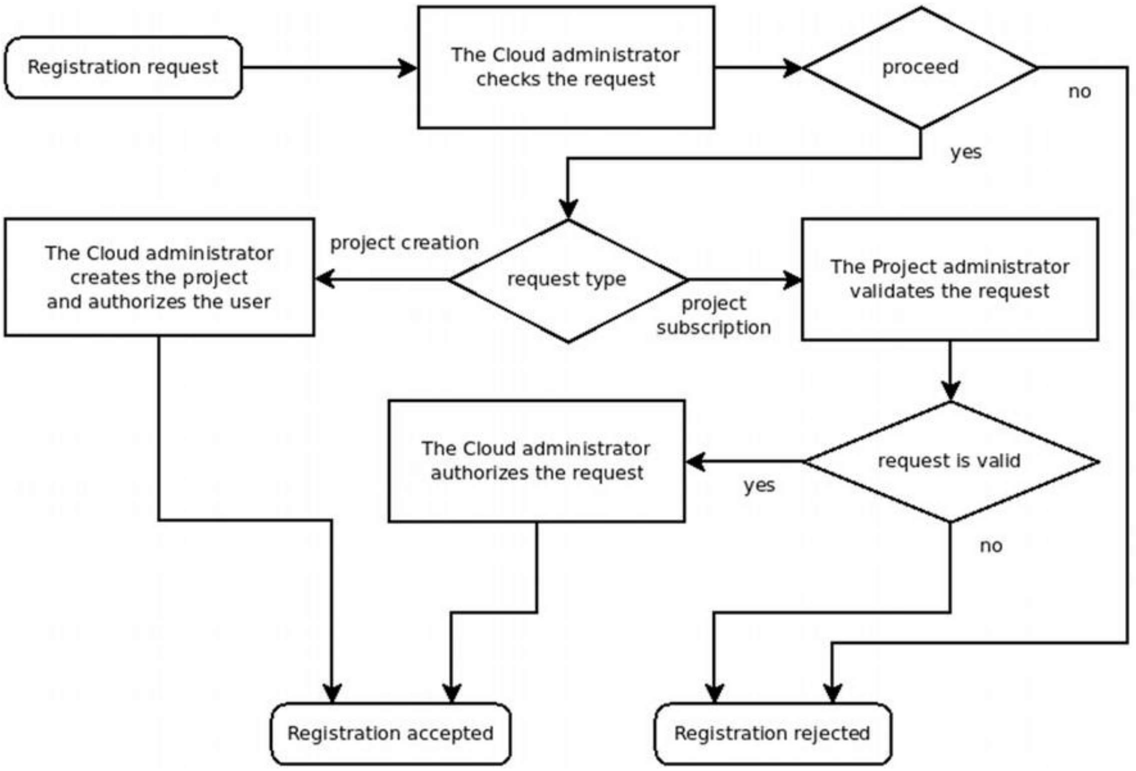
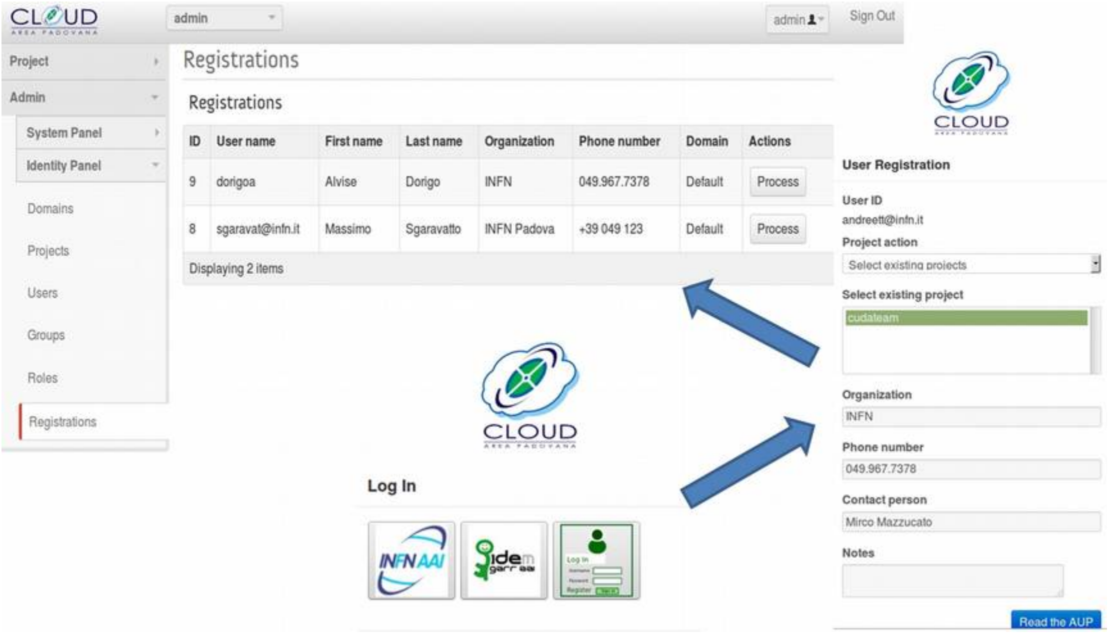

Registration request
The Cloud administrator checks the request
proceed
no
yes
The Cloud administrator creates the project and authorizes the user
project creation
request type
project subscription
The Project administrator validates the request
The Cloud administrator authorizes the request
yes
request is valid
no
Registration accepted
Registration rejected

**Figure 5: Registration flowchart**

Access and privileges can be granted to a user if, and only if, the user is a member of at least one active project in the infrastructure. The project membership is information that cannot be gathered from the attributes returned by the INFN-AAI Identity Provider. For those users who successfully pass the authentication step but do not have any project associated with them, a request for registration is necessary. In this case identified users must subscribe to one or more existing projects, or alternatively require the creation of a new one. The information retrieved during the authentication phase together with the list of projects to subscribe to or to be created represents a request for registration. Requests for registration must traverse a registration work-flow in order to be approved or rejected. The registration work-flow involves different subjects, each one with different roles and capabilities. In the current implementation two categories of actors are defined: the cloud administrator and the project administrator. The cloud administrator is the native role provided by any OpenStack installation that manages the cloud. It has all the privileges of users and projects and it is responsible, eventually, for accepting or rejecting the request for registration The project administrator is the user of the project that is responsible for validating any new request for registration. Since a cloud administrator is not supposed to know if a new user is allowed to be a member of an existing project, the project administrator is the reference required to solve this issue.

The complete registration work-flow is depicted in Figure 5 above. The very first step in the flow is carried out by the cloud administrator who performs a preliminary check of the request and sets several constraints such as the expiration date of the account. The following steps depend on the type of registration request. If it refers to a creation of a new project the registration flow involves only the cloud administrator, who creates the project, grants access and privileges to the user and declares the user as the first project administrator for the new project. In case the request refers to a subscription to an existing project, the project administrator is committed to validate the membership of the user before the cloud administrator concludes the registration forms. If the user is not recognized by the project administrator the request is rejected. When eventually the cloud administrator authorizes the user, privileges and project membership are stored onto the OpenStack Identity service. The entire registration flow is implemented in the OpenStack Horizon Dashboard with the following elements:

- a user registration form by which the user can specify the projects to subscribe to or to be created, together with other useful information
- a panel for the registration request management, visible only by the cloud administrator, and a set of pop-up menus dealing with different step of the flow
- a panel for the project subscriptions by which the project administrator can handle the validation of the registration request



Registrations

Registrations

| ID | User name | First name | Last name | Organization | Phone number | Domain | Actions |
|---|---|---|---|---|---|---|---|
| 9 | dorigoa | Alvise | Dorigo | INFN | 049.967.7378 | Default | Process |
| 8 | sgaravat@infn.it | Massimo | Sgaravatto | INFN Padova | +39 049 123 | Default | Process |

Displaying 2 items

User Registration

User ID
andreett@infn.it

Project action
Select existing projects

Select existing project
cudateam

Organization
INFN

Phone number
049.967.7378

Contact person
Mirco Mazzucato

Notes

**Figure 6: Horizon customisation enabling registration**

The elements above are registered into the web portal exploiting the extension mechanisms of the framework. No changes in the code of the original Horizon Dashboard are required. A snapshot of the new form and panels is shown in Figure 6 above.

## 5. Fair-share scheduler

### 5.1. The resource static partitioning issue

Computing activities performed by user groups in the Public Research and in the Public Administrations are usually not constant over long periods of time (e.g. one year). The amount of computing resources effectively used by such user teams may therefore vary in a quite significant way. On these environments it is usual for the teams to stipulate with the Data Centres contracts for the provision of an average computing capacity to be guaranteed during a long period (e.g. one year) rather than of an agreed amount of computing resources that should be available at any given time. In these Data Centres new hardware resources are acquired according to the user best estimates of the annual computing resources needed for their activities and partitioned among them. The partitioning policy is defined in terms of fractions of average usage of the total available capacity (i.e. the percentage of the Data Centre's computing resources each team has the right to use averaging over a fixed time window). In order to respect the contracts, the administrators have to enforce that each stakeholder team, at the end of any sufficiently long period of time, and hence for the entire year, has its agreed average number of resources. Moreover, since in general the request for resources is much greater than the amount of the available resources, it becomes necessary to seek to maximize their utilization by adopting a proper resource sharing model. In the current OpenStack model, the resource allocation to the user teams (i.e. the projects) can be done only by granting fixed quotas. Such an amount of resources cannot be exceeded by one group even if there are unused resources allocated to other groups. So, in a scenario of full resource usage for a specific project, new requests are simply rejected. The static resource allocation model fits the economic model well where users can benefit at most of the resources provided for in their contract but is not suitable in the public research context. Past experience has shown that, when resources are statically partitioned (e.g. via quota) among user teams, the global efficiency in the Data Centre's resource usage is usually quite low (often less than 50%). This resource allocation optimization problem has already been tackled and solved in the past initially by the batch systems and subsequently by the grid model which was conceived on the awareness that Research Data Centre resources are limited. In particular such advanced batch systems, in opposition to the OpenStack static model, adopt the dynamic allocation paradigm which allows a continuous full utilization of all available resources. This model is based on sophisticated scheduling algorithms (i.e. fair-share) in order to guarantee the resources' usage is equally distributed among users and groups by considering the portion of the resources allocated to them (i.e. share) and the resources already consumed. INFN has started to address this issue by enhancing the current OpenStack scheduler capabilities by providing the main batch system's features as a persistent queuing mechanism for handling user requests as well as the adoption of the resources provisioning model based on the advanced SLURM's Multifactor Priority strategy [6].

### 5.2. The fair-share scheduling solution

The OpenStack architecture is composed of a set of components each one having specific capabilities. Nova scheduler is the component that determines the proper and available host which best satisfies the user request for instantiating his virtual machine. Although Nova scheduler provides scheduling features, it is far from the real batch systems logic. It is just a resource supplier, which processes the user requests sequentially (FIFO scheduling) and in case a request cannot be satisfied due to non-availability of hosts or the assigned user quota is fully used, such requests will fail immediately and it will be lost. What is mainly missing in Nova scheduler is a queuing mechanism for handling the user requests and a fair-share algorithm in the resources provisioning which can guarantee at the same time the continuous full usage of all resources and the quota established for the different user teams. In the batch system terminology, the fair-share is one of the main factors in the job priority formula which influences the scheduling order of the queued jobs and its calculation is based on the historical resource utilization of all users. The approach we adopted is to fill the gap between Nova scheduler and the batch systems, by implementing a new component named FairShare-Scheduler which aims not to substitute the original scheduler, but to provide the missing advanced scheduling logic without breaking the OpenStack's architecture. In particular the FairShare-Scheduler has been designed in order to be fully integrated in OpenStack and especially to not require any changes on the existing scheduler. The schema in Figure 7 shows the high level architecture and in particular highlights the interaction with other components involved in the scheduling activity.

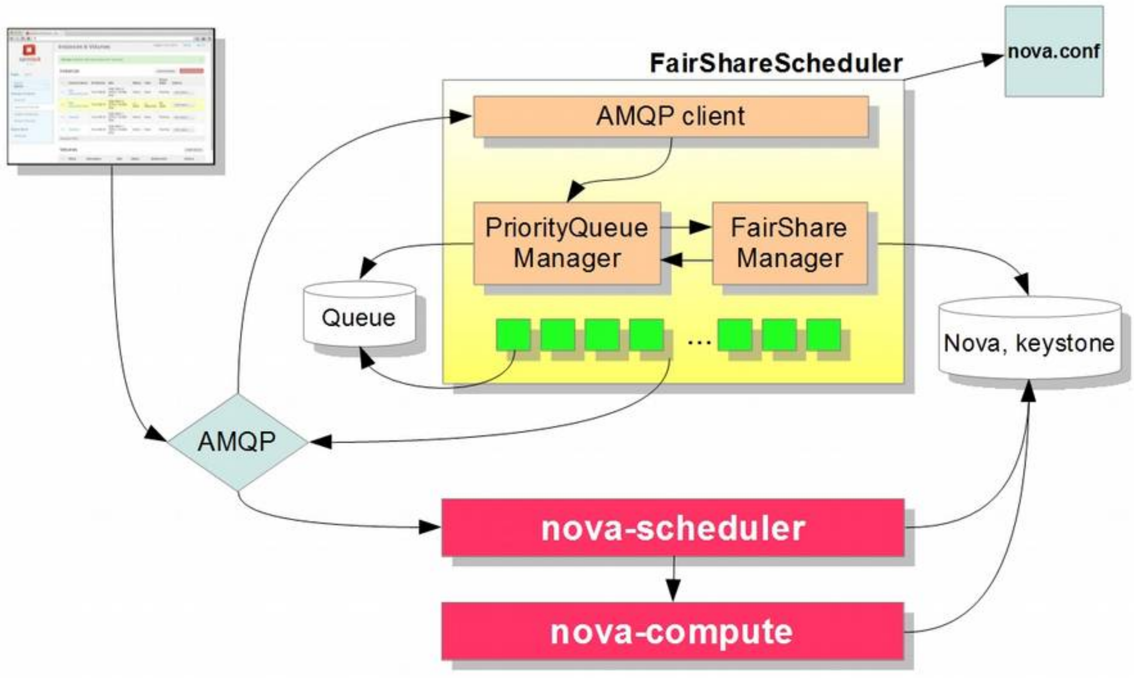


**Figure 7: FairShare scheduler architecture**

A proper priority value, calculated by the FairShare-Manager, is assigned to every user request coming from the Horizon Dashboard or command line. The request is immediately inserted in a persistent priority queue by the PriorityQueue-Manager. A pool of Workers fetches the requests having higher priority from the queue and sends them in parallel to the Nova scheduler through the AMQP messaging system. In the scenario of full resource utilization, the pool waits until the compute resources become again available. In case of failure, the FairShare-Scheduler provides a retrying mechanism, which handles the failed requests by inserting them again into the queue for n-times. Moreover, to prevent any possible interaction issue with the OpenStack clients, no new states have been added and from the client point of view the queued requests remain in “Scheduling” state till the compute resources are available. Finally the priority of the queued requests is periodically recalculated to give an opportunity to the older requests to rise up in the queue.

Currently the FairShare-Scheduler is a prototype ready for Havana and IceHouse versions and tests are in progress in the cloud test-beds of INFN-Padova and University of Victoria (Canada).

## 6. Application examples

### 6.1. LHC Experiments

The main LHC users of the Tier-2 Legnaro-Padova centre are ALICE [7] and CMS [8], that have been supported to run interactive and distributed processes for a long time. So they were the natural candidates to test and evaluate the new cloud infrastructure seen as the evolution of the grid concept. Even if the motivation to use the distributed computing infrastructure of grid was the same for both of them (huge amount of data to store in a secure way, high computing power to run analysis and event productions) they started to evaluate the Cloud Area Padovana with two different use-cases.

#### 6.1.1. ALICE

For the ALICE team of Padova the main use-case is to “translate” the analysis facility, a system to create and distribute analysis jobs to dedicated resources, to a Virtual Analysis Facility (VAF) based on virtual machines. The idea is the creation of an elastic cluster for the interactive analysis. Resources are not preassigned and static but allocated and released in dynamic way taking into account the workload and the single user needs. The framework for the simulation, reconstruction and analysis in ALICE is ROOT [9]. Generally in HEP the analysis of an event of a single collision of particles is independent from other events so the process can be parallelized. ALICE is using the PROOF facility to split input data across distributed and independent ROOT sessions run interactively and in parallel. These jobs are designed to be short, around 20 min. The possibility of having resources ready and configurable on demand to run these kind of jobs and to release them when not needed is the feature provided by the cloud. The user's credential for the VAF is the grid certificate. The user builds the cluster asking for resources with a simple shell command, releasing them when the work is done. Possible failures in the single user cluster do not propagate to the other user's clusters. One head node plus a scalable number of workers compose the VAF. The CernVM tool was adopted as the graphic interface for the creation and configuration of the VAF. The experiment code is accessible via CernVM-FS [10], a system developed by CERN for automatic code distribution. The workload management system is done with HTCondor. In the preproduction cloud, only ephemeral disks were available for the VAF. Here the set-up and complete flow was successfully tested with simple pilot analysis: access and authentication of the VAF, request of the virtual machines, analysis and results merging. The next step for the production cloud is the installation in Legnaro of 70 TB of external storage accessible from the virtual machines via xrootd protocol.

### 6.1.2. CMS

For the CMS team of Padova the first idea to evaluate the cloud potentiality is to reproduce on the cloud the local facility mainly used for interactive processes. It is based on a limited number of machines with different number of processors and RAM, available to authorized users to run analysis and production processes in an interactive way. Generally input data is read from the dCache grid storage located in Legnaro and results are written to a storage server located in Padova and running Lustre file system. The storage hosts the user's home too, imported from all the machines of facility. The code of the experiment used to run processes is accessible via CernVM-FS. The user authorization is managed by a LDAP system of Legnaro. In additions these machines work as LSF client to submit processes to a batch queue of resources, which are located in Padova and Legnaro managed by LSF. These nodes belong to the Legnaro-Padova Tier-2 CMS centre. Finally the machines of facility act as grid UI too, allowing jobs submission to the grid distributed environment. To reproduce this complex environment on the cloud, the project manager provided an image in qcow2 format to instantiate virtual machines on the cloud able to communicate with the Padova-Legnaro Tier-2 network through a dedicated NAT.

### 6.2. JUNO Experiment

The Jiangmen Underground Neutrino Observatory (JUNO) [11] is a multi-purpose neutrino experiment designed to determine neutrino mass hierarchy and precisely measure oscillation parameters by detecting reactor neutrinos from the Yangjiang and Taishan Nuclear reactors. The detector is under construction in China and the set-up will be ready to take data in 2020. The current activity of collaborators is to define the data and the computing model. The software for the offline provides packages for data simulation, calibration and reconstruction. The required computing power depends on the kind of process to run, e.g. the simulation of a calibration event needs more than 700 min. It means 12 hours for 10k events. Multi-core computing resources are required for algorithm parallelization. The possibility of having a configurable and dynamic infrastructure of computing resources is a strong motivation to explore cloud potentiality. Currently the framework adopted by JUNO to split a process in different jobs and to submit them in a distributed environment is DIRAC [12], already adopted by other experiments in the grid environment. The DIRAC framework uses the EC2 interface to instantiate virtual machines on the connected clouds. To provide the analysis software to all the distributed nodes the CernVM-FS tool is used. The authorized users belonging to the JUNO organization can submit jobs from an access point in Beijing. Job monitoring is done from there. Starting in December 2014, the JUNO tenant was added to the Cloud Area Padovana. The JUNO cloud manager provided the image in qcow2 format to instantiate the virtual machines as nodes able to run simulations and analysis. Therefore the virtual machines are included as nodes of the JUNO distributed farm, and can then run jobs submitted from the Beijing access point. These can also be accessed via SSH from INFN-Padova networks to run processes in interactive way. The test demonstrated the cloud is able to run jobs and it is a real help to fulfil the JUNO computing requirements. More than 700 jobs have been completed in the Cloud Area Padovana with 100% of success.

### 6.3. CMT Experiment

Cosmic Muon Tomography (CMT) [13] is a not invasive imaging system that can be used to scan inaccessible volumes of materials with high density (high atomic number). By studying the scattering of muons after matter collisions and applying statistical and iterative algorithms, the aim of the experiment is to find the optimal density of the target material that fits the muon trajectories and then its volume and shape. The algorithms for input data analysis require huge computing power, that explains the need to use parallel computing and the cloud. The detector is located at the INFN Legnaro National Laboratories. From here real and simulated input data are sent and stored in a GlusterFS storage server at the INFN-Padova data centre. A special image in qcow2 format was prepared by the cloud project manager. It allows the creation of virtual machines instrumented with analysis software and able to read the input data directly from the external storage, mounted via GlusterFS. This storage is a persistent storage for the cloud infrastructure. This server is also used to save the results of image reconstruction. Some tests to evaluate how parallelization is able to reduce the real processing time have been done. A comparison in performance using multi-core virtual machines with respect to the bare metal multi-core Intel processor has shown an improvement when using more than 6 virtual cores.

## 7. Conclusions

At the time of writing the Cloud Area Padovana IaaS provides computing and storage resources on demand to 8 physics experiments carried out within collaborations that include INFN-Padova and Legnaro National Laboratories research teams. The cloud is operated in high availability, load balanced mode, and managed and monitored by means of the most popular and effective tools that demonstrated their robustness when adopted in the past for the grid infrastructure. More than 50 users are currently registered with the OpenStack projects reserved to their experiments. Users from external institutes collaborating with the local teams can access the cloud through their SAML compliant federated identity providers, which have been integrated with the OpenStack Dashboard and Identity services. A fair share scheduling mechanism is under development in order to maximize the resource sharing among projects and increase the utilisation efficiency avoiding the static partitioning of resources. The cloud is expected to grow its overall capacity offer by including the new hardware resources that will be acquired in the next months by the research teams to address their increasing computing needs.

## References

[1] M. Biasotto et al., *The Legnaro-Padova distributed Tier-2: challenges and results*, J.Phys.Conf.Ser. **513** (2014) 032090

[2] P. Andrade et al., *Review of CERN Data Centre Infrastructure*, 2012 J. Phys.Conf. Ser. **396** (2012) 042002

[3] Hanks, Stan et al., *Generic routing encapsulation (GRE)* (2000). http://tools.ietf.org/html/rfc2784.html

[4] Tarreau, Willy, *HAProxy-The Reliable, High-Performance TCP/HTTP Load Balancer,* http://haproxy.1wt.eu

[5] Cassen, Alexandre, *Keepalived: Health checking for LVS & high availability,* (2002) http://www.linuxvirtualserver.org

[6] https://computing.llnl.gov/linux/slurm/priority_multifactor.html

[7] Aamodt, Kenneth et al., *The ALICE experiment at the CERN LHC,* Journal of Instrumentation **3.08** (2008): S08002.

[8] Chatrchyan, S et al., *The CMS experiment at the CERN LHC,* Journal of Instrumentation **3.08** (2008): S08004.

[9] Antcheva, I et al., *ROOT—A C++ framework for petabyte data storage, statistical analysis and visualization,* Computer Physics Communications **182.6** (2011): 1384-1385.

[10] Blomer, J et al., *Status and future perspectives of CernVM-FS,* J.Phys.Conf.Ser. **396** (2012) 052013.

[11] Y.-F. Li, *Overview of the Jiangmen Underground Neutrino Observatory (JUNO),* Int.J.Mod.Phys.Conf.Ser. **31** (2014) 1460300.

[12] Tsaregorodtsev, A et al., *DIRAC: a community grid solution,* J.Phys.Conf.Ser. **119** (2008) 062048.

[13] G. Bonomi et al., *Muon Tomography as a Tool to Detect Radioactive Source Shielding in Scrap Metal Containers,* Int. J. Mod. Phys. Conf. Ser. **27**, 1460157 (2014)