

# FIWARE OPS INFRASTRUCTURE PROVIDER MANUAL

---

## Contents

<b>FIWARE OPS INFRASTRUCTURE PROVIDER MANUAL .....</b>	<b>1</b>
<b>1 INTRODUCTION .....</b>	<b>3</b>
<b>2 CORE CONCEPTS .....</b>	<b>4</b>
2.1 What is an infrastructure according to XIFI .....	4
2.2 XIFI and stakeholders .....	5
2.3 XIFI offer to developers and infrastructure owners .....	5
2.4 XIFI Design Principles .....	6
<b>3 NETWORK SETUP .....</b>	<b>8</b>
3.1 Node Network Configuration .....	8
3.2 Géant MD-VPN solution .....	9
3.3 Federation nodes interconnection .....	11
3.4 Federation addressing plan .....	13
3.5 Registering a new node in the Federation .....	15
3.6 Monitoring Dashboard .....	19
3.7 Security Proxy .....	20
3.8 Cloud Portal .....	20
3.9 SLA Manager .....	20
3.10 Security Monitoring GE .....	21
<b>4 UPDATES ON PROCEDURES FOR OPERATING THE FEDERATION .....</b>	<b>22</b>
4.1 Operational requirements and procedures .....	22
4.1.1 Tenant deployment .....	22
4.1.2 Basic Tenant deployment procedure .....	22
4.1.3 Tenant Life cycle .....	28
4.1.4 Traceability of deployed Instances .....	29
4.1.5 Local catalogue management .....	30
4.1.6 Managing Images .....	31
4.1.7 Managing Blueprints .....	31
4.1.8 Use Case Handling .....	36
4.1.9 Tenant customization .....	37
4.1.10 Node administration .....	37
<b>5 HARDWARE DEPLOYMENT .....</b>	<b>41</b>
5.1 Deployment Architecture Reference Model .....	41
5.1.1 Concepts .....	41

5.1.2	Physical Deployment Models .....	43
5.1.3	Services Architecture Deployment Models .....	47
<b>6</b>	<b>SOFTWARE DEPLOYMENT .....</b>	<b>51</b>
6.1	IT Box .....	51
6.1.1	Installation Manual .....	51
6.1.2	User Manual .....	52
6.2	DCA .....	64
6.2.1	Installation Manual .....	64
6.2.2	User Manual .....	65
<b>REFERENCES .....</b>		<b>76</b>

## 1 INTRODUCTION

This manual aims to cover topics relevant to an infrastructure owner and infrastructure operators looking to deploy, setup and integrate a new infrastructure into the XIFI federation and by extension provide services on FIWARE Lab.

**Please note that the topics discussed in this manual are subject to change. It is recommended that you also reference the cited source documents online.**

## 2 CORE CONCEPTS

In this section the authors highlight some of the main guidelines that should be taken into consideration to drive the building of XIFI.

**XIFI as a community cloud:** XIFI is a federation of resources offered to the FI-PPP developer community by FI infrastructures. FI-PPP developers themselves may contribute to the community cloud by offering additional hardware capacity (under their own control or open to other PPP participants) and their own services (the so called Specific Enablers).

**XIFI as showcase for promotion of FI-PPP technologies for developers:** XIFI is the natural showcase to promote FI-PPP results by hosting Generic Enablers, Specific Enablers and end user applications build on top of them, by interconnecting data and advanced services offered by FI infrastructures.

**XIFI as opportunity for FI infrastructures to attract new communities of developers through FI-PPP services:** XIFI aims at identifying, within the project lifetime, ways to ensure its sustainability. The natural way is to proof to infrastructure owners that, keeping alive the community cloud, will allow them to attract communities of developers beyond what they achieved so far. Of course this has several implications related to FI-PPP IPR handling that will be analysed within the XIFI project.

**XIFI as validation and test of FI-WARE technologies in the field of Cloud Computing:** XIFI can be regarded as an horizontal use case project that leverage on FI-WARE Generic Enablers to show how to build a largely distributed and federated cloud platform that offer FI-PPP technologies to developers. In this sense, it is very important to collaborate with FI-WARE (and the upcoming Technical Foundation project) to define our fine-grained architecture and to provide requirements back to FI-WARE.

**XIFI as a flexible platform:** the need to integrate and federate different existing infrastructures, demands for ability to tackle different needs raised by infrastructures for their integration, both at technical level, operational level and business level. The flexibility regards also the capacity: for example at the moment it is very difficult to forecast the needed resources (e.g. CPU cores, RAM, storage, network bandwidth) and XIFI should be ready to scale-up (this means also that software and hardware architecture should support that) with the growth of resource demands.

### 2.1 What is an infrastructure according to XIFI

The authors used and will use the term infrastructure (and node as synonym) in different parts of this document and of XIFI documentation. Thus it is important to define what an infrastructure in our context is. In XIFI an infrastructure is any infrastructure that:

- offers capacity to host FI-WARE GEs for building FI applications (**compulsory**). To offer capacity to host FI-WARE GEs, the infrastructure should be able to host the Cloud Hosting GEs offered by FI-WARE through which the other GEs are provisioned. This means, in concrete terms, that the infrastructure should be equipped with a data center
- offers connectivity to Internet and GEANT network (**compulsory**). Connectivity to Internet allows developers and application end-users to access the services hosted on the infrastructure. This connectivity can be provided through different means (GEANT or other providers). XIFI, to leverage on EU FI facilities and to reduce the cost of connectivity, decided that the backbone network to allow XIFI nodes intra-communication is provided through GEANT
- offers additional capacities such as: sensing environment, advanced wireless connectivity, smart city datasets (**would like**). Future Internet developers may use such capacities to experiment GEs in real environments and on real data
- offers services to developers: support, backup, ... (**would like**). Infrastructure may offer

additional operational services to enrich their offer. The absence of these services should not be conflicting with operation of XIFI federation<sup>[1]</sup>

- offers access to end-user communities (**would like**). Infrastructures may be well connected to communities of end-users that developers can leverage on to test and validate their applications

## 2.2 XIFI and stakeholders

The above discussion revolves around the two main stakeholders of XIFI:

Future Internet Developers (intended as IT professionals involved in the development of FI applications): application developers that want to leverage on FI-PPP technology platforms to develop innovative applications for so called Future Internet scenarios (e.g. smart mobility, smart energy, smart healthcare).

Future Internet Infrastructures: infrastructures offering capacity to host Future Internet applications and advanced hardware/services that can be used to support Future Internet application developers. As such Future Internet infrastructures are service hosting providers.

The two stakeholders have different objectives and needs that XIFI should be able to balance in building its offer. Nevertheless, the main objective of FI-PPP programme is to foster large adoption (and validation) of FI-PPP technologies. This requires a critical mass of developers (beyond FI-PPP Large Trials projects) to adopt GEs and SEs to build Future Internet applications. Thus it is crucial for the success of FI-PPP (and of XIFI) to attract as much as possible developers willing to use FI-PPP tools. This clearly gives to FI Developers a privileged position among stakeholders. XIFI should balance these two perspectives (the one of FI Infrastructures and the one of FI Developers) keeping into account as well FI-PPP general objectives.

Other relevant stakeholders in the picture are:

Future Internet Core Platform developers – which correspond to FI-WARE (and the upcoming Technical Foundation project) developers, and that aim to offer the services (Generic Enablers) part of their platform to the Future Internet Developers.

Future Internet application sponsors and data providers – that support Future Internet Developers through financial and in-kind resources.

XIFI Consortium – that is in charge of the different operational and administrative activities to enable the provisioning of XIFI platform.

The stakeholders mentioned above may correspond to one or more role XIFI will take into consideration in the requirements analysis.

## 2.3 XIFI offer to developers and infrastructure owners

In this section the authors summarize some of the key elements of XIFI offer to developers and infrastructures.

XIFI will offer to developers: a single entry point to access FI-PPP technologies and underlying advanced Future Internet infrastructures. Through this entry point:

The developer will be able to access GEs and SEs deployed on different infrastructures in a transparent way.

The developer will be able to manage shared and private resources under her/his control.

The developer will be able to transparently deploy FI-WAREGEs and her/his own SEs.

The developer will be able to acquire information on the different characteristics of the infrastructures,

such as: advanced experimental services (e.g. sensor networks, smart energy grids), SLAs, usage term and conditions.

The developer will be able to create projects/experiments "encompassing" more than one site (infrastructure) in a transparent way.

The developer will have a single access point to monitor services he/she is using and platforms he/she deployed.

The developer will be supported in moving platforms he/she deployed from a location to another.

The developer will have access to tutorials for deploying FI-WARE GEs and its own SEs.

The developer will have access to the help desk granting Level 1 and Level 2 support.

XIFI will offer to infrastructure owners: a single entry point to publish their offer and access services meant for them. Through the single entry point:

The infrastructure owners will advertise: advanced experimental services (e.g. sensor networks, smart energy grids), SLAs, usage term and conditions.

The infrastructure owners will access to tutorials for: joining federation, deploying GEs, connecting GEs with their advances capabilities (e.g. sensor networks).

The infrastructure owners will access to the help desk granting Level 1 and Level 2 support for federation services and the installation process.

The infrastructure owners will deploy GEs to be made available in their datacentre.

## 2.4 XIFI Design Principles

In this section the authors highlight the most relevant design principles that have been followed in the definition of XIFI architecture and that should drive as well definition of single components of XIFI described in other technical deliverables. XIFI architecture defines a community cloud platform and as such it should adhere to canonical cloud computing design principles [2] and heterogeneous cloud deployment best practises [3]:

On-demand self-service [2]. "A consumer can unilaterally provision computing capabilities, such as server time and network storage, as needed automatically without requiring human interaction with each service provider"

Broad network access [2]. "Capabilities are available over the network and accessed through standard mechanisms that promote use by heterogeneous thin or thick client platforms (e.g., mobile phones, tablets, laptops, and workstations)".

Resource pooling [2]. "The provider's computing resources are pooled to serve multiple consumers using a multi-tenant model, with different physical and virtual resources dynamically assigned and reassigned according to consumer demand. There is a sense of location independence in that the customer generally has no control or knowledge over the exact location of the provided resources but may be able to specify location at a higher level of abstraction (e.g., country, state, or datacentre). Examples of resources include storage, processing, memory, and network bandwidth".

Rapid elasticity [2]. "Capabilities can be elastically provisioned and released, in some cases automatically, to scale rapidly outward and inward commensurate with demand. To the consumer, the capabilities available for provisioning often appear to be unlimited and can be appropriated in any quantity at any time".

Measured service [2]. "Cloud systems automatically control and optimize resource usage by leveraging a metering capability at some level of abstraction appropriate to the type of service (e.g., storage, processing, bandwidth and active user accounts). Resource usage can be monitored, controlled, and reported, providing transparency for both the provider and consumer of the utilized

service”

User centric [3]. Well-designed services and user interfaces are key to deliver best user experience that will facilitate adoption of cloud services.

Simplicity [3]. The adoption of complex solutions may impact delivery time and the service quality. Dealing with heterogeneous environments may require complex solutions to automate some of the processes. Some processes may be kept to a manual mode in an initial phase for time-to-market, rather than designing with full functionality and for all IT services.

Reuse [3]. Cloud computing is nowadays a well-established field where a plethora of solutions are available off the shelves to be deployed to support the creation of cloud based infrastructures. Such solutions should be reused as much as possible to allow for a fast kick-start of cloud provisioning activities, unless there are strong reasons to develop a new solution. Within XIFI, we will aim at adoption of FI-PPP technologies to achieve our architecture and to reuse, where needed, complementary Open Source software and Open Standards. XIFI when extending software to align them with its requirements will contribute back to the communities that originated adopted software.

Service dependability [4]. Dependability is a fundamental characteristic of cloud computing platforms. Service availability is one of the key attributes for a service to be defined dependable. Service availability requires dealing with issues such as no single point of failure and scalability/elasticity mentioned above. While some of these features are not dependant on the provisioning platform, but rather on the services developed on top of the platform, in the design of Cloud provisioning platform service availability should be carefully planned also taking into consideration derived costs [3]. XIFI architecture aims at delivery dependable services, with focus on availability, reliability, safety and transparency [2] attributes [4]. Services available in the single XIFI nodes, should be available regardless the availability of other nodes. Highly availability and reliability should be provided as well for management tools provided by XIFI platform. Integrity, maintainability and confidentiality should be guaranteed by the reuse of dependable cloud tools.

Flexibility. The requirements that XIFI will face may change over the time, as such XIFI architecture should be designed for adapting to new requirements. In this sense modularity and service orientation are fundamental principles to be considered to design XIFI architecture.

Compatibility. XIFI aims at bringing on board as nodes existing Future Internet infrastructures, as such it is important that the design of XIFI architecture keeps into consideration the continuity of their operational and business activities and that allows for integration of XIFI architecture on top of existing tools adopted by infrastructures (where this does not conflict with FI-PPP technology centric principle).

### 3 NETWORK SETUP

The most complex part of creating a XIFI site is the network deployment and configuration. The following guide is based on the experience of existing nodes.

#### 3.1 Node Network Configuration

In order to set up correctly the network, a good approach is to follow the OpenStack guidelines. OpenStack distinguishes four logical networks: [\[3\]](#).

- Management network
- Data network
- External network
- API network

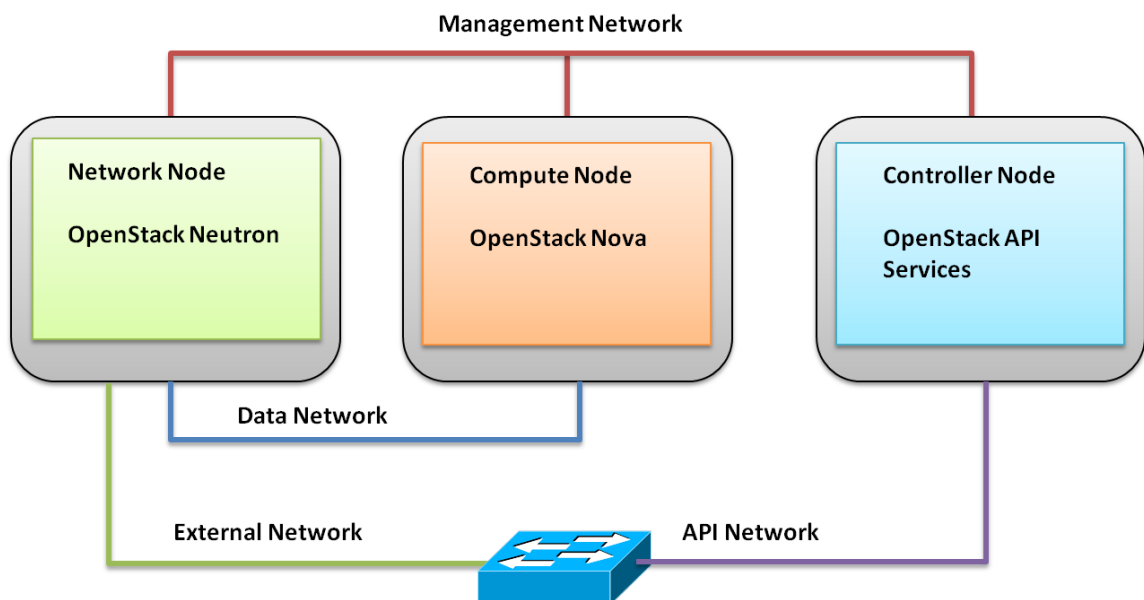


Figure 1 - OpenStack Networking Architecture

Basically the first two are intended as “internal networks”. The former is used for management purpose and provides connection between OpenStack components. The latter is used for VM data communication. These networks should have IP addresses reachable only within data centre. As stated in OpenStack Operations Guide [\[4\]](#) the recommended option is to use a management network with separate switch and separate NICs: “this segregation prevents system administration and monitoring system access from being disrupted by traffic generated by the guests themselves”. The external network and the API network are intended as “public networks” because they allow inbound connections: the IP addresses should be reachable by anyone on the Internet. In particular the external network is used to provide both VMs inbound and outbound connection from VMs, whereas the API network exposes all OpenStack APIs to tenants. These networks can be merged into a single network.

If it is not possible to have physically separated networks, i.e. different switches and NICs for different networks, is possible to use VLANs to segregate network data. In this case the switch must be configured accordingly (the configuration steps are different for different networks). For instance,



assuming nodes with at least 3 NICs, we can use the interfaces as follow:

- eth0: Management Network
- eth1: External/API Network
- eth2: Data Network

Each interface must be configured with the correct(s) VLAN(s) ID: in this way is possible to create logical separated networks.

### 3.2 Géant MD-VPN solution

This section describes the MD-VPN Multi-domain GÉANT Service found in [\[5\]](#) The GN3plus Multi-Domain Virtual Private Network (MD-VPN) service will deliver seamless private interconnection of two or more networks across multiple network domains such as GÉANT and the NREN backbones. The MD-VPN service is able to deliver L2-VPN (point-to-point) and L3-VPN (multi-point) across multiple network domains. This allows the users of the IPv4/IPv6 and/or layer 2 networks to work as if their networks are coupled together. The use of a VPN improves end-user performance, security, and facilitates the use of private IP addresses, RFC 1918[6], across the distributed user environment.

A typical scenario would be one where an organisation seeks to connect a number of sites from different physical locations as if they were in the same physical location, to enable the organization to access remote resources with the same level of security. This security improvement enables high network performance by avoiding deep firewall inspection, such as the case with standard IP.

Figure A illustrates how the NRENs are connected together to provide the MD-VPN service. The service is offered collaboratively by GÉANT and a set of adjacent NREN domains that adhere to the service. These joint networks form the multi-domain area where the service is provided (this is known as the "GÉANT service area)." The MD-VPN service guarantees that the data of VPN1 users cannot be delivered to sites outside VPN1, and that sites or machines outside VPN1 are unable to connect to machines that are in VPN1. This is achieved by isolating the MD-VPN customer data flows from any other traffic, standard IP traffic and traffic of other MDVPN customers.

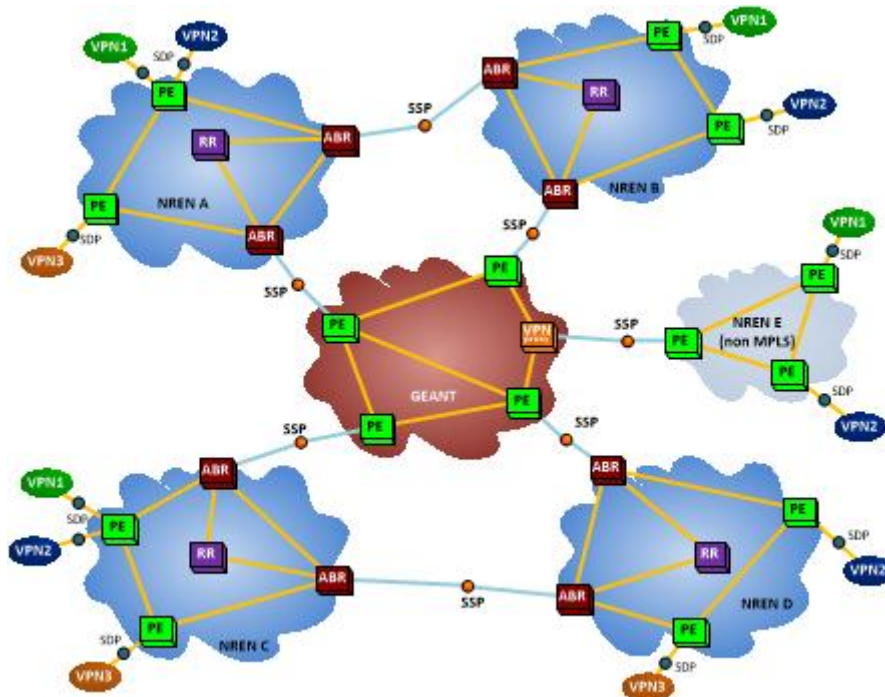


Figure 2 - The Multi Domain VPN

The multiple networks forming the MD-VPN service peer VPN information at the borders between NREN networks. These peering points are referred to as Service Stitching Points (SSPs) in Figure A. The Service Demarcation Points (SDPs) are the end points where the service is terminated for the end users of the MD-VPN service. These points are located at the interfaces of Provider Edge (PE) devices found at the NREN end of the site access link, which forms the edge of the federated MD-VPN service domain.

The MD-VPN service is based on the VPN transport service, provided by the GÉANT network, which allows transporting all the multi-domain VPNs set-up within MP-VPN service from one domain (NREN) to another domain. In fact GÉANT provides a Carrier of Carriers (CoC) VPN to connect the NREN resources together, with each NREN-GÉANT BGP-labelled unicast peering session being established as an ABR-PE session between the respective routers. Simply, the NREN network will function as a CoC VPN end user. This approach introduces a hierarchy and maintains great network flexibility and a peer model in NREN cooperation. The main goal of using BGP labelled-unicast protocol for the MD-VPN service is the distribution of routes and MPLS labels for all PE routers in all domains used to provide services for end users. BGP also introduces high scalability and robust solutions such as Route Reflectors, denoted by RR in Figure A.

For NRENs that do not support MPLS switching on the SSPs, the VPNs from those NRENs must be stitched to the service in back-to-back mode (Option A RFC 4364). This means that on the SSP, each VPN that a NREN would like to transport has to have its own sub-interface identified with a service delimiter (e.g. VLAN ID). The traffic and signalling for a particular VPN is exchanged over this sub-interface, and typically in the case of L3-VPN, a standard BGP protocol with VPN extension will be enabled for each VPN instance.

A VPN proxy system is defined in the GÉANT network that will provide access to the Carrier of Carriers VPN. This allows GÉANT to provide MD-VPN services for end users directly connected to the GÉANT network or to connect NRENs that do not support MPLS and BGP labelled unicast.

### 3.3 Federation nodes interconnection

The basic federation connectivity will be provided by using a layer-3 MD-VPN provided by GÉANT and the NRENs.

The current plan to provide traffic isolation for multi-site tenant deployment is to use layer-2 GRE tunnels dynamically created by OpenStack and the **Error! Reference source not found.** Network GE ssisted by OpenNaaS.

It is expected that many of the NRENs will be able to offer the MD-VPN service to the federation nodes.

Some NRENs who have not yet committed to deploy the MD-VPN service will be able to provide a separate L3-VPN instance and will connect these VPN instances to those supported by the MD-VPN platform via the GÉANT “VPN-Proxy” described above. This GÉANT VPN-Proxy is expected to be operational Q2 2014. In the case that a node cannot be connected to an NREN for some reason, it may be possible to directly connect to the VPN-Proxy in the GÉANT network.

It should be noted that work is in progress to engineer suitable solutions for XIFI, like for example, providing native IPv6 support on the MD-VPN.

#### Backup connectivity solution

A secondary communication channel for the inter-node connectivity, compatible with the L3-VPN deployment, easy and fast to setup with low cost even with reduced performances if compared with the primary channel, will be helpful in particular for the following scenarios:

- **failover:** for nodes already active and federated when the primary channel running L3-VPN service goes down.
- **initial/temporary:** for new nodes that are entering into the federation and still waiting the primary connectivity based on the GÉANT MD-VPN service.

One of the fastest and widespread technical solution to achieve this result is the VPN tunnel over Internet, to connect a generic leaf node directly to a master node. The master node has usually a certain bandwidth to offer, and it's delegated to handle the prefixes and traffic of all the federated nodes for the connected leaf node.

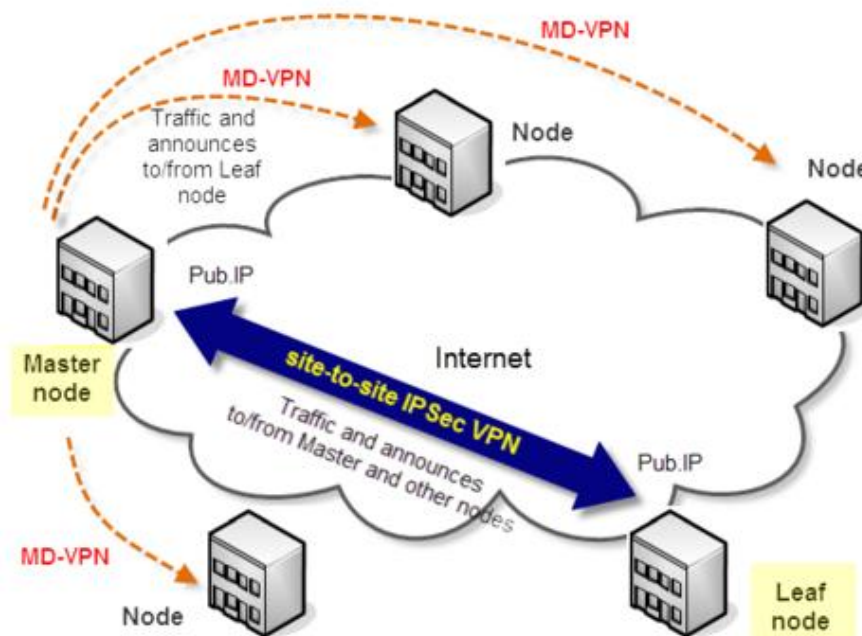


Figure 3: Backup connectivity

**VPN:** there are lots of protocols and many technical solutions used to manage VPN. The first aim in that way was to identify a solution able to guarantee a certain level of security using authentication of the session and encryption of the traffic, facing on Internet. The second step, considering the previous output, was to identify a widely supported technical solution to adopt to maximize the compatibility with different node equipment. Here under the result of the analysis, that will be an initial agreement between the leaf node and the master node where it will be connected:

- **VPN Protocols:**
  - **IKE:** Security Association Protocol with pre-shared key, AES encryption and SHA message digest.
  - **IPSEC:** suite for securing Internet Protocol with AES encryption and SHA message digest.
- **VPN Approach:**
  - **VPN Client-Concentrator:** the generic node setup the VPN transport tunnel dynamically toward the master node, when needed.
  - **Site to Site VPN:** the generic node and the master node setup together, with same roles, a static transport tunnel using on it a predefined point to point (PTP) subnet. This one is the preferable approach because of the intrinsic stability and the compatibility with the primary connectivity in order to minimize the adjustments with the BGP peering.
- **Routing protocols:**
  - **BGP:** In order to maintain the same routing protocol used on the primary connectivity, also on the VPN tunnel will be used the BGP if possible.
  - **Static + BGP:** A simple approach could be based on the use of static routes on the leaf node, to send all the internal federated traffic to the tunnel interface toward the federation mesh, and a modified instance of the BGP protocol on the master node which will announce also the leaf node prefix to the MD-VPN service in addition to their own prefix.

**Technical details:** Here is a table describing a proposal of IPsec tunnel implementation based on what it has already been implemented between Brittany and Trento. Configuration may be adjusted to much the needs. It only needs to be agreed between the two nodes participating to the tunnel:

Parameter	Value
<b>IKE</b>	Version 1
<b>Authentication Method</b>	Pre shared key
<b>IKE Coding Algorithm (encryption)</b>	AES 128 bit
<b>IKE Message Digest Algorithm (authentication)</b>	SHA1
<b>Ph. 1 Diffie Hellmann group (DH)</b>	2
<b>Ph. 1 Lifetime</b>	28800
<b>Nat Traversal</b>	disabled
<b>IPSec Protocol</b>	ESP
<b>Perfect Forward Secrecy (PFS)</b>	enabled
<b>Ph. 2 Diffie Hellmann group (DH)</b>	2

<b>IPSec Coding Algorithm (encryption)</b>	AES 128 bit
<b>IPSec Message Digest Algorithm (authentication)</b>	SHA1
<b>Ph. 2 Lifetime</b>	3600
<b>Replay detection</b>	disabled
<b>Auto key keepalive</b>	disabled

### 3.4 Federation addressing plan

Connecting the federation through a private VPN requires to define a shared addressing plan that must be respected by all partners.

A well-designed effective IP addressing plan has many benefits. It is proposed that in XIFI we use the below plan to bring about flexible, functional IP Structure. It allows for the easy reading and execution of routing tables, GE location, Scalability to new infrastructures.

The following table highlights all the IP address ranges associates with each node.

<b>XIFI site</b>	<b>Federated IP4 Range</b>
<b>Irish node (Waterford)</b>	10.0.0.0/20
<b>German node (Berlin)</b>	10.0.16.0/20
<b>Italian node (Trentino)</b>	10.0.32.0/20
<b>French node (Brittany)</b>	10.0.48.0/20
<b>Spanish node (FIWAT)</b>	10.0.64.0/20
<b>SERENA</b>	10.0.80.0/20
<b>CZIFI</b>	10.0.96.0/20
<b>SUNTAI</b>	10.0.112.0/20
<b>iLabt.XIFI</b>	10.0.128.0/20
<b>XIFI NFN</b>	10.0.144.0/20
<b>XFP</b>	10.0.160.0/20
<b>ZUFI</b>	10.0.176.0/20
<b>XIFI@Budapest</b>	10.0.192.0/20
<b>IWAVE</b>	10.0.208.0/20
<b>C4XIFI</b>	10.0.224.0/20
<b>XIFIBUR</b>	10.0.240.0/20
<b>NITOS</b>	10.1.0.0/20

*Table 1: Federation addressing plan*

*Please notice that addressing plan may evolve as new partners join the federation. In order to have the latest version of the IP addressing and in order to have the IP addresses of the federated published*

services, check the article on the WIKI<sup>1</sup>

## IP network description

XIFI has a clear distinction between network types and their functions. Portrayed in the sections below there are descriptions of each network profile and their roles to be considered when deploying a XIFI node.

### Public federation IP

The Public federation network (publicly routable IPv4 range) provides public IP accessibility within the federation cloud infrastructure. This allows FI-Ware Generic Enablers the ability to reach external IPv4 address exterior to the FI-Lab cloud infrastructure and vice versa. They are labelled as “external networks” in OpenStack and are mostly used by the Generic Enablers. The public IP addresses are allocated to a Generic Enabler via OpenStack’s floating IP process.

### Private federation IP

The Private Network is RFC1918 IP Range, and is not publicly routable. Federation private networks are defined in three distinct areas, two data networks and one management network:

- Federation Network :

The Federated private IP is allotted manually and via by OpenStack’s L3 agent. The primary reason for this network is twofold, firstly to provide connectivity between federation master/slave services and secondly to connect Generic Enables between federation nodes. IP address can be allocated via OpenStack floating IP range and are considered “external” with in an OpenStack deployment. Routing for this network is applied external to the OpenStack instance.

- Administrative network:

The IP addresses are manually allocated by the infrastructure operator. This network is the one used by ITBox to manage the nodes deployment and it is used by the infrastructure hosting the node for OAM tasks.

- Management Network:

These IP address are manually allocated by the infrastructure operator and handle traffic between OpenStack nodes. It can be deployed as a single network or sub divided into functional areas to provide functional isolation. Types of traffic include OpenStack’s Identity Service, Image Service, Dashboard, Block Storage and can carry tunnel overlay traffic outlined in internal network. This network has also the role of the *internal network* allowing VMs on the same tenant connect to each other and access the external network.

- Storage Network:

These IP address are manually allocated by the infrastructure operator and handle the storage traffic addressed to the storage nodes of the datacenter.

### Private XIFI Network IP address allocation

Each FI-Lab Node is allocated a range of private IP address space for federation use. Below outlines a description of private IP space and who takes on the role of allocating this resource.

- Federation Network: IP block assigned by Federation
- Administrative Network: IP block assigned by OpenStack Operator with node owner discretion.
- Management Network: IP block assigned by OpenStack Operator with node owner discretion.

<sup>1</sup> [http://wiki.fi-xifi.eu/Xifi:Wp5: Federation\\_services\\_connectivity](http://wiki.fi-xifi.eu/Xifi:Wp5: Federation_services_connectivity)



- Storage Network: IP block assigned by OpenStack Operator with node owner discretion.

### 3.5 Registering a new node in the Federation

Every Openstack component needs to be properly configured to make requests to Keystone Proxy instead of to the traditional Keystone. In most cases this will consist of changing three parameters:

- IP address: In case of installing Keystone Proxy on a different machine.
- Port: Keystone Proxy does not use the same ports as Keystone. TCP Port 5000 in Keystone is mapped to 4730 in Keystone Proxy, and 35357 is mapped to 4731.
- Region: the name of your node/region.

Below is a list of configuration files that need to be changed in Openstack Grizzly:

- /etc/nova/nova.conf
- /etc/quantum/dhcp\_agent.ini
- /etc/quantum/metadata\_agent.ini
- /etc/quantum/quantum.conf
- /etc/quantum/l3\_agent.ini
- /etc/quantum/api-paste.ini
- /etc/glance/glance-api.conf
- /etc/glance/glance-registry.conf
- /etc/glance/glance-cache.conf
- /etc/cinder/cinder.conf
- /etc/swift/proxy-server.conf

Some of these files can be also hosted in the Controller and in the Compute nodes. It depends on the OpenStack configuration adopted by each Infrastructure Owner. For example you can install Openstack Networking in the Controller server or in a dedicated one. Usually in Compute nodes you have only to change the nova configuration file.

Here are the specific lines to be modified in each component. Once these files are updated you should restart the corresponding services:

```
1. /etc/nova/nova.conf
...
quantum_admin_auth_url=http://keystone_proxy_host:4731/v2.0
quantum_admin_password=quantum
...
keystone_ec2_url=http://keystone\_proxy\_host:4730/v2.0/ec2tokens
...
[keystone_authtoken]
```

```
auth_port=4731
admin_password=nova
admin_user=nova
...
auth_host=keystone_proxy_host
...
quantum_region_name=[YOUR_REGION_NAME]
```

## 2. /etc/quantum/dhcp\_agent.ini

```
...
admin_user=quantum
...
admin_password=quantum
...
auth_url=http://keystone\_proxy\_host:4731/v2.0
admin_tenant_name=services
...
```

## 3. /etc/quantum/metadata\_agent.ini

```
...
auth_url = http://keystone\_proxy\_host:4731/v2.0
auth_region = [HERE YOUR REGION'S NAME]
admin_tenant_name = services
admin_user = quantum
admin_password = quantum
...
```

## 4. /etc/quantum/quantum.conf

```
...
[keystone_authtoken]
auth_host = keystone_proxy_host
auth_port = 4731
auth_protocol = http
admin_tenant_name = services
admin_user = quantum
admin_password = quantum
signing_dir = /var/lib/quantum/keystone-signing
auth_url=http://keystone\_proxy\_host:4731/v2.0
```



...

5. /etc/quantum/l3\_agent.ini

...

admin\_user=quantum

admin\_tenant\_name=services

admin\_password=quantum

...

auth\_url=[http://keystone\\_proxy\\_host:4731/v2.0](http://keystone_proxy_host:4731/v2.0)

...

6. /etc/quantum/api-paste.ini

...

[filter:authtoken]

paste.filter\_factory = keystoneclient.middleware.auth\_token:filter\_factory

auth\_host=keystone\_proxy\_host

admin\_password=quantum

auth\_url=[http://keystone\\_proxy\\_host:4731/v2.0](http://keystone_proxy_host:4731/v2.0)

admin\_tenant\_name=services

admin\_user=quantum

auth\_port=4731

...

7. /etc/glance/glance-api.conf

...

swift\_store\_auth\_address=[http://keystone\\_proxy\\_host:4730/v2.0/](http://keystone_proxy_host:4730/v2.0/)

...

swift\_store\_user=services:glance

...

auth\_port=4731

auth\_host=keystone\_proxy\_host

admin\_password=glance

admin\_tenant\_name=services

auth\_protocol=http

auth\_uri=[http://keystone\\_proxy\\_host:4731](http://keystone_proxy_host:4731)

admin\_user=glance

...

```
filesystem_store_datadir=/var/lib/glance/images/  
swift_store_region=[YOUR_REGION_NAME]
```

```
8. /etc/glance/glance-registry.conf  
[keystone_authtoken]  
signing_dir=/tmp/keystone-signing-glance  
auth_host=keystone_proxy_host  
admin_password=glance  
auth_port=4731  
signing_dirname=/tmp/keystone-signing-glance  
auth_protocol=http  
admin_tenant_name=services  
admin_user=glance  
...
```

```
9. /etc/glance/glance-cache.conf  
admin_user=glance  
...  
admin_tenant_name=services  
...  
admin_password=glance  
...  
auth_url=http://keystone\_proxy\_host:4731
```

```
10. /etc/cinder/cinder.conf  
...  
admin_password=cinder  
auth_port=4731  
auth_host=keystone_proxy_host  
admin_tenant_name=services  
auth_protocol=http  
admin_user=cinder  
...
```

```
11. /etc/swift/proxy-server.conf  
...  
[filter:s3token]  
paste.filter_factory = keystone.middleware.s3_token:filter_factory  
auth_port = 4731
```

```

auth_protocol = http
auth_host = keystone_proxy_host
...
[filter:keystone]
use = egg:swift#keystoneauth
operator_roles = admin, SwiftOperator, member
is_admin = true
cache = swift.cache
...
[filter:authtoken]
paste.filter_factory = keystone.middleware.auth_token:filter_factory
auth_host = keystone_proxy_host
auth_port = 4731
auth_protocol = http
auth_uri = http://keystone\_proxy\_host:4731
admin_tenant_name = services
admin_user = swift
admin_password = swift
...

```

Note: If the node is in HA mode you have to change and restart also the quantum service using the corosync/pacemaker commands:

1) crm configure

2) edit the l3-agent and the dhcp-agent replacing the old IP/password with the new one:

```

params password="quantum" username="quantum" tenant="services"
os_auth_url="http://keystone_proxy_host:4731/v2.0" \

```

3) commit

### 3.6 Monitoring Dashboard<sup>2</sup>

The Monitoring Dashboard is a mashup of different monitoring tools for some of the XIFI components currently deployed. Nowadays it consists of two components: VMs monitoring tool and NAM monitoring tool.

VMs monitoring tool (integrated in the Cloud Portal) is in charge of the Virtual Machines monitoring. It gets data from Federating Monitoring API in order to show the user the CPU, Memory and Disk

<sup>2</sup> NB: This section was taken from deliverable D2.4 XIFI Handbook v2, which is not yet released

status of its deployed Virtual Machines.

NAM monitoring tool is oriented to XIFI IO and provides them information about the Network status of the connections between XIFI nodes. They can obtain monitoring data about bandwidth, latency and packet loss in two ways: real data information and historical data information.

More details can be found in the deliverable D4.2 – Baseline Tools v2 [29]**Error! Reference source not found..**

### 3.7 Security Proxy<sup>3</sup>

Security Proxy is the component responsible for authenticating requests that are sent from/to Cloud services. It generates special tokens for representing users acting on behalf of organizations, and these tokens are later used manage different services in the Cloud infrastructure. Federated Keystone Proxy aims at providing federated access to Cloud resources for members of XIFI federation. It is distributed on all the nodes of the federation and stores authentication tokens. The Federated Keystone Proxy extends the FI-WARE Keystone Proxy component to support XIFI requirements. The figure below shows how this architecture will be implemented for different nodes in XIFI.

### 3.8 Cloud Portal<sup>4</sup>

The Cloud Portal provides a support for the users of the cloud infrastructure and platform to manage their services and resources deployed in cloud. It is implemented as a Web GUI, following the example of the portals that the most common cloud infrastructure managers (like Amazon EC2, Eucalyptus, Cloud Sigma, Rackspace, etc.) have today. In concrete it bases its design principles on the OpenStack Horizon Dashboard. The basic objective of the user portal is to facilitate the user of the cloud to perform operations over the underlying infrastructure. This includes performing actions such as: create user; manage projects; lunch instances on a base of images; create images in the image repository; retrieve flavors from the resource; etc. Moreover the portal facilitates management of a Virtual Data centers (VDCs), Services and Service Data Centers (SDCs), PaaS management and will offer monitoring statistics of the physical and virtual machines.

More details can be found in the deliverable D4.2 – Baseline Tools v2 [29]**Error! Reference source not found..**

### 3.9 SLA Manager<sup>5</sup>

SLA Management, based on SLA management standards and integrated in the XIFI portal, covers the SLA lifecycle and allows the direct interaction among the different actors through the graphical user interface. It supports these actions: i) the providers indicate the QoS metrics that can be monitored on their infrastructures ii) the users view the available metrics for the service and decide the boundaries which should monitor.

There are two main roles: users (developers, End User, SMEs...) and providers (Infrastructure Owners, technological providers...). They can use the SLAs Dashboard, included in the SLA Manager component, to monitor the Service Level Agreement (SLA) established between them during the

<sup>3</sup> NB: This section was taken from deliverable D2.4 XIFI Handbook v2, which is not yet released

<sup>4</sup> NB: This section was taken from deliverable D2.4 XIFI Handbook v2, which is not yet released

<sup>5</sup> NB: This section was taken from deliverable D2.4 XIFI Handbook v2, which is not yet released

resource provisioning. The QoS metrics included in the SLA agreement will be recovered from the Federation Monitoring GE and checked with the negotiated values to detect any violation of the SLA. More details are available at the SLA Manager section included in the deliverable D4.2 [29].

### 3.10 Security Monitoring GE<sup>6</sup>

The Security Monitoring GE is one of the main Generic Enablers provided by the FI-WARE Security Architecture which, in the context of the XIFI Federation, allows not only to Federation Managers but also to Infrastructure Owners, to identify and assess attacks and security risks that can impact on their federated resources. Besides, in conjunction with the XIFI Security Dashboard and due to its integration with the Identity Management GE, it offers an efficient and user-oriented monitoring from the security perspective.

In particular, the component of the Security Monitoring GE used in the XIFI Federation is the Service Level SIEM (Security Information and Event Management), working in a distributed way where Security Probes with SIEM agents are installed on the slave nodes for the collection of logs and normalization into security events, whereas the SIEM server is installed on the master nodes for the analysis and correlation of those security events received from the different nodes to ensure the compliance with the security directives established in the XIFI Federation.

More details can be found in XIFI deliverable D4.2 [29].

---

<sup>6</sup> NB: This section was taken from deliverable D2.4 XIFI Handbook v2, which is not yet released

## 4 UPDATES ON PROCEDURES FOR OPERATING THE FEDERATION

### 4.1 Operational requirements and procedures

Below a number of procedures are defined that are intended to describe the Infrastructure Owner operations. This definition is built on the experience of XIFI partner TID gained from running the FI-Lab legacy platform in FI-WARE. The information in this section complements the information in the handbook Deliverables D2.1 and D2.4.

#### 4.1.1 Tenant deployment

Below, a basic procedure is described of what must be deployed by a user (developer) in order to have a tenant with instances up and running.

- Use of IP addresses:

Public IP addresses are a scarce resource. They are assigned to tenants as a way to deploy a tenant with its own private router. Care has to be taken that IP addresses are used efficiently. The application of quota, i.e. the maximum number of IP addresses per tenant, is already in place by some nodes.

- Quota:

Default values for a Node are given in the D2.1 "XIFI Handbook v1" as following:

- quota\_instances: 3 (number of instances allowed per tenant)
- quota\_floating\_ips: 3 (number of floating ips allowed per tenant)
- quota\_cores: 6 (number of instance cores allowed per tenant)
- quota\_volumes: 10 (number of volumes allowed per tenant)
- quota\_gigabytes: 1000 (number of volume gigabytes allowed per tenant)
- quota\_ram: 2034 (megabytes of instance ram allowed per tenant)

These numbers are default values and might be changed by any IO in order to fit the dimensioning of his node.

As an example, the default number of floating IPs that is allowed by tenant is 3. This number could be judged by an IO as too big taking into account that FI-Lab has its portal accessible to anyone without really restrictive measures. A fair number could be put to 1 and be changed by an IO in a case by case manner.

- Volumes:

Attachment of volumes is an ongoing problem that need to be solved and this is why it has not been detailed in the below procedure.

#### 4.1.2 Basic Tenant deployment procedure

##### Create new organisation:

New organisations are created only in the federation portal, not at nodes level. To create a new organization, you must go to the Account part. In this field, you will also grant the different kind of accesses have the users of your organization.

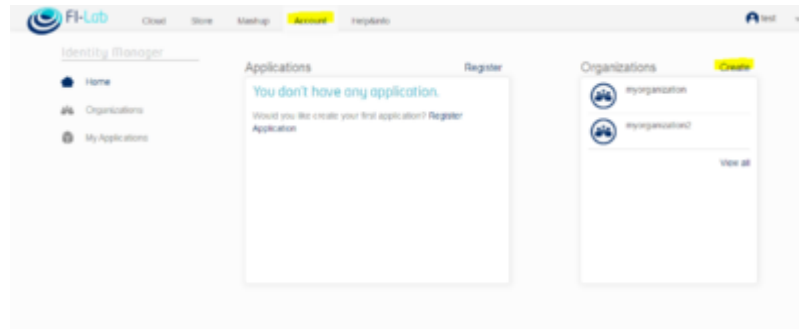


Figure 4: Account part

Once you are done with granting access, you can go back to the cloud portal and choose the organization you just created as "Project Name" and the Region where you want to start building your tenant.

### Create network and subnet

First of all you must create a network with a private subnet. To do this, you should click on the "Networks" button then click on "Create Network" (see screen shot below)

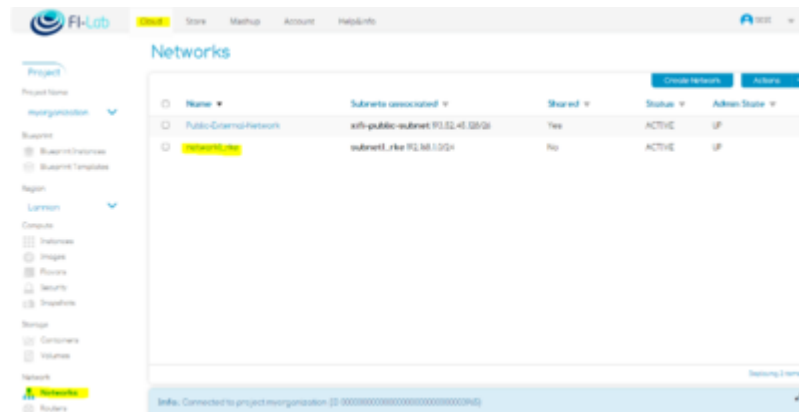


Figure 5: Create a network

Fill the information to create your network, e.g.:

Network name: mynetwork

Subnet name: mysubnet

network address: 192.168.0.0/24

Gateway IP: 192.168.0.1

and push the create button.

### Create a router:

- Create a router by click on "Routers" on the bottom of the left menu, then "Create Router"

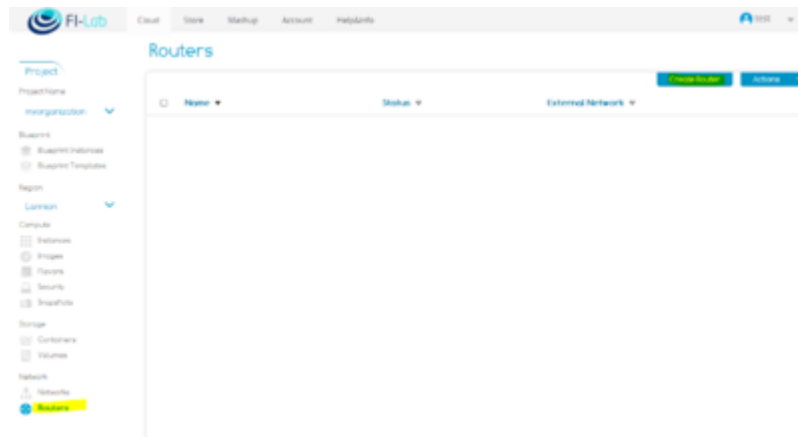


Figure 6: Create a router

- Add an interface: Click on the "router", then add an interface corresponding to your private subnet you created earlier
- Set Gateway: From the main router menu, click on set the gateway and choose the "Public External Network"

If you go back to your interface details of your router, you should at this stage, 2 interfaces: 1 corresponding to your private subnet and 1 for the external network.

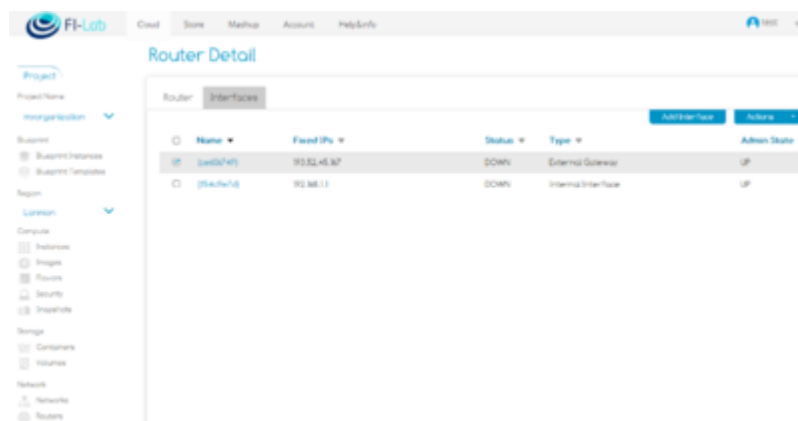


Figure 7: Add an interface

## Security groups

To create your security group, you should click on the "Security" button then click on "Security Groups", then "Create Security Group" (see screenshot Figure 8).

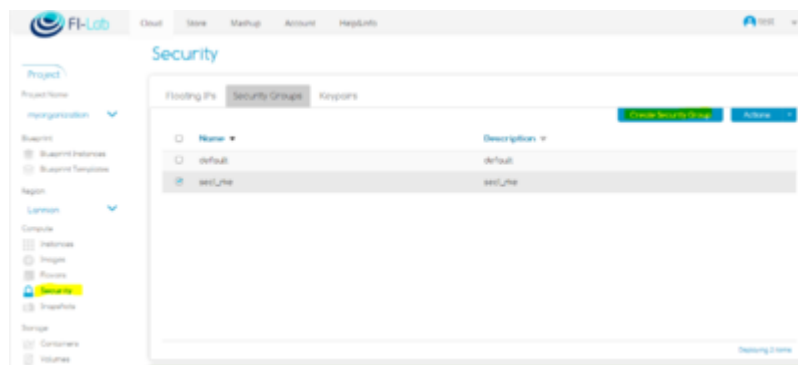
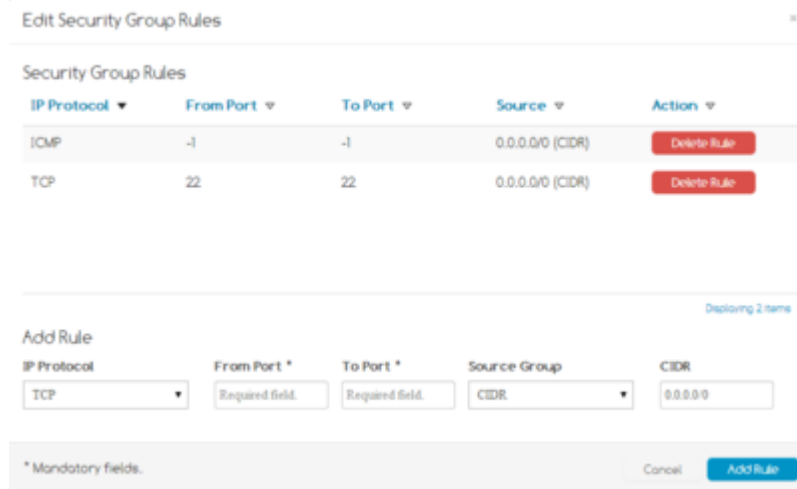


Figure 8: Create Security Group



Add (some) rules to your security rules, see Figure 9.



**Edit Security Group Rules**

Security Group Rules

IP Protocol	From Port	To Port	Source	Action
ICMP	-1	-1	0.0.0.0/0 (CIDR)	Delete Rule
TCP	22	22	0.0.0.0/0 (CIDR)	Delete Rule

Deploying 2 items

**Add Rule**

IP Protocol: TCP  
 From Port: Required field.  
 To Port: Required field.  
 Source Group: CIDR  
 CIDR: 0.0.0.0

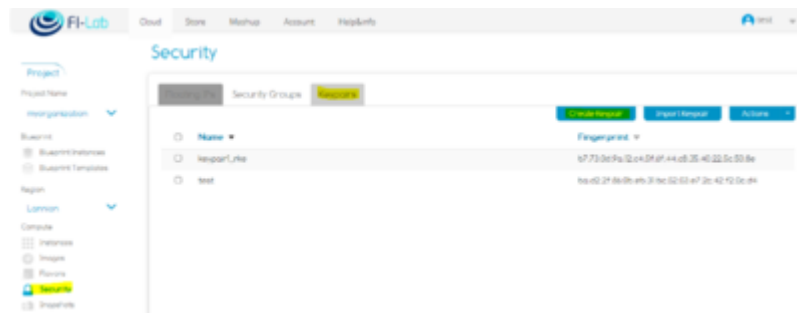
\* Mandatory fields.

Cancel Add Rule

Figure 9: Add rules

## Create the Keypair

Click on "Keypairs" Tab (Figure 10), create your own keypair. At the end of the creation, it is proposed to download the keypair, then you must answer "yes" as it is the only time you can do it.



**Security**

Security Groups

Keypairs

Name	Fingerprint
keypair1	97:73:0a:Pa:2e:04:01:44:cd:35:40:22:5c:50:8e
test	ba:42:2f:8b:0b:3f:bc:02:03:a7:3e:42:f2:0e:04

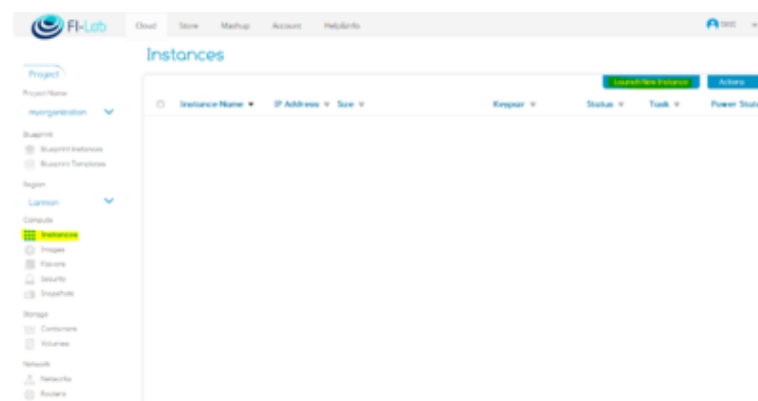
Buttons: Create Keypair, Import Keypair, Actions

Figure 10: Define the Keypairs

Note: It is important to know that a keypair is associated with a user and with a tenant. In other words, it means when you create keypairs, other users having access to the tenant won't see and won't be able to use keypairs you created.

## Create your first VM

From the left menu (Figure 11), select "Instances", then click on "launch New Instance"



**Instances**

Instances

Instance Name	IP Address	Size	Keypair	Status	Task	Power State
---------------	------------	------	---------	--------	------	-------------

Buttons: Launch New Instance, Actions

Figure 11: Create an instance

- Image: Choose the cloud image you want to use to create your VM
- Name: Put the name of the VM of the VM you want to create
- Flavour: Choose the flavor you want to be applied for your VM
- Instance Count: 1
- Define the keypair and the security group: Select the one you just created
- Networking: Choose the private subnet you created earlier
- Then push the "Launch instance" button

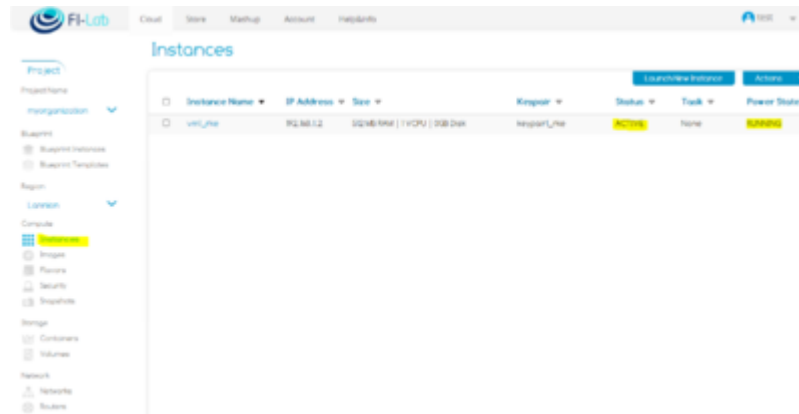


Figure 12: Launch instance

Once launched (Figure 12), you can click on it, to check the logs and that you instance has been created successfully, Figure 13.



Figure 13: Instance Log

### Floating IP:

- Allocate an IP to the project:

Under the left menu, click on "Security", choose the "Floating IPs" Tab, then click on "Allocate IP to Project"(Figure 14)

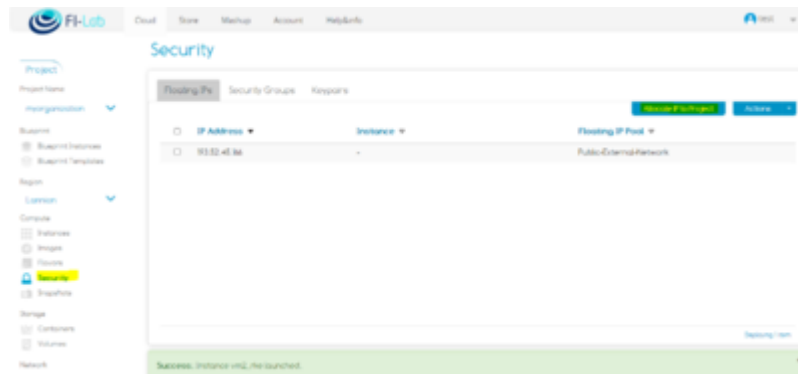


Figure 14: Allocate IP

- Associate the IP to your Instance

In the "Action" field, click on associate IP and select the instance you want to associate the IP (Figure 15)

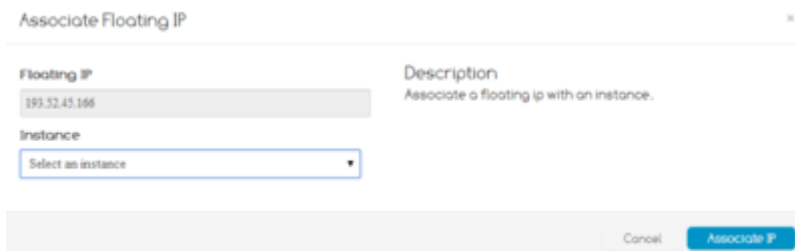


Figure 15: Associate IP

Once you have associated the IP to your instance, it is accessible through internet by SSH and ping. These are the only two protocols you allowed in your security group.

- Try to ping your instance from your personal computer

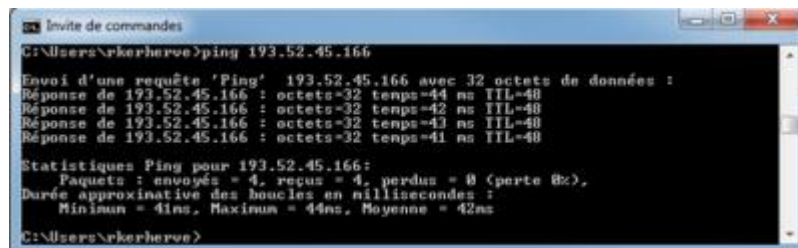


Figure 16: Ping

- Connect to your instance via SSH:

Do not forget to use the public key, you downloaded and used earlier for your instance.

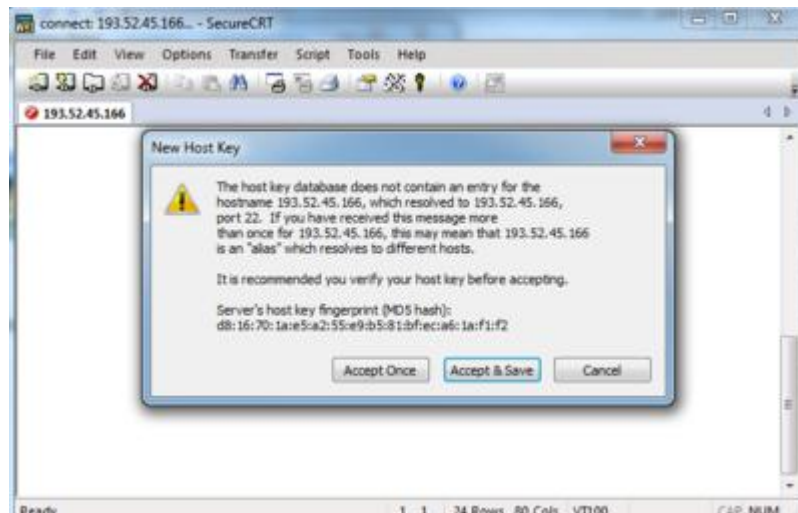


Figure 17: Connect via SSH

### 4.1.3 Tenant Life cycle

This section describes the tenant life cycle and the actions that have to be done by the IOs. Actually, there are a few scenarios in which a definition of a tenant life cycle is needed and all of them are related to the identification of fraudulent use of resources. As the creation of a tenant involves no use of “real resources”, the problem comes with the use of the user inside this tenant. There are 3 scenarios:

- **Tenant with no use:** In this case we have a tenant with or without resources, but after a predefined period of time, there is no use of any resource. This period of time could be fixed as 3 months but could be redefined for each IO depending of the availability of resources or the misuse of them. Irrespective of whether this tenant uses resources or not, the system sends an email to the owner of the tenant in order to inform about the situation. Depending the decision of each IO this message could give details about the lifetime ineligibility if no activities are detected after a defined period (predefined with 3 months but IO could change it depending of their own management resources).

In case that there are user(s), the administrator will send an email to each user informing the about the situation in order to correct it or in other cases proceed to release those resources. If those resources continue not to be used, the admin will automatically release them after a period of time. If this situation applies to all users, the admin will proceed in the same way as in the previous one with no users.

- **Tenant with a user with a black email account:** This is a special situation in which a user has been created with an incorrect email address. After some period of time the IO administrator detects that the email corresponds to an email generator and proceeds to include it into the email black list in order that it cannot be used. The IO notifies the tenant owner of the situation in order to correct it and not to repeat it in the future. There is also the possibility to delete the tenant if the situation continues in the future.
- **Tenants with fraudulent users:** In this case the IO administrator detects that a user is fraudulently using resources. The procedure is to send an email to the users in order to inform them to resolve the problem, or the IO administrator could deactivate the resources. In the same way a notification is sent to the tenant owner informing of the situation in order to resolve it and not repeat it in the future. If malicious use continues, the IO will deactivate the tenant and release the resources associated to the tenant.

#### 4.1.4 Traceability of deployed Instances

This section deals with the IO's capability to identify who has allocated a resource on the IO's infrastructure at a given time, in the present or past.

- Correlation between Instance and public IPs in real time

You can use the nova command `nova list --all-tenants` to list all IPs used and their matching instances on a region:

`nova list --all-tenants`

ID	Name	Status	Networks
a8009367-b905-4f7f-9ad2-b35cf155979b	Access_control	ACTIVE	ILB-MonitoringNet=192.168.0.7, 10.0.48.46
9a593b80-6502-4f33-83d8-18456541e7b6	CEP-PTRAK	ACTIVE	panos=192.168.3.4
96d829fb-6e46-4901-98aa-544d9a69c8b5	ConnectedTV-v1	ACTIVE	UC-FI-CNT2-PrivateNetwork=192.168.103.10
311a72ce-e675-4627-9c44-8b13c6e9af7b	Fire2FIPPP01	ACTIVE	fire4fipp-private-network=192.168.101.102, 193.52.45.137
47b00d4d-15aa-4c44-a3ac-3e9c9da91155	Fire2FIPPP02	ACTIVE	fire4fipp-private-network=192.168.101.100, 193.52.45.138
3a085423-0b32-4a2e-9030-ab03a30e7c76	Fire2FIPPP03	ACTIVE	fire4fipp-private-network=192.168.101.103, 193.52.45.139
ab5002a0-c462-41d8-9550-1a2c5e6e901	KURENT02	ACTIVE	panos=192.168.3.5
8a0f8b04-c816-4e47-9814-d88bb79add8d	LeCloudCestLavie	ACTIVE	
64a7cebd-f927-473b-a055-19b06c83fca3	MARKET-TEST	ACTIVE	panos=192.168.3.6
6bbcca3f-1213-4653-b202-d5e6e517d6cc	MARKET5	ACTIVE	panos=192.168.3.7
f667c3da-927f-4d97-88a8-b6b12c9c4304	MQTT_IAE	ERROR	
47096eb5-8030-4944-b622-010ae38ee95e	Monitoring_second_ext	ACTIVE	ILB-net01=192.168.234.2, 193.52.45.144
1970c750-fb87-4404-9a3d-4cb26b30e409	NAM_NPM_DEM	ACTIVE	
3df1192a-a076-457d-9f37-ff53b206f2a0	NAM_NPM_DEM	ACTIVE	ILB-MonitoringNet=192.168.0.2, 10.0.48.31
1fc44ab3-6ccf-496d-9151-009e8ebfcd0c	NGSI_CB	ACTIVE	ILB-MonitoringNet=192.168.0.4, 10.0.48.32
c42e259b-0d28-40b8-bc6b-0d6ddf9ee14	NGSI_CBORTON	ACTIVE	
4af6e993-8ba0-491a-a59a-41c0c718921a	Qt	ACTIVE	Public-External-Network=193.52.45.146
d154d8d3-0b02-445a-aa63-d34d1fc61e73	REGISTRY_PTRAK	ACTIVE	panos=192.168.3.2
308ea322-c547-41f4-a184-236443ffb6de	SecurityProbe	ACTIVE	ILB-MonitoringNet=192.168.0.6, 10.0.48.34
479a9fb5-2ff2-4cbb-ac25-ad12cc88acf	SecurityProbe	ACTIVE	
f37372b6-ec57-42b8-aa3b-109894233e59	T0482	ACTIVE	Public-External-Network=193.52.45.136
11407fc0-e19e-498d-b950-ad112b58ddf7	Ubuntu	ACTIVE	
c8439b5e-dfe5-4f93-887b-572e77c5c42	Ubuntu	ACTIVE	ILB-MonitoringNet=192.168.0.5, 10.0.48.33
937583ec-ae3b-4390-98aa-c9ca47042dfc	cdva_check	ACTIVE	Public-External-Network=193.52.45.145
d3ff59ca-623d-4fe9-b21e-ea47a2d2d14	complex-event	ACTIVE	Public-External-Network=193.52.45.135
beb092d-0a31-40cd-9955-a3e9d5146b3f	disposable_lanmion	ACTIVE	Public-External-Network=193.52.45.161
7fae8dd-81bc-4e22-aa7e-8543b1b32008	keystone-test_image	ACTIVE	ILB_Cluster_Network=192.168.102.104, 10.0.48.53
3931c3d0-a3a2-4ff5-9a72-8a31f457fb8e	kiwanoL01	ACTIVE	Public-External-Network=193.52.45.132
ca08c183-af02-4f98-bc10-3c47d2085ed5	mv1-v2	ACTIVE	UC-FI-CNT2-PrivateNetwork=172.16.26.15
9fcf78c3-7a1d-4995-b158-22afe146c3de	pajamakids	SUSPENDED	Public-External-Network=193.52.45.154
4a2d0151-212f-4b08-b735-20cd2e3a706	reperio-v1	ACTIVE	UC-FI-CNT2-PrivateNetwork=172.16.26.16
a13026de-7672-40f0-83bd-869d7f964590	solr-v1	ACTIVE	UC-FI-CNT2-PrivateNetwork=172.16.26.17
163d0c38-8723-4fa0-8f38-90aa42f3d6d0	test-erwan2-163d0c38-8723-4fa0-8f38-90aa42f3d6d0	ACTIVE	Network-Erwan_Test=192.168.222.2
eb1144b8-e990-4c04-9071-1b367e386dcb	test-erwan2-eb1144b8-e990-4c04-9071-1b367e386dcb	ACTIVE	Network-Erwan_Test=192.168.222.3
5250601f-8b3e-47ab-b486-b6889e7c3639	ubuntu	ACTIVE	Public-External-Network=193.52.45.157
e6912443-e086-4a6a-835e-e0645b00988c	ubuntu	ACTIVE	Public-External-Network=193.52.45.163

- Identification of Instance & Instance's owner to which a public IP has been affected

An IO is not in charge of the user database. This is managed by the administrator of the IDM component. On the time of the release of this document, the IDM component is only deployed in Spain. As a consequence, User information is subject to Spanish law and according to this, user data cannot be disclosed except if requested by legal authorities. In case of a malicious usage of a Public IP there is no possibility for an IO to obtain the related user data other than by starting a legal case against the malicious user.

#### 4.1.5 Local catalogue management

Currently, the images that have to be located on each local catalogue come from the Spain node (FIWARE catalogue). For the purpose of copying the various images to the rest of IO, we have installed an Apache server in the Spain glance instance controlled by user and administrator password in order to access to it and to download the corresponding images, as described below. The server is located at <https://glance.lab.fi-ware.org>.

Currently, there are only users for Lannion, Waterford, Berlin and Trento. If any other IO wants to access it, he should contact the administrator of this server (Fernando López, fernando dot lopezaguiar at telefonica dot com). In the following table you can see the responsible of each IO

The list of persons with authorized access per IO is the following:

Nodes	Responsible	Account
Lannion	Riwal Kerherve	glance2
Lannion	engineering@imaginlab.fr	glance5
Berlin	Thomas Günther	glance4
Trento	Cristian Cristelotti Coll	glance3
Federator	Joe Tynan	glance1

*Table 2: List of users with access to the glance-apache server in Spain*

A personalized email was sent to the users with the password to be used for the server. The administrator of this server maintains the correlation between users and password for all users.

We are evaluating the alternative that the Spain administrator node will automatically update the images for the other nodes but we are still in the process of defining the procedure to do it.

#### Images to be added to the local catalogue

A list of cloud images is required to be present on the catalogue.

- repository-image-R3.2-2
- dbanonymizer-dba
- marketplace-ri\_2
- meqb-image-R2.3
- cep-image-R2.3
- datahandling-ppl
- orion-psb-image-R3.3
- registry-ri
- ofnic-image-R2.3

- kurento-R4.2.2
- kurento-image-4.0.0
- kurento-image-R3.3
- cdva-image-R2.3

NID is a property metadata and anytime we create a new image, this NID number need to be associated to its glance metadata.

### Procedure to add the required images

Below we show at the example of the dbanonymizer-dba how the required images can be added. The full information on all images is given in **Error! Reference source not found.**

- dbanonymizer-dba:
  - Public: Yes
  - Protected: No
  - Name: dbanonymizer-dba
  - Status: active
  - Size: 3339124736
  - Disk format: qcow2
  - Container format: ovf

```
$ glance image-create --name dbanonymizer-dba --disk-format qcow2 --container-format ovf --min-disk 0 --min-ram 0 --is-public True --is-protected False --property nid=64 --file <name of the file of the corresponding downloaded image>
```

#### 4.1.6 Managing Images

If you need to manage an image, i.e. modify it permanently, you can use guestfish **Error! Reference source not found.** If you want to mount an image with read-write mode as root, use the following:

```
guestfish --rw -a <my_image.img>
```

You should then get a `><fs>` prompt. First thing to do then is to type "run" which will launch a virtual machine used to perform all the file manipulation. You can now list file systems with the `list-file systems` command.

```
><fs> run
```

```
><fs> list-file systems
```

```
/dev/vda1: ext4
```

```
/dev/vg_centosbase/lv_root: ext4
```

```
/dev/vg_centosbase/lv_swap: swap
```

Then mount your selected fs:

```
mount /dev/vda1 /
```

And then you can operate inside your image. When you're done, just type `exit` to leave the guestfish tool. You can now use your modified image file.

#### 4.1.7 Managing Blueprints

The management of Blueprint is based in the utilization of the PaaS Manager together with the SDC Manager. The corresponding recipes have to be incorporated into the SDC Recipes Catalogue in order to make use of these functionalities. These recipes currently are based in chef distribution programme. Nevertheless a new version of the SDC is being deployed in the Spain Node which allows the



instantiation both Chef and Puppet recipes. In the following paragraphs we see the normal management operations over blueprint.

### Create a blueprint template.

First of all you must create a blueprint template or take a predefined template previously defined in the catalogue. If we decided the first option, you should click on the "Blueprint Templates" button, then click on "Create New Template" (see screenshot below).

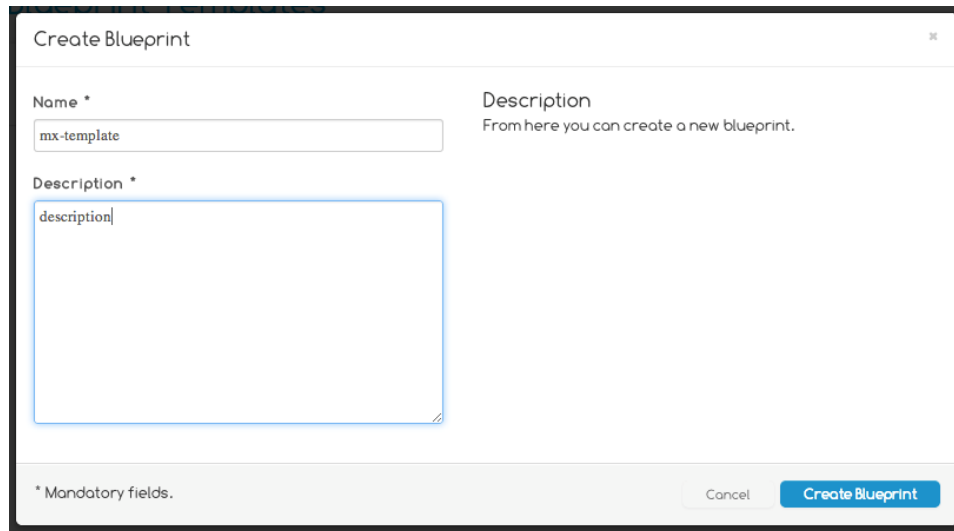


Figure 18: Create blueprint template

You have to complete the information related to the name of this template and a description in order to know afterward what the purpose of this template was. Then you should click on the “Create Blueprint” button to finalize the creation of the template.

If you decide to take one template from the catalogue, you should click on the "Blueprint Templates" button and then click on "Open Catalog". This shows you a list of predefined templates, take the one and click on the “Clone” button. This will create for you a new Blueprint template to work with.

### Adding Tier(s) to your blueprint template

Secondly, if we want to add some Tiers to our blueprint template, we should click on the “Actions” button or over the right button of the mouse in order to go to the windows to add/edit/delete Tiers associated to this blueprint template (see the screen shown below).



## Blueprint Templates

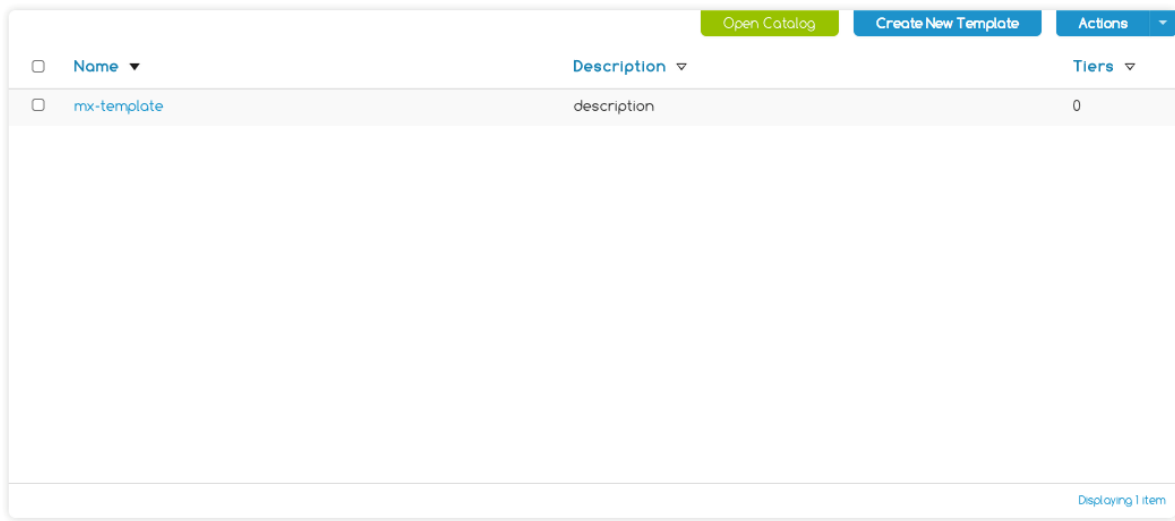


Figure 19: Adding tier(s) to a blueprint template

If we want to add a new tier, click on the “Add Tier” button to open a new window (see the screen shown below). It is a modal window in which you need to select the appropriate data. The marked attributes are mandatory. The selection of the Regions means that this Tier will be deployed in those regions. The data of flavours, Images and keypairs correspond to the data contained in the selected region (by default Spain Node). It is not mandatory to select an icon to represent the purpose of the tier but it is a good practice to know in a simple view what we are doing on this tier. Last but not least, you should indicate the minimum, maximum and current number of instances to be deployed using this template. This is made in the circle located to the left of the window (see the screen shown below).

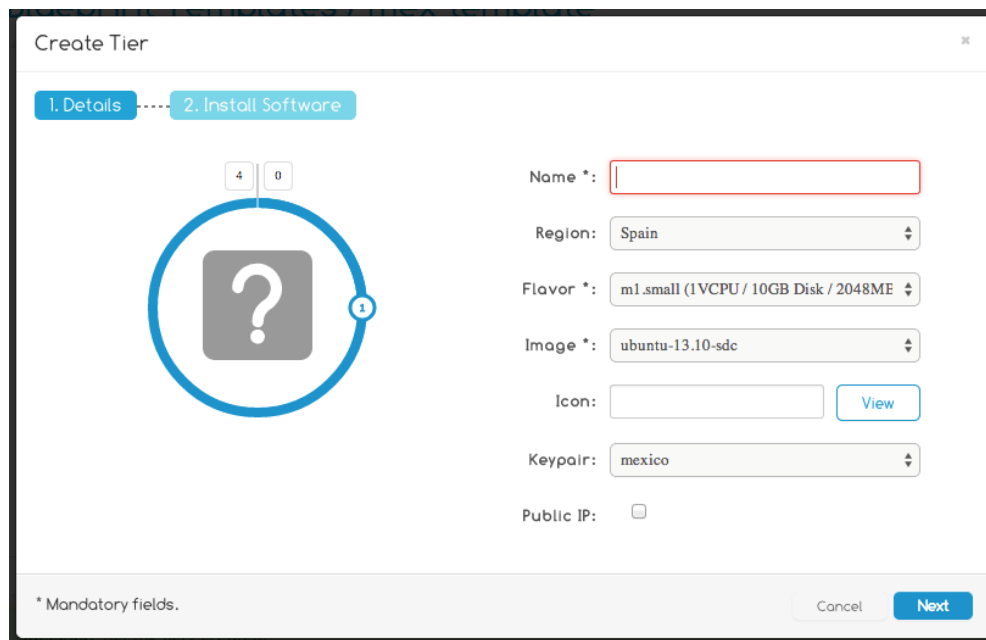


Figure 20: Create a tier

If we finish the introduction of data, we should click on the “Next” button, which moves to the next window in which we can select the corresponding recipe(s) to be installed on these instances. It corresponds to the list of Software Catalogue included in the SDC Manager. We can drag & drop from

the list of “Software in Catalogue” and translate it to the “Software in Tier” list (see screenshot below).



Figure 21: Adding software to a tier

To finish the template, click “Create Tier”.

#### Editing/Creating the software attributes.

In some cases, the software to be installed has an attribute or a group of attributes (ports, installation directory, and so on) that they are leaving by default. By contrast, if you want to change them, it is possible by clicking over the right button of the mouse over the selected software on the “Software in Tier” list (see the screen bellow) and click on “Edit Attributes”.

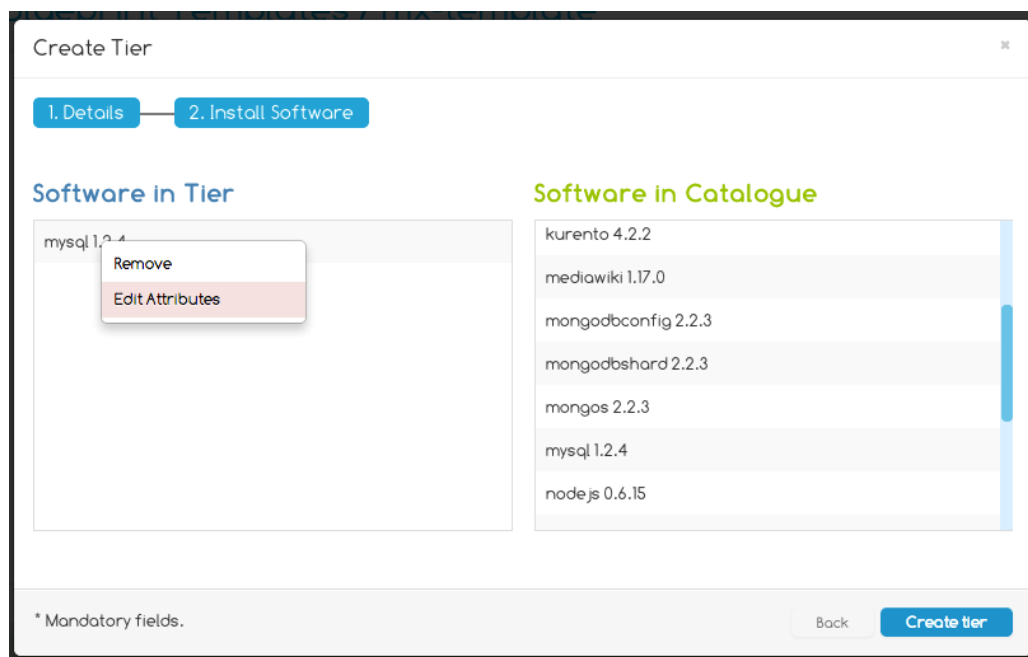
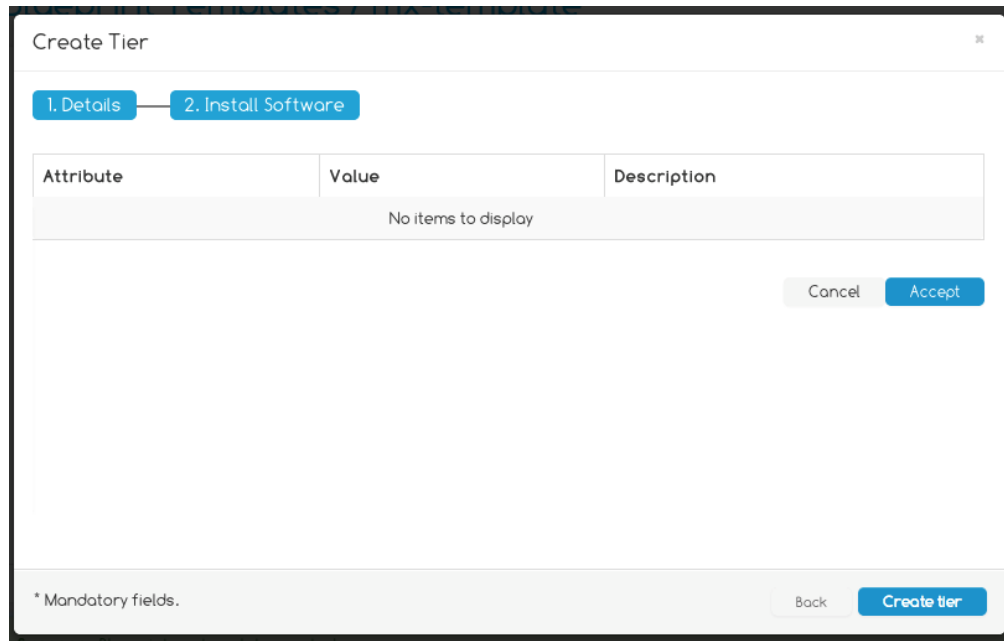


Figure 22: Selecting the menu to change the software attributes

This shows a modal window in which we can introduce the attributes to be used for the installation of the software (see the screen below). Please, refer to each product in order to know which the attributes for each case are.

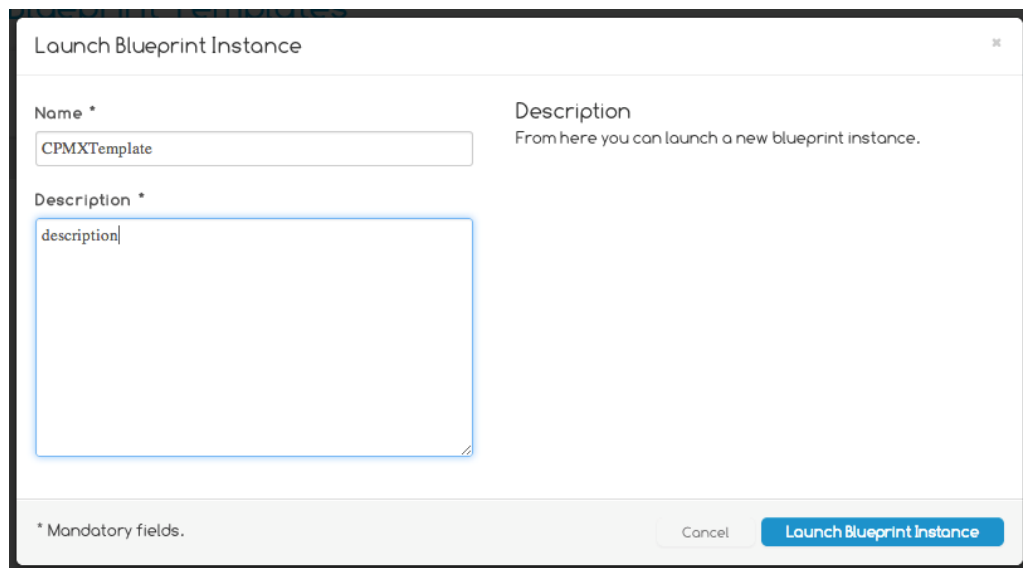


The 'Create Tier' modal window has two tabs: '1. Details' and '2. Install Software'. The '2. Install Software' tab is active, showing a table with columns 'Attribute', 'Value', and 'Description'. The table is currently empty, displaying 'No items to display'. At the bottom right of the table area are 'Cancel' and 'Accept' buttons. At the bottom of the modal, there is a footer with '\* Mandatory fields.', a 'Back' button, and a 'Create tier' button.

Figure 23: Editing the software attributes

### Launching an instance.

After the creation of a blueprint template, if we want to launch it, we should click on the “Action” button, see Figure 19. It shows a menu with the option “Launch Instance” in which we click on in order to launch it. It shows a screen in which it asks us about the name of the blueprint instance and a brief description of it (see the image below).



The 'Launch Blueprint Instance' modal window contains two input fields. The 'Name \*' field has the text 'CPMXTemplate' entered. The 'Description \*' field has the text 'description' entered. To the right of the description field, there is a text label 'Description' and a sub-label 'From here you can launch a new blueprint instance.' At the bottom of the modal, there is a footer with '\* Mandatory fields.', a 'Cancel' button, and a 'Launch Blueprint Instance' button.

Figure 24: Launch a blueprint template

If we have finished the introduction of data we should click on “Launch Blueprint instance” in order to launch it. The screen changes to the main windows in which we can see the different states of the instantiation process (see the image below).

- Deploying the required infrastructure (deploying).
- Installing the selected products (installing)
- Installed (installed), which corresponds to the final status.

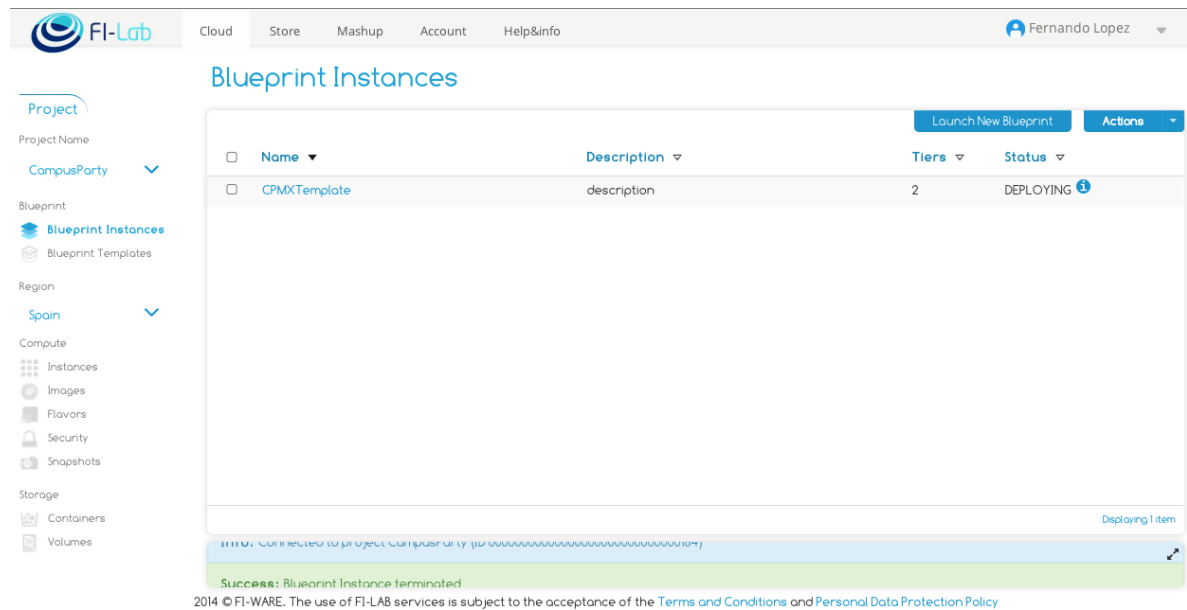


Figure 25: Blueprint instances

## 4.1.8 Use Case Handling

### Information to get

When a Use Case should be deployed within a region, the person in charge of the Use Case needs to contact the IO of the node if the quota applied to the node does not fit the needed dimensioning. Indeed, most of the time, quota and flavours on a node are quite limited to prevent abuse. These can easily be changed or extended by the IOs, but it requires manual intervention by the IO.

Below is a list of the information that an IO needs in order to adapt quota, flavour or configuration applied to a tenant.

- Global Architecture presentation from the Use Case
- Description of Instances:
  - Numbers of instances needed
    - For each of them, give the dimensioning needed:
      - Memory,
      - Disk,
      - Number of processors
      - etc.
  - Snapshot availability in case of migration from an existing server to XIFI
- Network connectivity:
  - Number of network interfaces per instances,
  - Configuration of the interfaces wanted,
  - Ports to be opened:
    - Management ports: Give the public IP needed in order to do the filtering
    - Service ports that will need to be opened (Service provided by the Use Case)
- Login:

- FI-Lab login ID (in order for the IO to add the ID to the granted list of users of the Use Case tenant)
- Snapshot login/pwd

#### 4.1.9 Tenant customization

##### Quota

- You might need to list your default quotas, so use the following commands (respectively for compute/network/block storage):

```
nova quota-defaults
quantum quota-show
cinder quota-show
```

- If you need to update a quota for a particular tenant, use the following commands:

```
nova quota-update --<quotaName><quotaValue><tenantID>
quantum quota-update --tenant_id<tenantID> --<quotaName><quotaValue>
cinder quota-update --<quotaName><quotaValue><tenantID>
```

- Some examples:

```
nova quota-update --ram 8192 <Tenant_ID>
quantum quota-update --tenant_id<Tenant_ID> --network 5
cinder quota-update--gigabytes 50 <Tenant_ID>
```

##### Flavors

- You might need to list your default quotas, so use the following command:

```
novaflavor-list
```

- In case you want to add a new flavor, just 2 steps are needed:
  - Create your new tenant:

```
novaflavor-create --is-public
<true|false><flavor_name><ID><ram><disk><vcpus>
```

E.g.:

```
novaflavor-create --is-public false Test-flavor auto 2048 0 2
```

- Then your freshly created flavor needs to be available for the tenant, use the following command:

```
novaflavor-access-add <flavor><tenant_id>
```

E.g.:

```
novaflavor-access-add a5abb478-9672-46f1-979e-99d2ca023fdc
00000000000000000000000000000000xxxx
```

#### 4.1.10 Node administration

**Levels of administrative access (users, local admins, federation admins):**

By joining the federation some administrative tasks are delegated to the master node, in particular identity management and authentication of OpenStack management actions. In consequence, some administrative commands (e.g. fetching user information or tenant information for all tenants) are no longer admitted for nodes but require collaboration with the federation (e.g. with the federation maintainer in case of maintenance procedures). This is a restriction imposed by federating, is a part of the operational level agreement between node and federation and causes a number of inconveniences including the need for continuous exchange of “service catalogues” and “authentication tokens” across the federation network infrastructure for almost all actions.

It could be disconcerting, but since the keystone component of OpenStack has been replaced by federation components (the keystone proxy and the IDM), some of the administration tasks that an IO needs to do on his node cannot be done anymore after joining the federation. The management of the user database is then removed from the tasks that an IO usually manages on his node and it is moved under the responsibility of another stakeholder: The Federation Maintainer (for roles c.f. section **Error! Reference source not found.**).

- Grants on the node (API, Cloud portal)

To administrate a node, three different types of access are available.

- CLI: This is the most common way to manage a node. This is a SSH connection on the controller that provide your CLI interface.
- OpenStack API: OpenStack provides a normalised API to manage its cloud. These APIs (Nova API, Cinder API, Quantum API, Swift API, Glance API) can be made accessible via the Internet or not. For the federation to work, these APIs must be openly accessible at least for cloud portal requests.
- Cloud Portal: It has access to the different OpenStack API (Nova, Cinder, Quantum, Swift, Glance) that are made accessible from the node. This access provide only a basic administration: user oriented.

- Roles

Here the level of administrative access, depending on the type of user, is defined.

- Users

- Have basic access to tenants that he created.
- Manage and administrate their tenants through cloud portal. It is the only administrative access they can have.
- Manage and grant access of their own tenants for other users.

- Local admins

- Is an IO, and is in charge of administration of its own node (IO)
- Has basic access to his tenants through the cloud portal like other users has.
- Has CLI access to the node and can administrate Nova, Glance, Swift, Quantum and Cinder

- Federation admins

- Manage keystone proxy and IDM
- Administrate users and tenants

## Procedures:

- How to change the administrative level for a given user
  - Privilege on a tenant created by an IO

An IO can only manage a tenant he created. To manage a tenant, go to the Identity Manager and click on the arrow near the name of the user (Figure 26). Then choose "switch session" and click on the tenant you would like to modify roles for some users.

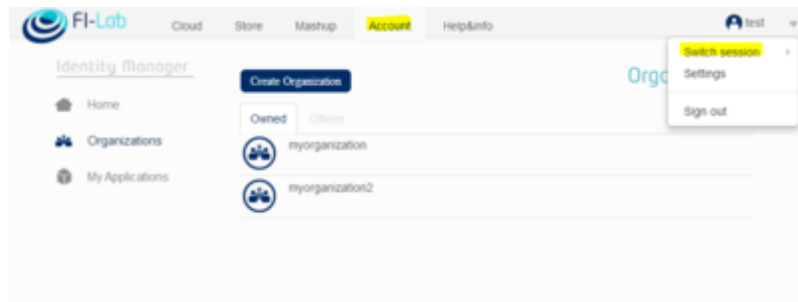


Figure 26: Manage tenant – select user

Click on members and then do the modification you would like to do, e.g. adding users or adding roles to a certain user.



Figure 27: Modifications on a tenant

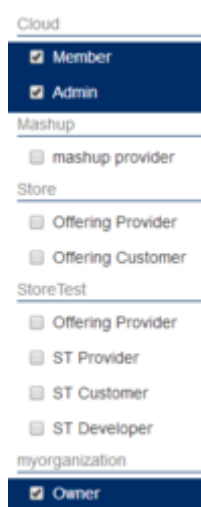


Figure 28: Modifications on a tenant II

- Privilege on a tenant not created by an IO

Please note that only persons in charge of IDM and the federation maintainers have the privilege to manage users on these tenants

- How to list tenants and users on a node:

The command below permits you to list all tenants in a given node

```
# sourceopenrc (to have nova rights)
# nova--os-region Lannion usage-list
```

See Table 3 for an example from Lannion node, 2014-07-15 - 2014-08-13.

Tenant ID	Instances	RAM MB-Hours	CPU Hours	Disk GB-Hours
00000000000000000000000000000009	2	53271.77	104.05	0.00
00000000000000000000000000000049	1	64407.91	125.80	0.00
000000000000000000000000000000356	2	5505024.48	2688.00	53760.00
0000000000000000000000000000002559	1	1376256.12	672.00	13440.00
0000000000000000000000000000002983	11	6720124.52	4993.41	54212.24
0000000000000000000000000000002988	3	2578811.02	1259.19	25183.70
0000000000000000000000000000003437	5	11010048.97	5376.00	107520.01
0000000000000000000000000000003449	9	33405.01	62.28	19.77
0000000000000000000000000000003478	3	24772610.17	8064.00	161280.01
0000000000000000000000000000003847	1	11010048.97	5376.00	107520.01
0000000000000000000000000000003851	1	344064.03	672.00	0.00
0000000000000000000000000000003940	1	195.70	0.10	1.91
0000000000000000000000000000003965	6	269624.50	388.78	918.84
0000000000000000000000000000003997	26	5758557.76	3816.22	0.00
0000000000000000000000000000004004	9	4930476.50	2411.06	58553.86
0000000000000000000000000000004012	1	344064.03	672.00	0.00
0000000000000000000000000000004019	1	344064.03	672.00	0.00
0000000000000000000000000000004098	1	2752512.24	1344.00	26880.00
0000000000000000000000000000004287	3	74.67	0.15	0.00
0000000000000000000000000000004291	1	1277297.91	623.68	12473.61
0000000000000000000000000000004351	2	595610.87	290.83	5816.51
38aec686f107485ebc1ca9763d96d958	5	6881280.60	3360.00	67200.01

Table 3: Lannion usage list

The command below permits you to list all VM created on your node as well as the name of the user who created it. Table 4 shows an example result.

```
# nova-manage vm list
```

instance	node	type	state	launched	image	kernel	ramdisk	project	user	zone	index
LeCloudC estLaVie	node-3	m1.s mall	active	22/04/2014	760d4409- 731c-4009- b368- 4a7ad78d83 35	38aec686f 107485eb c1ca9763d 96d958	f7b2f1315c4a47b284aa142fb0728d43			None	0
CEP- PTRAK	node-3	m1.m edium	active	21/05/2014	32f0120d-7533-4d3f-a7c4-0e492b93b740			3437	ptrak- syn	None	0
KURENT O2	node-1	m1.m edium	active	21/05/2014	25c3b46b-a91c-4bfb-8fd2-dc3d46858e57			3437	ptrak- syn	None	0
Fire2FIPP P01	node-5	fire2fi ppp	active	27/05/2014	dd5859f2-fbfa-4d12-9eac-2ec306685a75			3478	smorant	None	0
Fire2FIPP P02	node-5	fire2fi ppp	active	13/06/2014	b6d7fe2c-ee42-4e80-9978-d01a35f32d21			3478	smorant	None	0
Connecte dTV-v1	node-8	fi- cnt2- ctv	active	29/08/2014	88df7e0b-3cc6-4620-9e19-e76b832efb66			4004	smorant	None	0

Table 4: VM list



## 5 HARDWARE DEPLOYMENT

Based on [7], we've gathered the models of hardware deployment for use in different scenarios.

### 5.1 Deployment Architecture Reference Model

This section discusses the reference model for the physical and software deployment of a XIFI node based on OpenStack Grizzly, FI-WARE add-ons and XIFI tools. It is important to understand that there is no one model fits all, since the deployment architecture depends on, and is heavily related to resources available in an infrastructure. Dealing with existing infrastructures that connect to the federation has the impact that the deployment architecture must be adapted according to the existing hardware and to the planned upgrade of the infrastructure.

The following text is largely inspired by best practices in the deployment and operations of OpenStack based-clouds [8][9].

#### 5.1.1 Concepts

##### 5.1.1.1 Physical Equipment

In the deployment of a cloud-based data centre we deal with interconnected physical equipment that composes the physical architecture of the data centre. The most important equipment types for the definition of the deployment architecture are:

- **Rack:** Modern servers and network equipment are designed to be installed in a framework called a rack. A rack contains multiple mounting slots called bays, each designed to hold a hardware unit. Hardware may occupy more than one unit. Recent evolution for high-density servers, introduces blade servers that are hosted in a blade (which allows packing several hardware component in a blade enclosure). Blade enclosures are mounted within racks.
- **Server:** A server is a node in the data centre (usually hosted in a rack) that offers computation and storage capacities. A server node in a cloud-based data centre may have different role according to his hardware configuration, and hence being able to host different services (that correspond to a given role). Generally speaking, server equipped with large number of CPUs and RAM are more efficient for computational tasks, while server equipped with large amount of hard drives are more efficient for storage tasks. This discussion will become clearer in the next paragraphs that discuss node roles in an OpenStack based-cloud.
- **Switch:** Hardware equipment that allows the physical interconnection of different server nodes. Like a server, a switch may have different roles according to the network services it provides (e.g. management network or data network).

##### 5.1.1.2 Node Roles

In a cloud environment, servers usually have different roles. In the following discussion we take into consideration roles usually adopted in OpenStack deployments. These roles are:

**Controller (node).** A controller node provides the central management for multi-node OpenStack deployments.

**Compute (node).** A compute node provides the computational capacity (i.e. virtual machines) to OpenStack deployments.

**Block storage (node).** A block storage node provides non ephemeral storage for virtual machines.

**Object storage (node).** An object storage node provides access to large storage solutions via Web

APIs.

**Object proxy (node).** A proxy that distribute the objects to different storage nodes according to replica settings and region availability settings.

**Network management (node).** A network management node provides (dynamic) configuration on the VLANs that interconnect the VMs.

Furthermore XIFI will consider the following roles:

**Load balancer (node).** A node that in high-availability configurations, provides load balancing of requests among the available redundant services.

**Monitor (node).** A monitor node provides monitoring of resources included in a XIFI node.

**Deployment (node).** A deployment node provides the ability to control the deployment of a XIFI node, including a monitor node and all other nodes needed to run OpenStack and FI-WARE extensions.

It is important to underline that a node may serve different roles according to the OpenStack services it runs. Given the difference of type of service, different roles may perform better on different type of hardware. Accordingly, certain roles should not be covered by the same machine in a well-designed cloud deployment. In the next paragraphs we discuss quickly the different services. The distribution of services on actual nodes defines the role of a node and the architecture of the OpenStack deployment, according to the type of configuration of the services.

### 5.1.1.3 OpenStack Services

The XIFI installation of OpenStack considers the following services distributed on the nodes [10]:

**Nova** [11]: Provides the management of computational resources. It includes three basic services: the scheduler, to define where the VM will be allocated, an API to remotely control the scheduler, the compute service that actually provides the VM on the single nodes and other support services.

**Neutron** [12]: provides network management for OpenStack. It includes the following services: server to manage the network as service functionality for Nova, agent to apply the configuration of the single nodes, DHCP-agent to automatically assign IPs to VMs, and other services. It requires specific plugins to configure the different network apparatus (e.g. OpenVSwitch [13]).

**Glance** [14]: provides image management for OpenStack. It includes the following services: a registry that provides a catalogue of available VM templates and an API to control the services. Different back end are available for glance [15].

**Keystone** [16]: provides identity and service registry functionalities.

**Cinder** [17]: provides block storage (i.e. volumes) functionalities for OpenStack. It includes three basic services: the scheduler to define where the volume will be stored, an API to remotely control the scheduler, and the volume service that actually provides the storage;

**Swift** [18]: provides object storage functionalities. It includes the following services: the proxy to accept API requests and to route them to storage services, the object storage that take care of the actual storage.

**Horizon** [19]: provides a graphical user interface to support management and self-provisioning of cloud resources for the services mentioned above.

### 5.1.1.4 Network Services

As mentioned above, Neutron requires an actual plugin to be able to configure switches and creating

VLANs in an OpenStack cluster.

**DOVE** : The reference plugin for XIFI is a customized version of IBM's Distributed Overlay Virtual Ethernet (DOVE), provided by FI-WARE. DOVE is an SDN management solution for data centres that allows traffic shaping inside the data centre. It is based on OpenVSwitch.

**OpenVSwitch**: as an alternative; we foresee the adoption of the standard version of OpenVSwitch.

Other network services are required to support the inter-node XIFI connectivity. These are currently under development. More details will be provided in the next version of this guide.

#### 5.1.1.5 Other Services

XIFI deployment will require other services:

**HAProxy**: to provide load balancing across OpenStack and FI-WARE APIs in the high-availability configuration.

**XIFI Monitoring Management Middleware**: a middleware that is currently under development in XIFI to integrate physical and virtual infrastructure monitoring data collected from the nodes. The XIFI Monitoring Management Middleware provides adapter mechanisms for monitoring tools adopted by infrastructures (e.g. OpenNMS [\[20\]](#), Perfsonar [\[21\]](#)).

### 5.1.2 Physical Deployment Models

The physical architecture of a node influences the software architecture and QoS characteristics such as availability of services. Servers are usually hosted in racks and if all servers, for example, playing the role of a controller are in the same rack and power to the rack is interrupted, the cluster may not be available externally even though other services may be still running in other racks. Similar issues apply in case of switches. Therefore when possible, it is better to plan the physical architecture without a single point of failure.

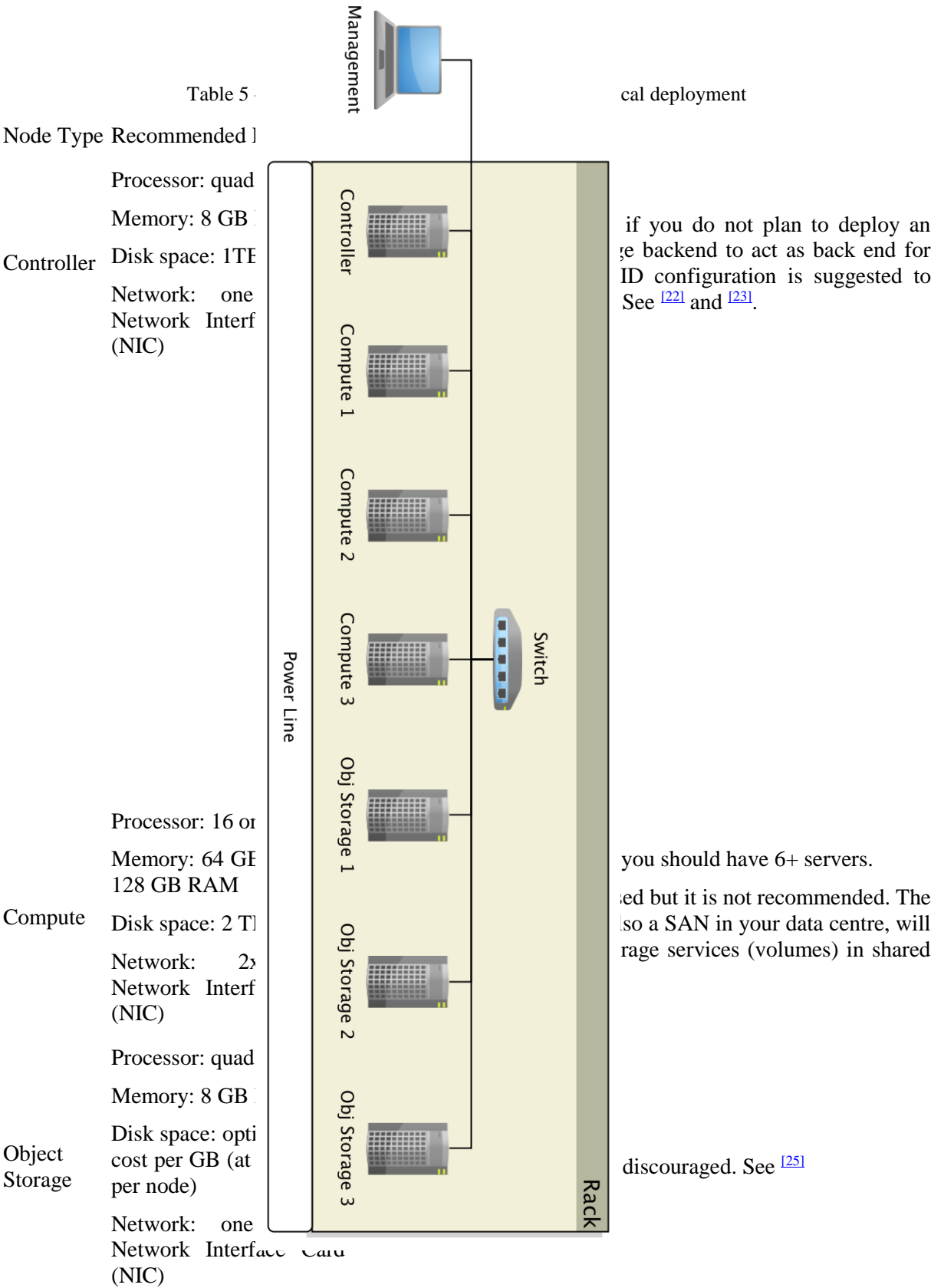
#### 5.1.2.1 Basic Physical Deployment

In a basic physical deployment, resources are not redundant and are not made resilient. In the simplest case, we will have a rack (or more racks) with a single power source that will power all servers part of the node including the switches that connect the servers. In the simplest configuration this requires:

- 1 controller node
- 3+ compute nodes
- 1 manager node
- 1 switch 24 port (OpenFlow enabled)
- Optionally, we can include as well:
- 3+ object storage nodes

Such flat configuration is not recommended for production nodes, unless high-availability deployment cannot be achieved. Production nodes should refer to the high-availability deployment. Corresponding service architecture deployment is discussed separately.

Figure 29 - Basic Physical Deployment



### 5.1.2.2 High Availability Physical Deployment

In a high availability physical deployment, resources are redundant and they are located to be resilient. The objective of high availability deployment is to minimize:

- System downtime — the unavailability of a user-facing service beyond a specified maximum amount of time, and
- Data loss — the accidental deletion or destruction of data.

To avoid system downtime and data loss it is important to avoid the presence of single point of failure. Either in the hardware or in the software. In this section we highlight the deployment from the hardware perspective.

We assume to have two (or more) racks where the nodes are replicated with separate line power supply. This will ensure that if a power line will go down and hence turn off a rack, the second power line will be still accessible. As better alternative it is possible to consider single racks with support for 2 independent power lines. In this case all equipment in the rack should be equipped with 2 power supply units attached to the 2 power lines of the rack. This reference configuration requires:

- 2+ controller node
- 6+ number of compute nodes
- 3+ object storage nodes
- switch 24 port 1GB and 10GB up-link (OpenFlow enabled)
- 1 manager node (also a laptop may do the work)

This reference configuration is the recommended one for XIFI nodes. Tweaks may be applied according to specificity of XIFI nodes.

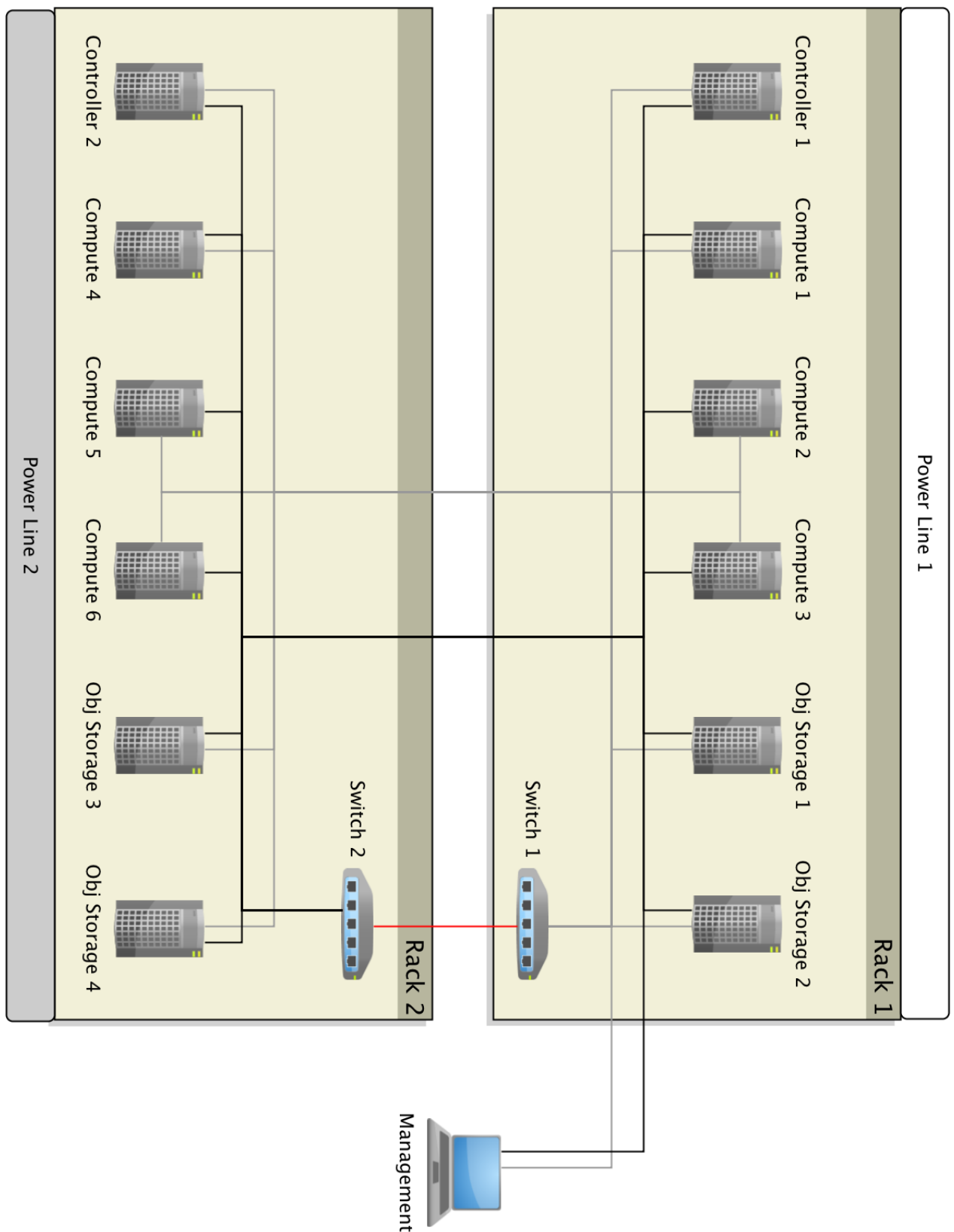


Figure 30 - High availability physical deployment

	<p>Memory: 12 GB RAM</p> <p>Disk space: 1TB</p> <p>Network: 2 x 1 GB Network Interface Card (NIC)</p>	<p>Storage of other storage backend to act as backup site for the VM registry (2TB). RAID configuration is suggested to increase controller reliability. See <a href="#">[22]</a> and <a href="#">[23]</a>.</p>
Compute	<p>Processor: 16 or 32 cores</p> <p>Memory: 64 GB RAM or</p>	<p>If you adopt a 16 core server, you should have 12+ servers.</p> <p>RAID configuration can be used but it is not recommended. The disk</p>

	128 GB RAM Disk space: 2 TB Network: 2x1 GB Network Interface Card (NIC)	space, unless you have also a SAN in your data centre, will be as well used for block storage services (volumes) in shared modality. See <a href="#">[22]</a> and <a href="#">[24]</a>
Object Storage	Processor: quad core Memory: 8 GB RAM Disk space: optimized for cost per GB (at least 4TB per node) Network: 2 x 1 GB Network Interface Card (NIC)	RAID configuration is highly discouraged. See <a href="#">[25]</a>

### 5.1.3 Services Architecture Deployment Models

In the previous section we discussed the physical deployment and listed the nodes type needed for that. But we didn't enter in any details regarding the services to be deployed on the nodes. Depending on the selected architecture, the different nodes will support different roles [\[9\]](#).

#### 5.1.3.1 Basic Architecture

In the basic deployment services are not configured in high-availability. In this section we details which services are supposed to run on the different nodes discussed in the section #Basic Physical Deployment. It is important to underline that we foresee the computational node to cover as well the block-storage node role through the set-up of a shared file system (e.g. NFS). Also, we foresee the installation of XIFI specific services on the controller node, if the node offers enough capacity to run them.

The controller node will host all the services related to the management of the XIFI node. The services include:

- The nova-scheduler service, that allocates VMs on the compute nodes.
- The cinder-scheduler service, that allocates block storage on the compute nodes.
- The glance-registry service that manages the images and VM templates. The backend for the registry maybe the controller node, or the Object Storage if included in the deployment architecture.
- The neutron-server service that manages the VM networks.
- The swift-proxy service (optional) that manages request to the object storage nodes.
- The nova-api service, that exposes the APIs to interact with the nova-scheduler.
- The cinder-api service, that exposes the APIs to interact with the cinder-scheduler.
- The glance-api service, that exposes the APIs to interact with the glance-registry. If the object storage nodes are deployed, we recommend their usage as back-end for glance [\[15\]](#).
- The keystone service that manages OpenStack services in a node.
- The horizon service, that provides a dashboard for the management of OpenStack in a node.
- The IdM GE service, that provides identity management for users.

- The SLM GE service, that provides scalability and elasticity management. it is connected the SOM GE service, hosted in the Main XIFI node.
- The XIFI-MMM service, that collects monitoring data for physical appliances.
- The DCRM GEs is not listed as it is essentially a plugin to nova-scheduler and neutron.
- The compute node will host all the services related to the provisioning of VMs and block storage. The services include:
  - The nova-compute service that manages VMs on the local node.
  - The cinder-volume service that manages block storage on the local node.
  - The neutron-agent service that manages VM networks on the local node.
- The object storage node (optional) will host all the services related to the provisioning of object storage. The services include:
  - The swift-account-server service, that handles listing of containers.
  - The swift-container-server service, that handles listing of stored objects.
  - The swift-object-server service, that provides actual object storage capability.

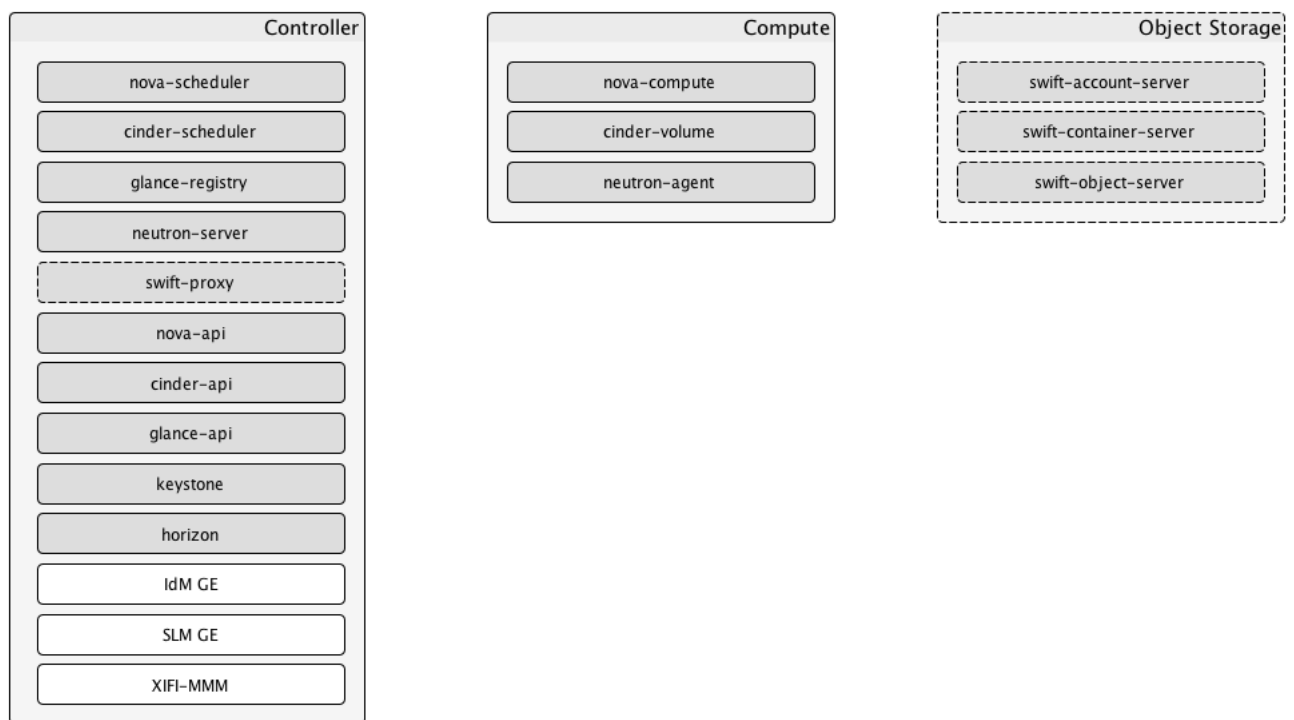


Figure 31 - Service per node in the basic architecture deployment model

### 5.1.3.2 High Availability Architecture

In the high availability services are redundant and they are located to be resilient. In this section we details which services are supposed to run on the different nodes discussed in the section #High Availability Physical Deployment. In this section we discuss the deployment from the software perspective. The deployment, except the injection of services to support high-availability of the controllers (the other are in high-availability modality by default so to say), is very similar to the basic one. In fact in OpenStack, computational, block storage and object storage nodes, are handled by the different scheduler to provide high-availability. The issue is to guarantee high-availability as well to



the controller.

Generally speaking, high-availability can be provided in two modalities:

- active/passive: in this configuration, a main controller operates the resources and in case of a problem, it switches the request to a backup controller.
- active/active: in this configuration, a number of controller operates the resources at the same time, in case of a problem to a controller instance, requests are not issued anymore to that node.

In this paragraph we refer to the active/active configuration.

The controller node will host, additionally to the services mentioned in the previous section, the services needed to ensure the high-availability of the controller node:

- ha-proxy service, that provides load balancing across OpenStack and FI-WARE APIs in the high-availability configuration.
- pacemaker service [\[26\]](#), that provides high-availability for neutron and other services.
- galera service [\[27\]](#), that provides high availability for databases used by the different services.
- RabbitMQ service, present as well in the basic deployment, should be configured for high-availability policy support.

More information is available in [\[28\]](#)

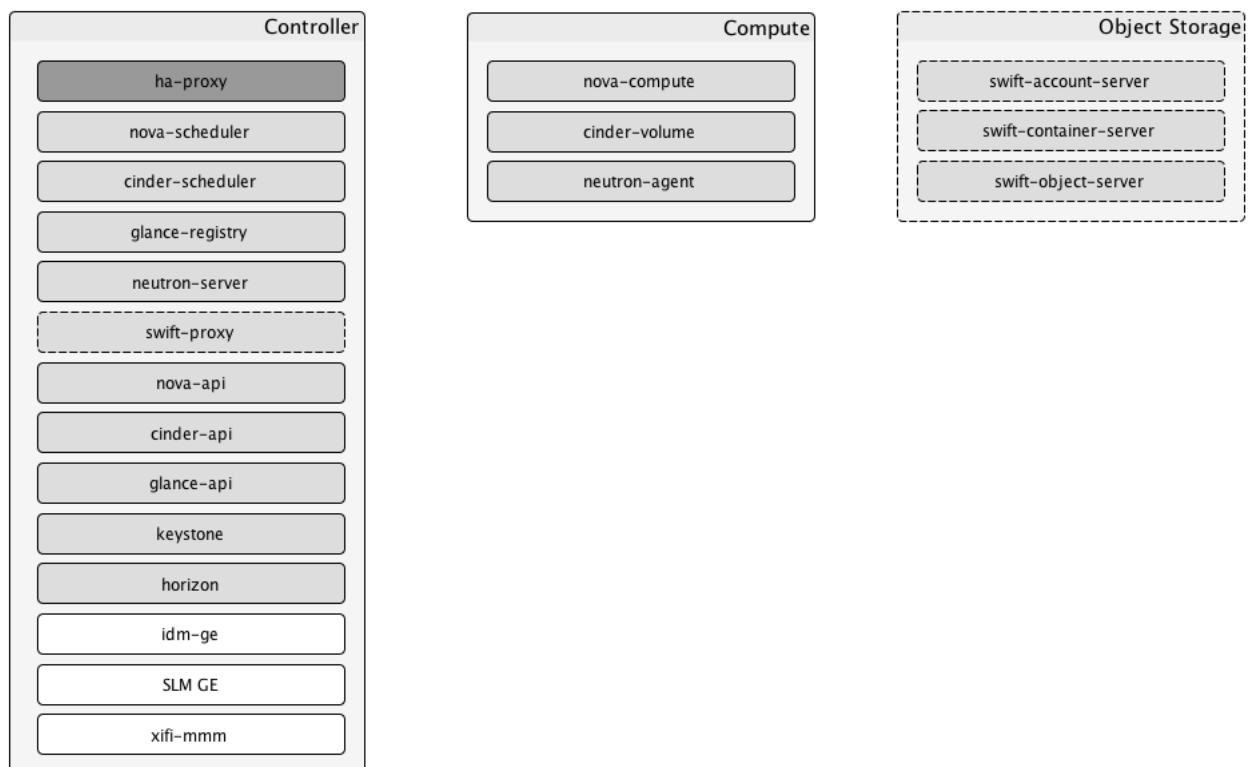


Figure 32 - Service per node in HA architecture deployment model

### 5.1.3.3 Block Storage Configuration

Different modalities to run block storage services are possible in OpenStack refer to [\[9\]](#) [\[29\]](#) for a complete discussion. In this section we refer to the default

configuration selected for XIFI, that relies on shared file system across the compute nodes. In this configuration, each compute node has a large storage capacity that is share through a distributed file system that allows the disks of the different compute node to be seen as a single drive. This configuration allows for high scalability and easy live-migration and does not require for dedicated nodes to the block storage. Of course, the solution may have drawbacks such as high network i/o in case the block is accessed from a remote virtual machine instead than locally. The default shared file system solution foreseen in XIFI is NFS.

## 6 SOFTWARE DEPLOYMENT

### 6.1 IT Box

The ITBox supports the automated installation of the main components of a XIFI node. The download version and some configuration parameters will be provided by the portal, following the registration of the new node. Monitoring and network adapters will be included (or linked) in the ITBox distribution from the adapters repository, while the same applies to GEs and related software needed to complete the XIFI node installation.

#### 6.1.1 Installation Manual

ITBox is distributed as an ISO image which contains an installer for ITBox Master Server. The ISO can be installed in the same way, using a virtualization software package, such as VirtualBox, or a bare-metal server. The first solution is suggested for testing scopes, whereas the second solution is suggested for production environment.

Suggested minimum hardware requirements for installation in testing environment:

- Dual-core CPU
- 2+ GB RAM
- 1 gigabit network port
- HDD 80 GB with dynamic disk expansion

Suggested minimum hardware requirements for installation in production environment:

- Quad-core CPU
- 4+ GB RAM
- 1 gigabit network port
- HDD 128+ GB

Once the Master server is installed, all other servers can be powered on, and the user can login into the ITBox UI using the default address <http://10.20.0.2:8000/>, or he can start using the command line interface [7]. The cluster's servers will be booted in bootstrap mode (CentOS based Linux in memory) via PXE. Thus, these servers will be seen by the system as “discovered”, and user will see notifications in the user interface. At this point the user can create an environment, add servers into it and start with the configuration.

##### 6.1.1.1 How to update ITBox to the latest version

If you would update your ITBox node to the latest version, you can install the new version and migrate the production nailgun database instance.

The main steps are:

- `pg_dump -c -h 127.0.0.1 -U nailgun nailgun > outfile.sql` with password nailgun (on the old ITBox installation)
- `psql -d nailgun -U nailgun < outfile.sql` with password nailgun (on the new ITBox installation)

Note: it is possible that the system may show the following error message: “FATAL: Ident authentication failed for user “. To fix this error open `/var/lib/pgsql/data/pg_hba.conf` file and in local section change as follows:

```
local all all md5
```

Finally, restart the database (/etc/init.d/postgresql restart)

You can find the PostgreSQL's official documentation at the url <http://www.postgresql.org/docs/8.4/static/backup-dump.html> [8]

### 6.1.2 User Manual

When the user has completed the master node installation, he can access ITBox UI, visiting the default url <http://10.20.0.2:8000/> (Fig. 3).

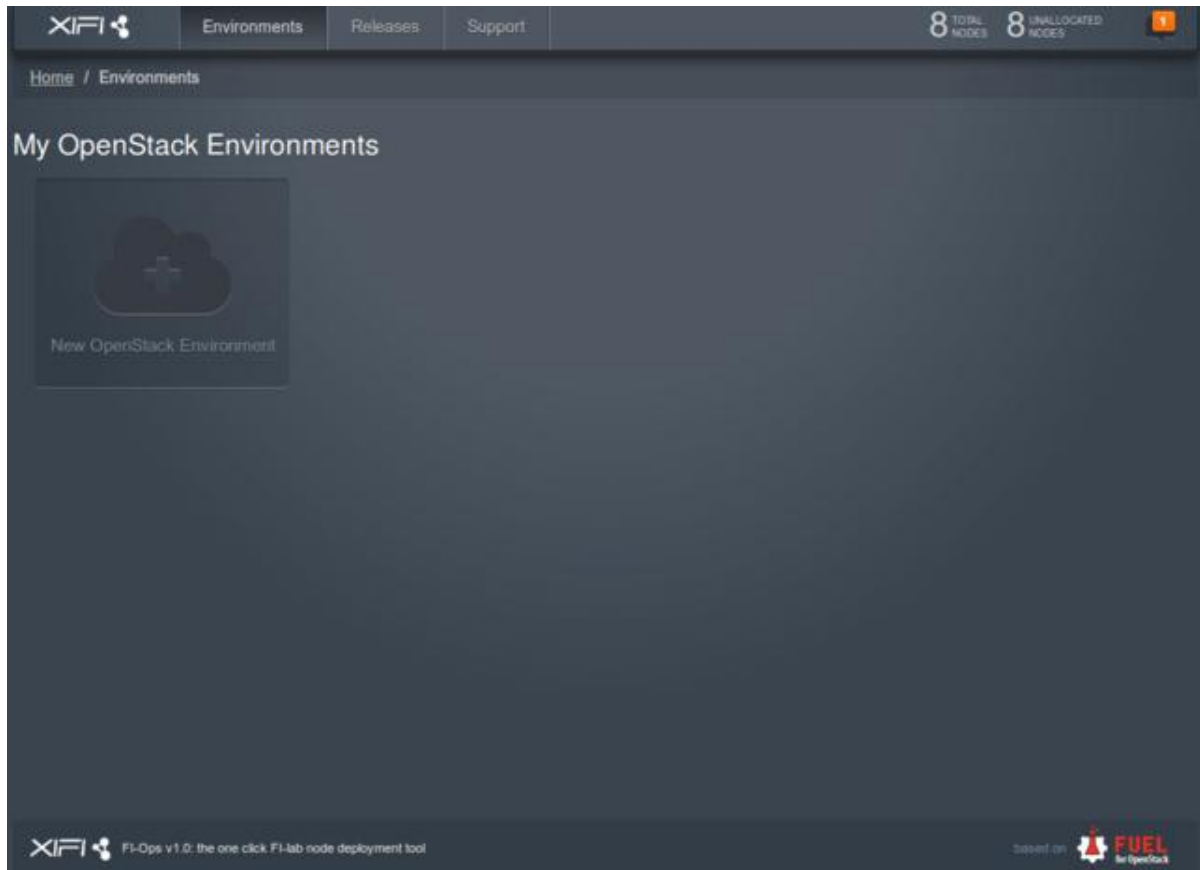
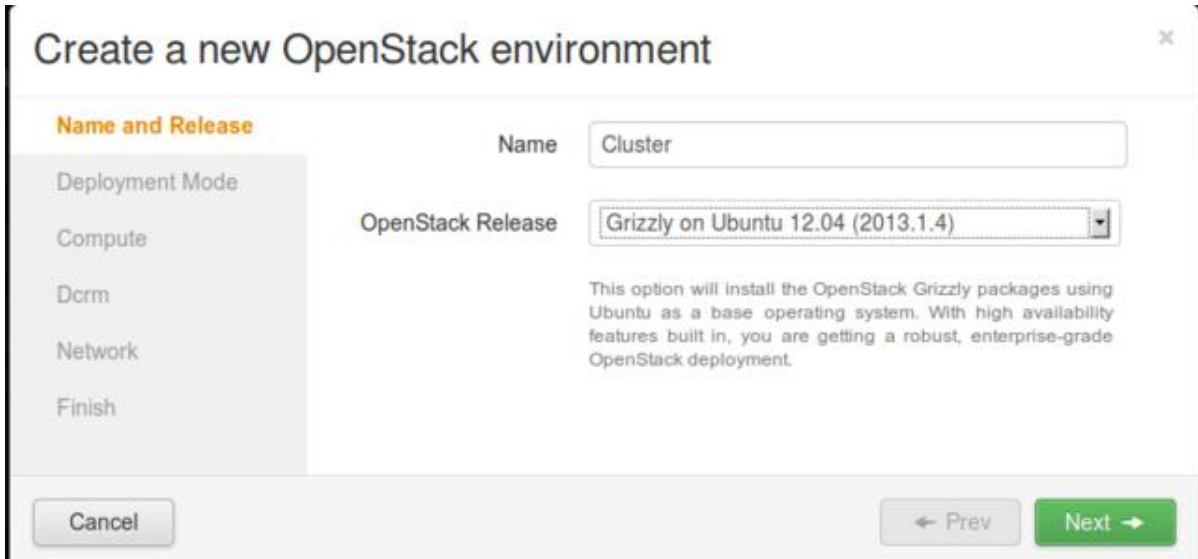


Figure 33 - the ITBox homepage

The user sets bare-metal servers to boot from network via PXE and power them on. They will start automatically with a bootstrap operating system, based on Centos. The ITBox will notify discovered nodes on ITBox UI (see Fig. 3 in the upper right corner). At this moment, the user could create a new environment.



**Create a new OpenStack environment**

**Name and Release**

Name:

OpenStack Release:

This option will install the OpenStack Grizzly packages using Ubuntu as a base operating system. With high availability features built in, you are getting a robust, enterprise-grade OpenStack deployment.

Deployment Mode: ☐ Compute ☐ Dcrm ☐ Network ☐ Finish

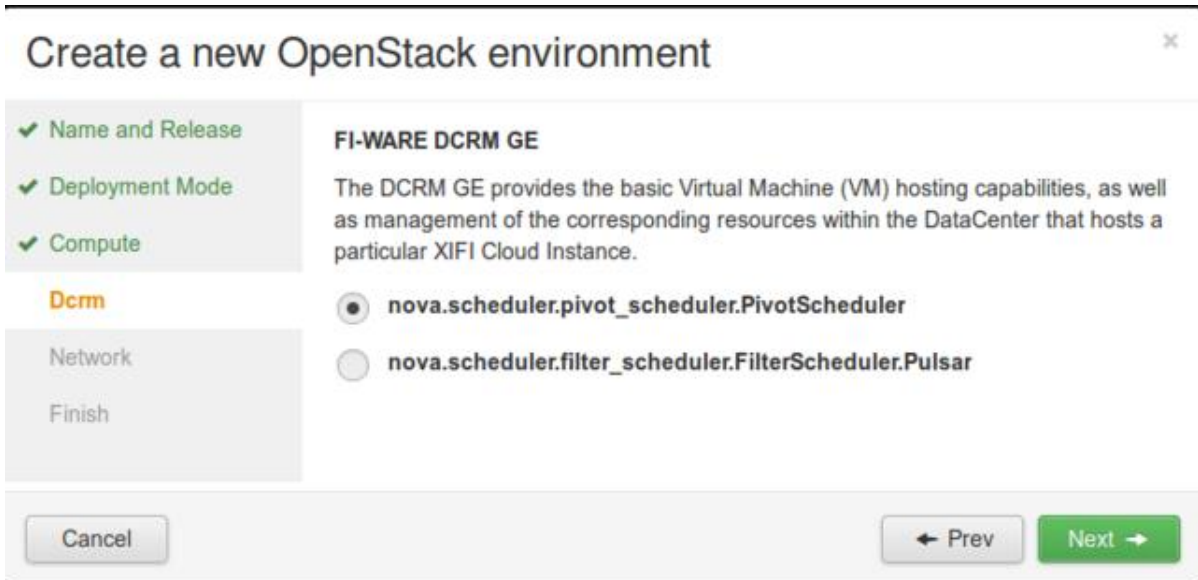
Cancel Prev Next

Figure 34 - creation of a new environment

The first step that involves the user is the “New OpenStack Environment” creation (Fig. 4), where the user inserts such basic information about the environment as name, operating system, deployment mode (multi-node or multi-node with High Availability), hypervisor, DCRM GE with Pivot or Pulsar scheduler and network manager (Nova-Network, Neutron with GRE, Neutron with VLAN).

The DCRM GE installation (Fig. 5) is a XIFI specific feature. If the user selects a DCRM GE scheduler, the ITBox will install all necessary packages and configure Pivot or Pulsar scheduler.

If the user skips this step, then the ITBox will install the FilterScheduler as default.



**Create a new OpenStack environment**

✓ Name and Release  
✓ Deployment Mode  
✓ Compute

**FI-WARE DCRM GE**

The DCRM GE provides the basic Virtual Machine (VM) hosting capabilities, as well as management of the corresponding resources within the DataCenter that hosts a particular XIFI Cloud Instance.

☒ nova.scheduler.pivot\_scheduler.PivotScheduler

☐ nova.scheduler.filter\_scheduler.FilterScheduler.Pulsar

Network  
Finish

Cancel Prev Next

Figure 35 - DCRM install options

Now the environment is ready for deployment.

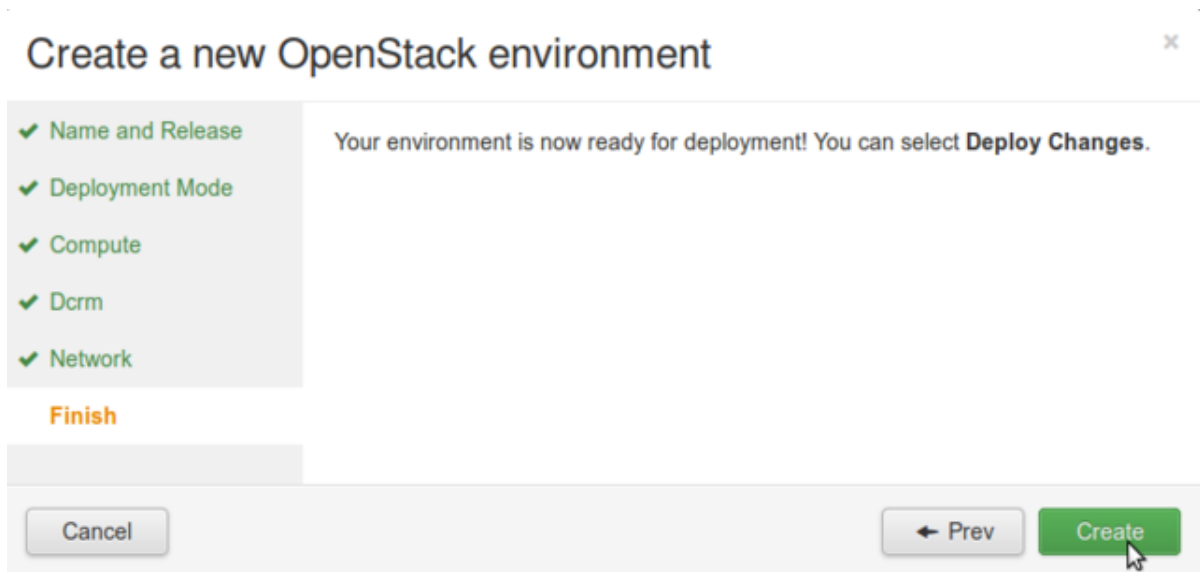


Figure 36 - Final creation step

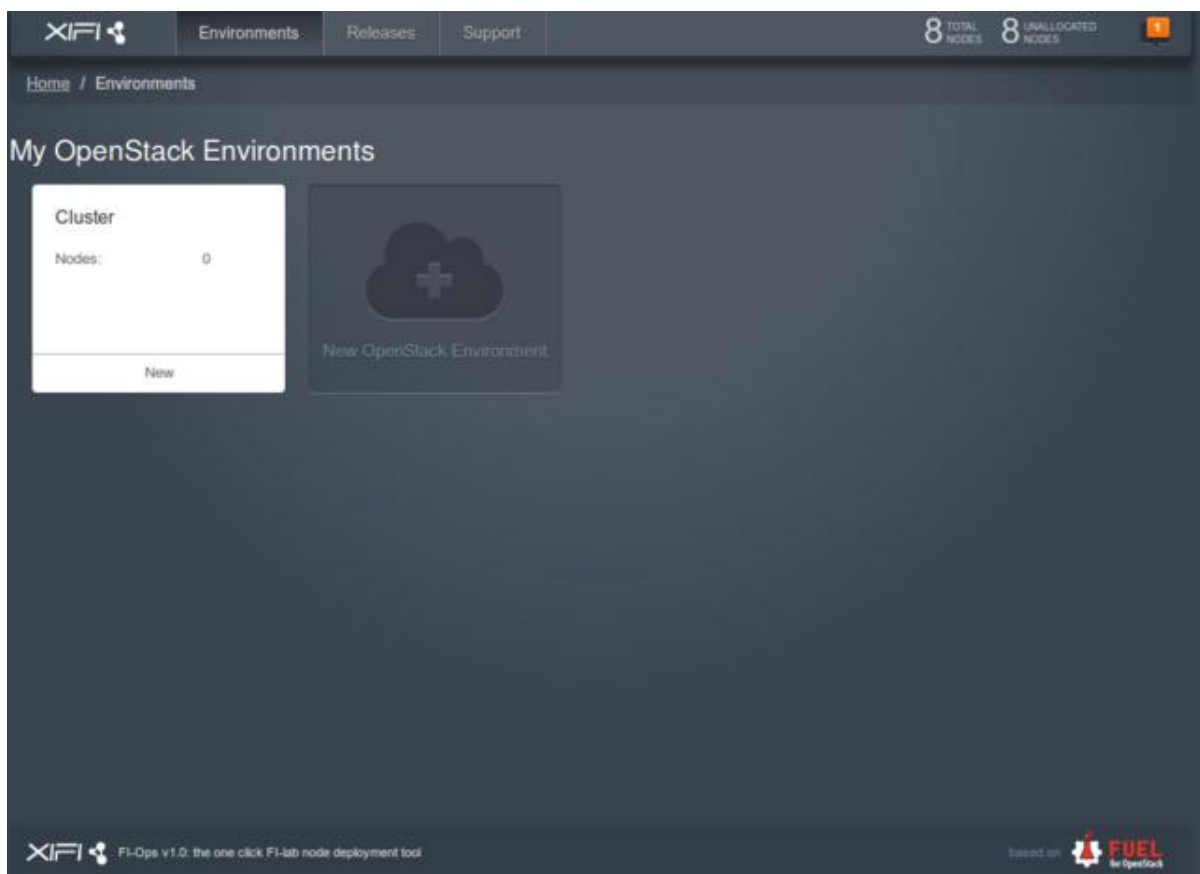


Figure 37 - the page of the created environment

In environment creation process the user should define the architecture of his cloud infrastructure. The user assigns the role to every server, configures the network and defines the space allocated to hard disks and settings other OpenStack options (Fig.21).

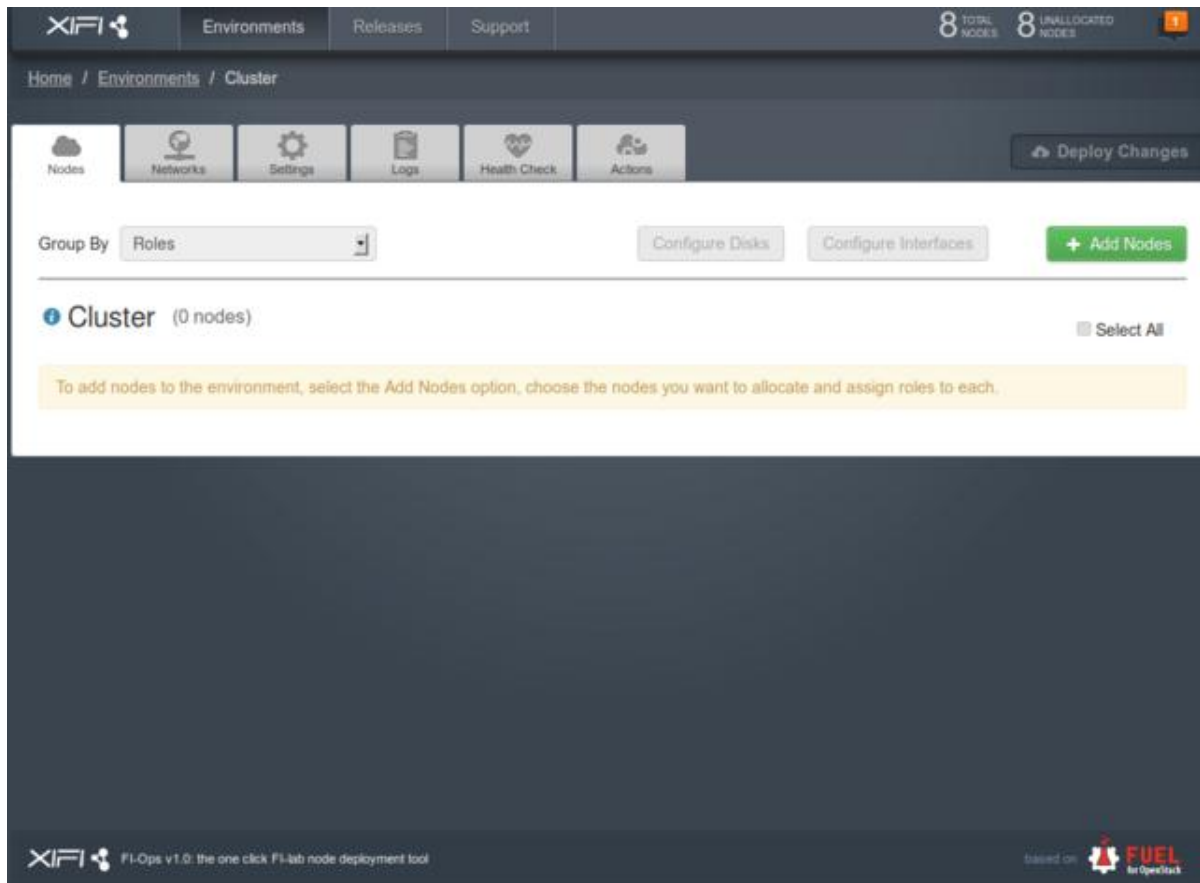


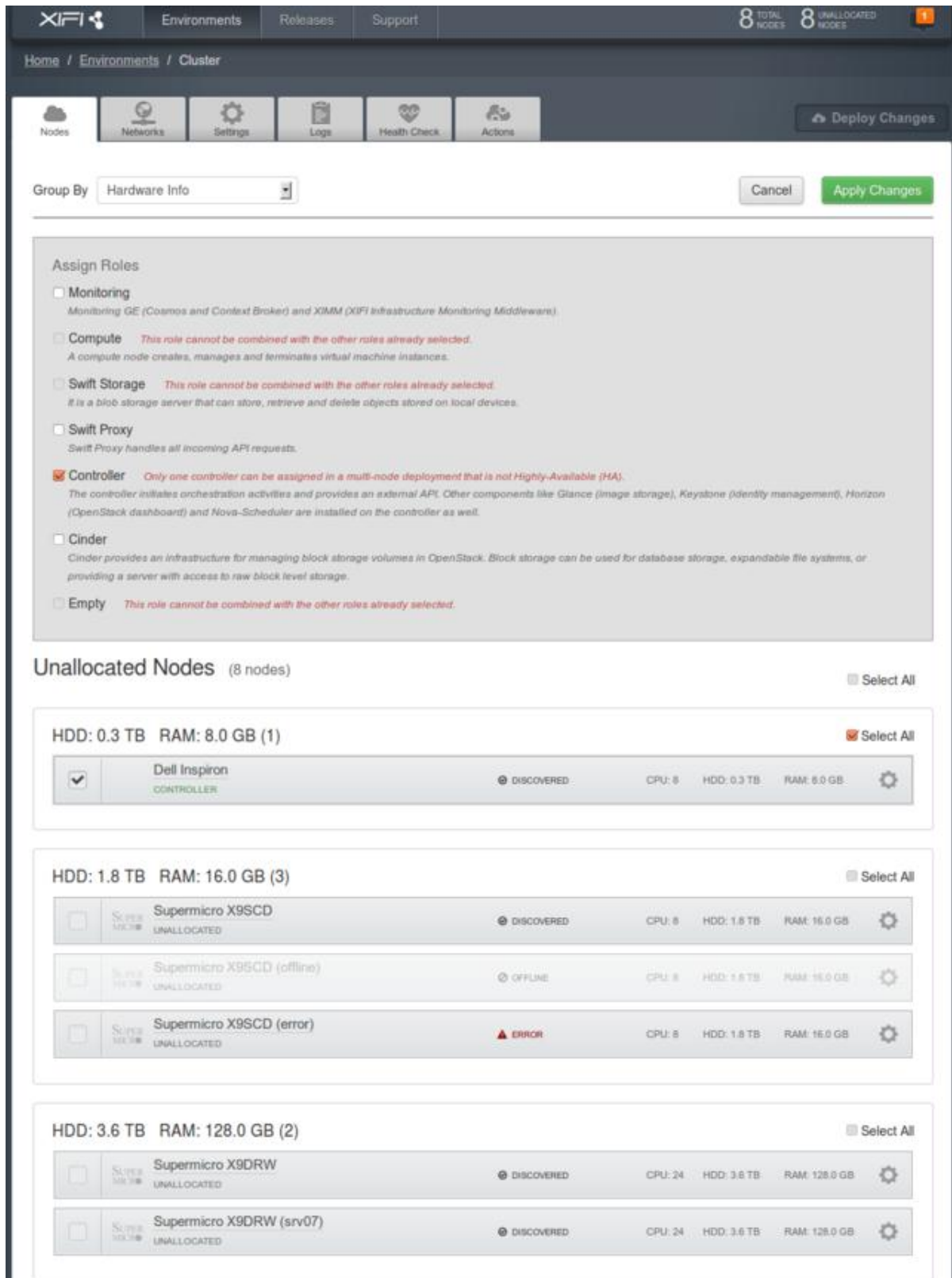
Figure 38 - environment definition

### Giving roles to servers

In “Nodes” tab, the user can view the state of his environment, where the nodes are ordered by Roles. Thus, the user can view the node's details and configure them appropriately.

By clicking on “Add Nodes” button, the ITBox shows users the list of available roles and the list of unallocated nodes. After selecting a role, other incompatible roles are automatically disabled. For example, a controller node cannot be together with a compute node simultaneously, and so on.

Finally the user applies changes (Fig. 22).



**Assign Roles**

- ☐ **Monitoring**  
Monitoring GE (Cosmos and Context Broker) and XIMM (XIFI Infrastructure Monitoring Middleware).
- ☐ **Compute** *This role cannot be combined with the other roles already selected.*  
A compute node creates, manages and terminates virtual machine instances.
- ☐ **Swift Storage** *This role cannot be combined with the other roles already selected.*  
It is a blob storage server that can store, retrieve and delete objects stored on local devices.
- ☐ **Swift Proxy**  
Swift Proxy handles all incoming API requests.
- ☒ **Controller** *Only one controller can be assigned in a multi-node deployment that is not Highly-Available (HA).*  
The controller initiates orchestration activities and provides an external API. Other components like Glance (image storage), Keystone (identity management), Horizon (OpenStack dashboard) and Nova-Scheduler are installed on the controller as well.
- ☐ **Cinder**  
Cinder provides an infrastructure for managing block storage volumes in OpenStack. Block storage can be used for database storage, expandable file systems, or providing a server with access to raw block level storage.
- ☐ **Empty** *This role cannot be combined with the other roles already selected.*

**Unallocated Nodes** (8 nodes) Select All

**HDD: 0.3 TB RAM: 8.0 GB (1)** Select All

Node	Status	CPU	HDD	RAM	Actions
<input checked="" type="checkbox"/> Dell Inspiron	CONTROLLER	DISCOVERED	0.3 TB	8.0 GB	

**HDD: 1.8 TB RAM: 16.0 GB (3)** Select All

Node	Status	CPU	HDD	RAM	Actions
<input type="checkbox"/> Supermicro X9SCD	UNALLOCATED	DISCOVERED	1.8 TB	16.0 GB	
<input type="checkbox"/> Supermicro X9SCD (offline)	UNALLOCATED	OFFLINE	1.8 TB	16.0 GB	
<input type="checkbox"/> Supermicro X9SCD (error)	UNALLOCATED	ERROR	1.8 TB	16.0 GB	

**HDD: 3.6 TB RAM: 128.0 GB (2)** Select All

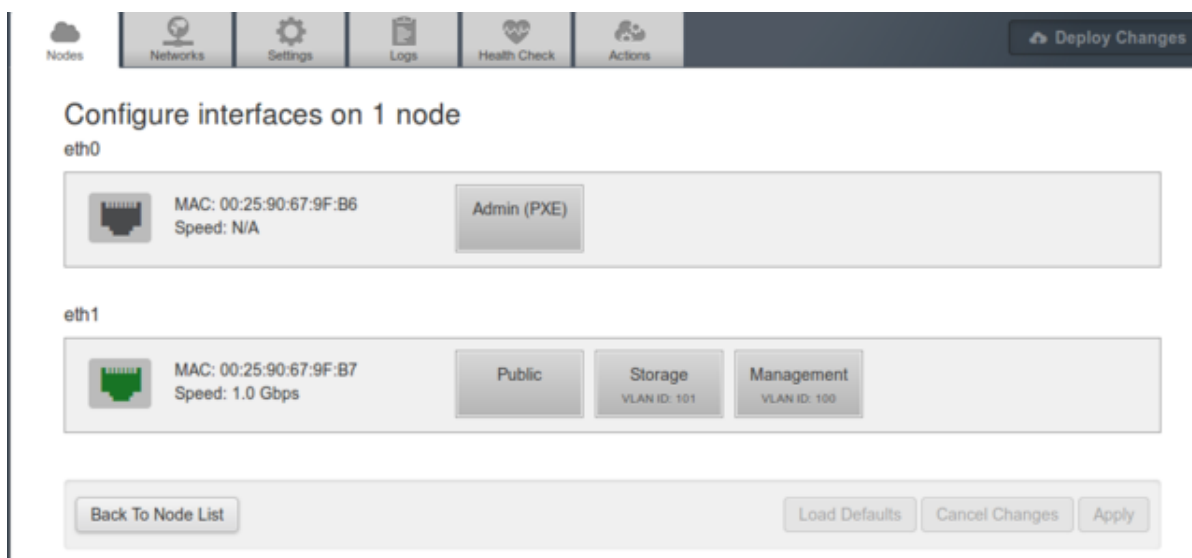
Node	Status	CPU	HDD	RAM	Actions
<input type="checkbox"/> Supermicro X9DRW	UNALLOCATED	DISCOVERED	3.6 TB	128.0 GB	
<input type="checkbox"/> Supermicro X9DRW (srv07)	UNALLOCATED	DISCOVERED	3.6 TB	128.0 GB	

Figure 39 - list of available servers

When the changes are applied, it is possible to tune the node, by clicking on the right button indicated by the gear icon. The ITBox shows a dialog where the user can configure network interfaces, defines the space allocated to hard disks and views server information (e.g. Service tag, Mac addresses,



hardware specifications, etc.) (Fig.22, 23, 24).



Nodes | Networks | Settings | Logs | Health Check | Actions | Deploy Changes

### Configure interfaces on 1 node

**eth0**

MAC: 00:25:90:67:9F:B6  
Speed: N/A

Admin (PXE)

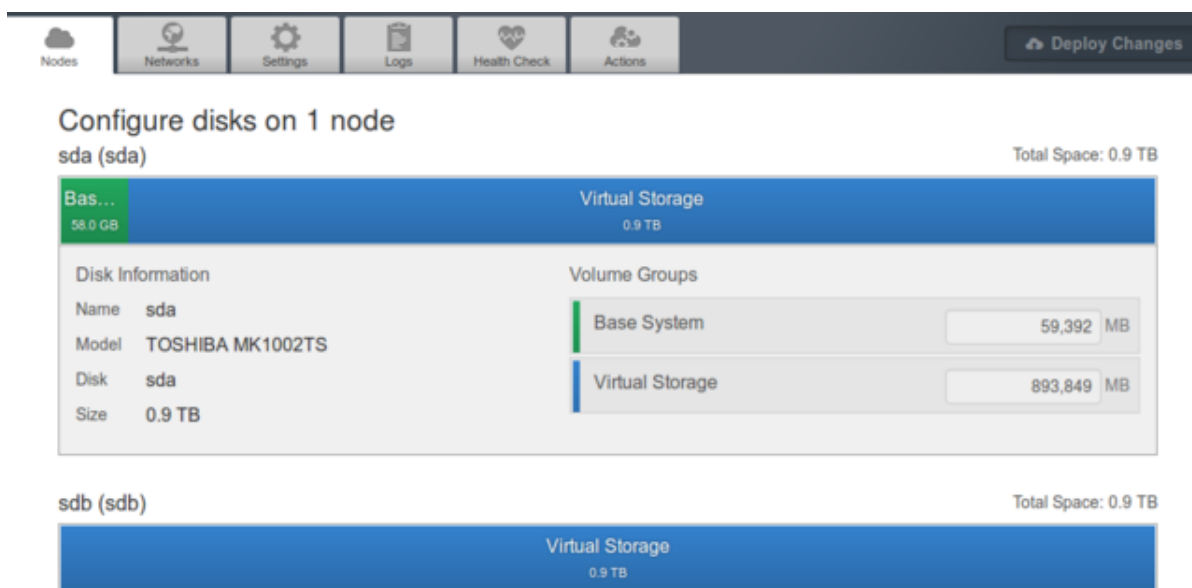
**eth1**

MAC: 00:25:90:67:9F:B7  
Speed: 1.0 Gbps

Public | Storage (VLAN ID: 101) | Management (VLAN ID: 100)

Back To Node List | Load Defaults | Cancel Changes | Apply

Figure 40 - network interface configurations



Nodes | Networks | Settings | Logs | Health Check | Actions | Deploy Changes

### Configure disks on 1 node

**sda (sda)** Total Space: 0.9 TB

Bas... 58.0 GB | Virtual Storage 0.9 TB


Disk Information		Volume Groups	
Name	sda	Base System	59,392 MB
Model	TOSHIBA MK1002TS	Virtual Storage	893,849 MB
Disk	sda		
Size	0.9 TB		

**sdb (sdb)** Total Space: 0.9 TB

Virtual Storage 0.9 TB

Figure 41 - hard disk configuration

## Supermicro X9SCD ✕



**Manufacturer:** Supermicro  
**MAC Address:** 00:25:90:67:9F:B7  
**FQDN:** mc0n1-srt.srt.mirantis.net

<b>System</b>	Supermicro X9SCD	+
<b>CPU</b>	8 x 3.20 GHz	+
<b>Memory</b>	4 x 4.0 GB, 16.0 GB total	+
<b>Disks</b>	2 drives, 1.8 TB total	+
<b>Interfaces</b>	1 x 1.0 Gbps, 1 x N/A	+

Configure Interfaces

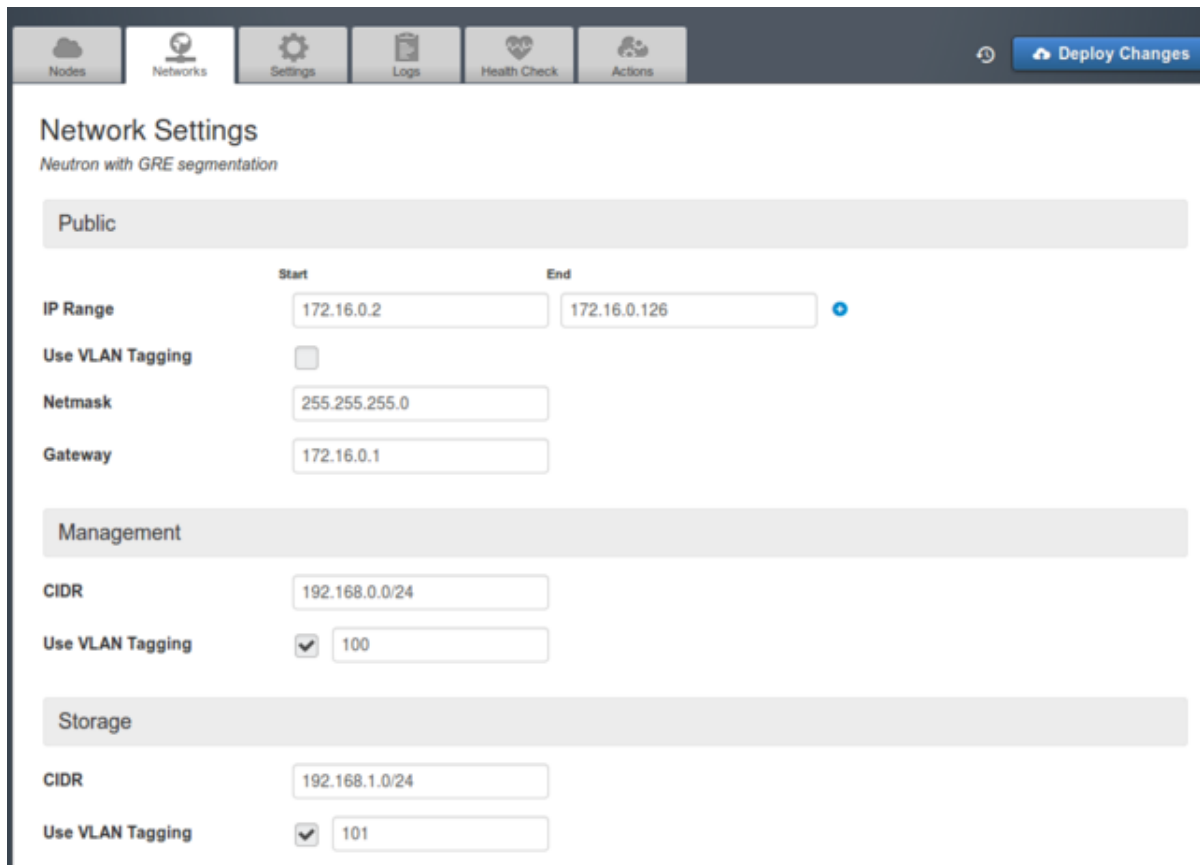
Configure Disks

Close

Figure 42 - detailed information about the selected server

### Network settings

In the Network section, the user can manage configuration parameters. Based on the OpenStack network architecture, ITBox considers three networks: Public, Management and Storage. Management and Storage sections indicate the network subnet in CIDR notation and VLAN tags, whereas the Public section allows to set the IPs pool and its VLAN tag (Fig. 26).



**Network Settings**  
*Neutron with GRE segmentation*

**Public**

Start End

IP Range 172.16.0.2 172.16.0.126

Use VLAN Tagging ☐

Netmask 255.255.255.0

Gateway 172.16.0.1

**Management**

CIDR 192.168.0.0/24

Use VLAN Tagging ☒ 100

**Storage**

CIDR 192.168.1.0/24

Use VLAN Tagging ☒ 101

Figure 43 - infrastructure network settings

The ITBox gives user the opportunity to manage the Neutron plugin and to define the L2 connection tunnel ID range and the L3 floating IP range. Furthermore, the user can verify the network configuration by clicking the “Verify Network” button, which checks for connectivity between nodes using the configured VLANs. It also checks if if some external DHCP interferes with the current deployment (Fig. 27).

### Neutron L2 Configuration

Tunnel ID range

Start

2

End

65535

Base MAC address

fa:16:3e:00:00:00

### Neutron L3 Configuration

External network

Floating IP range

Start

172.16.0.130

End

172.16.0.254

Internal network

CIDR

192.168.111.0/24


Gateway

192.168.111.1

Name servers

8.8.4.4

8.8.8.8



**Network Verification is done in 4 steps:**

1. Every node starts listening for test frames
2. Every node sends out 802.1Q tagged UDP frames
3. Nodes listeners register test frames from other nodes
4. Send DHCP discover messages on all available ports.

Verify Networks

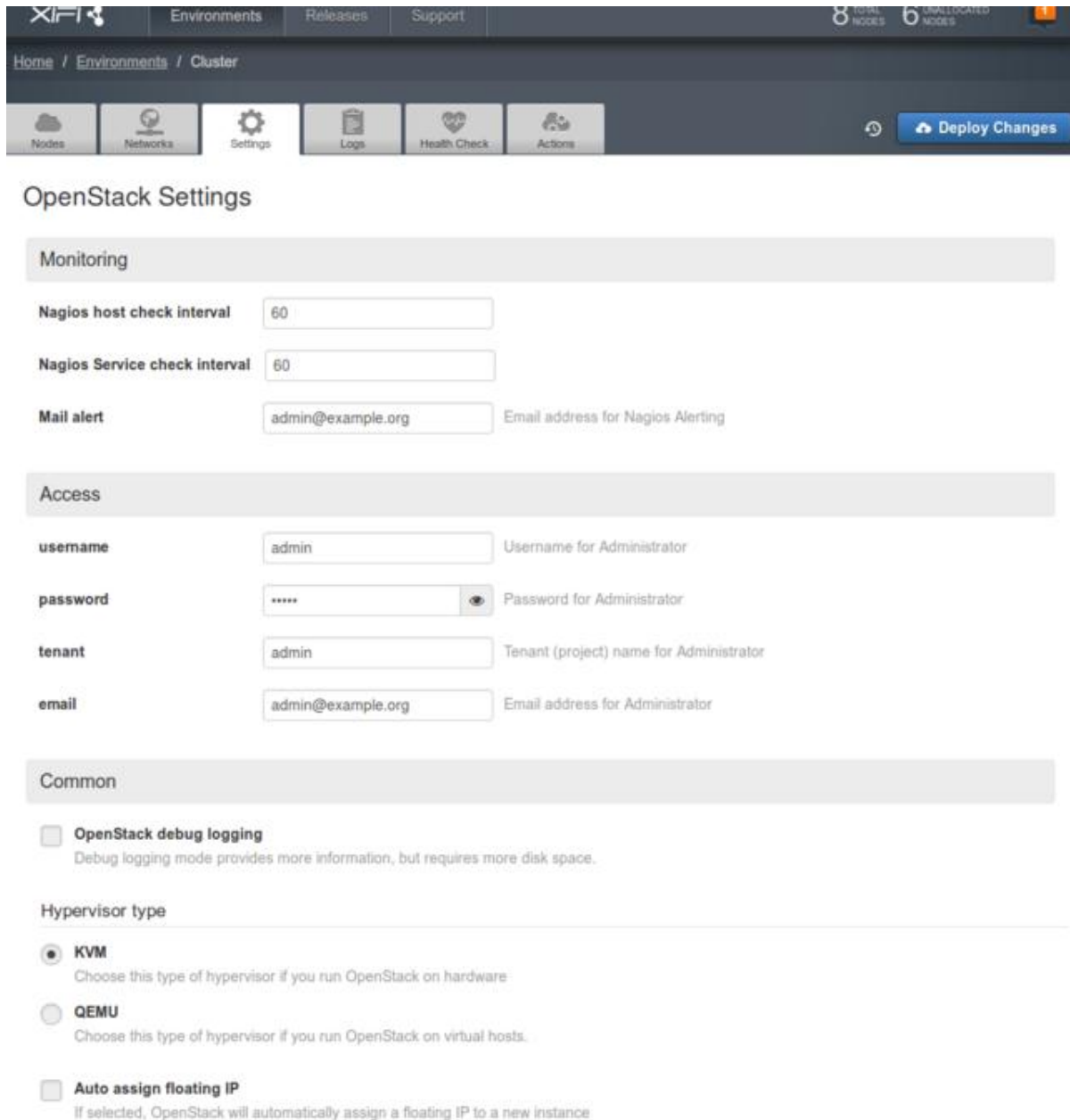
Cancel Changes

Save Settings

Figure 44 - L2/L3 Neutron Configuration

## General Settings

The "Settings" tab contains options useful to manage the current environment. For example, the user can change the OpenStack admin account or can change the hypervisor type or the scheduler driver. To make variations permanently it is necessary re-deploy the changes. (Fig. 26, 27).



The screenshot shows the XIFI web interface for managing infrastructure settings. The top navigation bar includes links for Environments, Releases, and Support, along with status indicators for 8 total nodes and 6 unallocated nodes. The breadcrumb trail is Home / Environments / Cluster. Below the navigation bar is a toolbar with icons for Nodes, Networks, Settings (active), Logs, Health Check, and Actions, followed by a 'Deploy Changes' button.

### OpenStack Settings

**Monitoring**

Nagios host check interval:

Nagios Service check interval:

Mail alert:  Email address for Nagios Alerting

**Access**

username:  Username for Administrator

password:  Password for Administrator

tenant:  Tenant (project) name for Administrator

email:  Email address for Administrator

**Common**

☐ **OpenStack debug logging**  
Debug logging mode provides more information, but requires more disk space.

**Hypervisor type**

☒ **KVM**  
Choose this type of hypervisor if you run OpenStack on hardware

☐ **QEMU**  
Choose this type of hypervisor if you run OpenStack on virtual hosts.

☐ **Auto assign floating IP**  
If selected, OpenStack will automatically assign a floating IP to a new instance

Figure 45 - Infrastructure settings (monitoring, admin account, common)

**Scheduler driver**

☐ **Filter scheduler**  
Currently the most advanced OpenStack scheduler. See the OpenStack documentation for details.

☐ **Simple scheduler**  
This is 'naive' scheduler which tries to find the least loaded host

☒ **Pivot scheduler**  
PIVOT is an advanced placement manager for clouds capable of deploying, optimizing and relocating virtual machines.

☐ **Pulsar scheduler**  
ResourceManager Advanced Capacity Manager.

**Public Key**  Public key(s) to include in authorized\_keys on deployed nodes

**Syslog**

**Hostname**  Remote syslog hostname

**Port**  Remote syslog port

**Syslog transport protocol**

☒ **UDP**

☐ **TCP**

**Storage**

**Object replication factor**  Defines the number of object replicas. At least that many Swift Storage nodes must be deployed.

Figure 46 - infrastructure settings (scheduler drivers, syslog, storage)

## Logs

The log section is designed to monitor the state of installation and support the troubleshooting. The user can select the node to monitoring, the log level and the generator source.

## Health Check

It is very useful, running a post deployment test, to see if the installation process is correctly finished. The Health check process runs a set of tests, and when it is done, the user will see green Thumbs Up sign if it was correct and a red Thumbs Down sign if something went wrong (Fig. 30).

## OpenStack Health Check

☐ Select All





<input type="checkbox"/> Functional tests. Duration 3 min - 14 min	Expected Duration	Actual Duration	Status
Create instance flavor	30 s.	0.1 s.	
Create instance volume <b>Timed out waiting to become available Please refer to OpenStack logs for more details.</b>	200 s.	162.5 s.	
Target component: Compute Scenario: 1. Create a new small-size volume. 2. <u>Wait for volume status to become "available".</u> 3. Check volume has correct name. 4. Create new instance. 5. Wait for "Active" status 6. Attach volume to an instance. 7. Check volume status is "in use". 8. Get information on the created volume by its id. 9. Detach volume from the instance. 10. Check volume has "available" status. 11. Delete volume.			
Keypair creation	25 s.	—	
Security group creation	25 s.	—	

Figure 47 - health check result

**Start deploy** When the user has finished setting the environment, he can start the deployment process, clicking on "Deploy changes" button (Fig. 31).

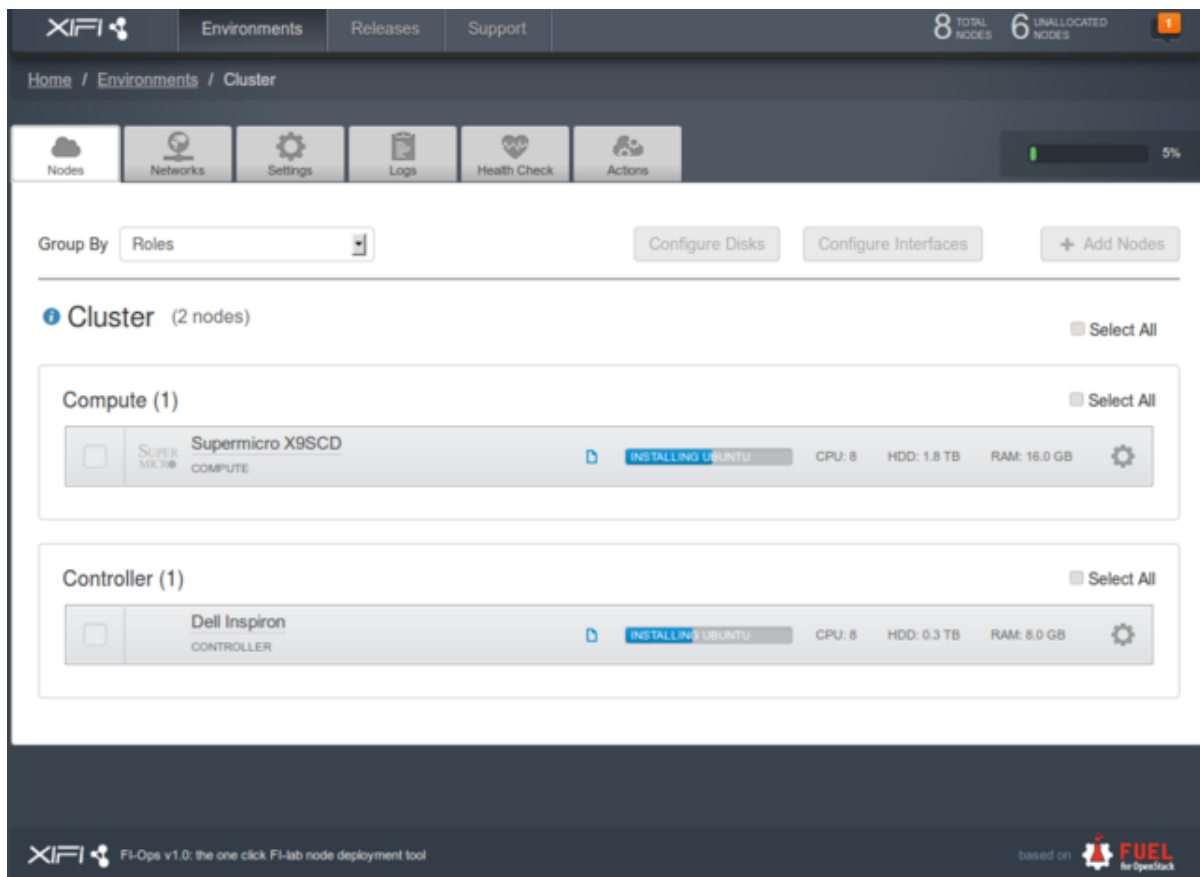


Figure 48 - installation in progress

Fig. 18: installation in progress

## 6.2 DCA

The Deployment and Configuration Adapter (DCA) is the XIFI component that caters for the enhanced deployment functionality, as needed by the project users forming in parallel a Deployment Registry. Briefly the DCA provides:

**Deployment of multiple GEs and XiFi components upon XiFi infrastructure** The DCA supports the deployment and configuration of multiple GEs in a batch mode (as images, through recipes or in a combination), allowing the user to select details (including the sequential or parallel deployment and the notification capabilities). Such multi-GE deployment can take place in a single node or upon federated XiFi nodes. The DCA can also be used to deploy XiFi components upon the infrastructure.

**Check of Available Resources prior to the Deployment** The DCA performs check on the resources that are available to the user, prior to the deployment of one or multiple GEs and according to the documented hardware requirements of the GEs. This functionality can protect the user from receiving errors (by the platform) after invoking the deployment procedure. The resource availability check is performed considering the user's quota upon the XiFi infrastructure, the resources that have been already reserved by the user and the hardware needs of the GEs under deployment (currently quantified in CPU cores, memory and storage). The checks can be performed per node and / or federated nodes.

**Persistency of information related to the deployed GE instances** The DCA holds all pertinent information from the whole lifecycle of GE deployment. This includes the requests on behalf of the users (through the portal) and the system responses as related to the GE instances (going well beyond the typical awareness of the VM instances). This information is adapted and then exposed upon request to the components of WP4, through a set of meaningful queries.

### 6.2.1 Installation Manual

This section provides a step-by-step guide for the installation and configuration of the required software components in order to setup the DCA component in a particular node in XIFI federation.

In particular, guided manuals are foreseen for:

- installation and configuration of Apache Tomcat,
- installation and configuration of MySQL Server,
- installation of the DCA binaries.

Further, configuration instructions are given for all the aforementioned software components.

#### Prerequisites

For the node that will accommodate the DCA (XiFi federation platform):

- CentOS (6.2 and above) is already installed
- Oracle Java Runtime Environment (1.7.0 and above) is already installed
- Apache Tomcat (7.0.35 and above)

The cloud-hosting Generic Enablers (GEs) that support the deployment and configuration of the GEs. Namely the PaaS Manager and the SDC.

A persistency server (DCA currently tested with MySQL DB Server)

For the Infrastructure Node:

The DCRM GE or OpenStack Grizzly



Installation steps:

First of all, verify that the right version of Oracle Java has been installed:

```
[root@dca ~]# java -version
java version "1.7.0_45" Java(TM) SE Runtime Environment (build 1.7.0_45-b18) Java HotSpot(TM) 64-Bit Server VM (build 24.45-b08, mixed mode)
```

Then, install Apache Tomcat as follows:

```
[root@dca ~]# cd /opt [root@dca opt]# wget
http://archive.apache.org/dist/tomcat/tomcat-7/v7.0.41/bin/apache-tomcat-7.0.41.tar.gz
[root@dca opt]# tar xzf apache-tomcat-7.0.41.tar.gz
[root@dca opt]# ./apache-tomcat-7.0.41/bin/startup.sh
```

Tomcat should be up and running. Optionally, you may want to redirect the traffic of port 8080 to port 80 using the following iptables rule:

```
[root@dca opt]# iptables -t nat -A PREROUTING -p tcp --dport 80 -j REDIRECT
--to-port 8080 [root@dca opt]# iptables -I INPUT -p tcp --dport 8080 -j
ACCEPT [root@dca opt]# service iptables save [root@dca opt]# service
iptables restart
```

Upon successfully installing Apache Tomcat, MySQL server and the respective Java JDBC connector should be also installed by issuing the following commands:

```
[root@dca opt]# yum install mysql-server mysql-connector-java [root@dca
opt]# chkconfig --levels 235 mysqld on [root@dca opt]# service mysqld start
```

Next, place the database creation script (dca.sql) in the root directory (/root) and issue the following commands:

```
[root@dca opt]# cd ~ [root@dca ~]# mysql -uroot -p < dca.sql
```

Next, the DCA binaries (dca.war) should be uploaded to the webapps directory of the Apache Tomcat installation (in the context of the present installation guide, this should be /opt/apache-tomcat-7.0.41/webapps). Apache Tomcat should then automatically deploy the war file and extract its contents under the webapps directory. In order to enable transactions, one should initialize a keystone instance into the DCA platform. Supposing that the DCRM instance of the datacenter operator advertises its identity service at the URL <http://hostname.example.com/keystone/v2.0> and is located in a region identified as A\_Region, then, the following command should be issued:

```
curl http://<dca-server-ip>/dca/addEndpoint -X POST -H "Content-Type:
application/json" \
-d '{region:"A_Region",
url:"http://hostname.example.com/keystone/v2.0"}'
```

The answer of the DCA server is the following:

```
{url:"http://hostname.example.com/keystone/v2.0", region:"A_Region"}
```

## 6.2.2 User Manual

The user manual describes the requests and responses of the methods currently offered by the DCA (version 1.0). The interaction has been performed using two trial accounts in the FI-WARE infrastructure (one in lab.fi-ware.eu and one in testbed.fi-ware.eu, regions RegionOne and RegionTwo, respectively), as well as a local (deployed in SYNELIXIS) OpenStack environment (RegionThree). This means that many of the (proprietary) fields are currently blank (appearing as having null values).

### List

### flavors

We request the flavors available in RegionThree. **Request**

```
curl http://localhost:8080/dca/flavors?region=RegionThree
```





```
"checksum": "4a5fa01c81cc300f4729136e28ebe600",
"owner": "e9312e85dde04636a63c1b340f89242a", "is_public": true,
"deleted_at": null, "properties": { },
"size": 358959104 }, { "status": "active",
"name": "Cirros 0.3.1", "deleted": false,
"container_format": "bare", "created_at": "2013-10-23T14:28:21",
"disk_format": "qcow2", "updated_at": "2013-10-23T14:28:22",
"id": "c4c6463f-0acb-4e06-8051-1e14070c154d", "min_disk": 0,
"protected": false, "min_ram": 0,
"checksum": "d972013792949d0d3ba628fbe8685bce",
"owner": "e9312e85dde04636a63c1b340f89242a", "is_public": false,
"deleted_at": null, "properties": { },
"size": 13147648 }, { "status": "active",
"name": "orion", "deleted": false,
"container_format": "bare", "created_at": "2013-11-20T18:06:47",
"disk_format": "qcow2", "updated_at": "2013-11-20T18:12:22",
"id": "791c1279-4d38-4f89-a33b-a3a93a29e75c", "min_disk": 0,
"protected": false, "min_ram": 0,
"checksum": "d60b7cfc2f87a9b69095cbd627805bf0",
"owner": "e9312e85dde04636a63c1b340f89242a", "is_public": false,
"deleted_at": null, "properties": {
"instance_uuid": "a9259820-dd5e-4fb4-9581-09acaddf199b",
"image_location": "snapshot", "image_state": "available",
"instance_type_memory_mb": "2048", "instance_type_swap": "0",
"instance_type_vcpu_weight": "None", "image_type": "snapshot",
"instance_type_id": "5", "ramdisk_id": null,
"instance_type_name": "m1.small",
"instance_type_ephemeral_gb": "0",
"instance_type_rxtx_factor": "1", "kernel_id": null,
"instance_type_flavorid": "2", "instance_type_vcpus": "1",
"user_id": "752627b3561f46b08bc27726ce2630ce",
"instance_type_root_gb": "20", "base_image_ref": "79e9245c-3a02-48af-8dd2-ada0d893331e",
"owner_id": "e9312e85dde04636a63c1b340f89242a" },
"size": 358875136 } ] }
```

## List

## Networks Request

The networks available to the user on RegionThree are requested.

```
curl http://localhost:8080/dca/networks?region=RegionThree
```

## Response

```
{ "networks": [ { "status": "ACTIVE", "subnets": [
"3b873329-6469-4d44-9814-93be7b6d7eb2" ], "name": "net-0",
"id": "a714bca6-7f2d-4181-91fb-44e6b603a7e4", "shared": "false",
"provider: physical_network": null, "admin_state_up": true,
"tenant_id": "e9312e85dde04636a63c1b340f89242a", "provider:
network_type": "gre", "router: external": "false",
"provider: segmentation_id": "1" }, {
"status": "ACTIVE", "subnets": [ "ed1fa327-81ba-40ca-
b523-aa64e45ba091" ], "name": "ext_net",
"id": "c3a72055-71ac-4cc7-afad-4869dc602b19", "shared": "false",
"provider: physical_network": null, "admin_state_up": true,
"tenant_id": "e9312e85dde04636a63c1b340f89242a", "provider:
network_type": "gre", "router: external": "true",
"provider: segmentation_id": "2" } ] }
```

## List

## Servers Request

The already deployed (in total) GEs are requested.

```
curl http://localhost:8080/dca/servers/ge
```

## Response

The response is retrieved. The servers are presented along with the related details.

```
[{"id": "6fda6be8-0e70-4d08-9731-c858d6a27f50",  
  "name": "test-xifi-proton",  
  "imageRef": "90d4865d-5e7b-4d95-af2c-69753e1740d6",  
  "flavorRef": "2",  
  "keyName": "xifi",  
  "securityGroups": "default,cep",  
  "created": 1390051880000,  
  "status": null,  
  "tenantId": "00000000000000000000000000000101",  
  "userId": "artemis-voulkidis",  
  "region": "RegionOne"}], [{"id": "a335265d-777e-42fa-8f5a-355160bc93cc",  
  "name": "test-xifi-proton-r1",  
  "imageRef": "90d4865d-5e7b-4d95-af2c-69753e1740d6",  
  "flavorRef": "2",  
  "keyName": "xifi",  
  "securityGroups": "default,cep",  
  "created": 1390135132000,  
  "status": null,  
  "tenantId": "00000000000000000000000000000101",  
  "userId": "artemis-voulkidis",  
  "region": "RegionOne"}], [{"id": "29clee5e-fac8-451d-85d9-963ed7ae2386",  
  "name": "test-xifi-orion",  
  "imageRef": "02fdb0bc-6b47-4af4-ab13-95508033cdb4",  
  "flavorRef": "2",  
  "keyName": "xifi",  
  "securityGroups": "default",  
  "created": 1390135136000,  
  "status": null,  
  "tenantId": "00000000000000000000000000000158",  
  "userId": "artemis-voulkidis",  
  "region": "RegionTwo"}], [{"id": "1ae2d8d8-e8ed-4e4e-88bd-4209b808eeda",  
  "name": "test-xifi-proton-r2",  
  "imageRef": "7b001833-5aaa-4a4d-84fa-803e3c59377d",  
  "flavorRef": "2",  
  "keyName": "xifi",  
  "securityGroups": "default,cep",  
  "created": 1390052553000,  
  "status": null,  
  "tenantId": "00000000000000000000000000000158",  
  "userId": "artemis-voulkidis",  
  "region": "RegionTwo"}]]
```

More complex queries are also supported by issuing relevant API calls containing http parameters. For example, the CEP instances deployed in RegionOne could be discovered as follows:

## Request

```
curl http://localhost:8080/dca/servers/ge?desc=cep&region=RegionTwo
```

## Response

```
[ { "id": "1ae2d8d8-e8ed-4e4e-88bd-4209b808eeda",  
  "name": "test-xifi-proton-r2", "imageRef": "7b001833-5eaa-4a4d-84fa-803e3c59377d",  
  "flavorRef": "2", "keyName": "xifi",  
  "securityGroups": "default,cep", "created": 1390052553000,  
  "status": null, "tenantId": "00000000000000000000000000000000158",  
  "userId": "artemis-voulkidis", "region": "RegionTwo" } ]
```

The DCA component also support active VM (GE) listing, where the data is requested directly by the DCRM instances of the federation. For example, an infrastructure owner may be interested on checking the GEs that are deployed on its own infrastructure, located at RegionOne:

## Request

```
curl http://localhost:8080/dca/servers/live?region=RegionOne
```

The response of the DCA module could be as follows (the full listing has been suppressed for reasons of brevity):

Response

```
{  "list":[      {          "id":"a335265d-777e-42fa-8f5a-355160bc93cc",
"name":"test-xifi-proton-r1",          "addresses":{
"addresses":{          "private":[          {
"macAddr":null,          "version":"4",
"addr":"10.0.1.202",          "type":null          }
]          }          },          "links":[          {
"rel":"self",          "href":"http://cloud.lab.fi-
ware.eu:8774/v2/00000000000000000000000000000101/servers/a335265d-777e-
42fa-8f5a-355160bc93cc",          "type":null          } ,
{          "rel":"bookmark",
"href":"http://cloud.lab.fi-
ware.eu:8774/00000000000000000000000000000101/servers/a335265d-777e-42fa-
8f5a-355160bc93cc",          "type":null          } ],
"image":{          "id":"90d4865d-5e7b-4d95-af2c-69753e1740d6",
"links":[          {          "rel":"bookmark",
"href":"http://cloud.lab.fi-
ware.eu:8774/00000000000000000000000000000101/images/90d4865d-5e7b-4d95-
af2c-69753e1740d6",          "type":null          }
]          },          "flavor":{          "id":"2",
"links":[          {          "rel":"bookmark",
"href":"http://cloud.lab.fi-
ware.eu:8774/00000000000000000000000000000101/flavors/2",
"type":null          }          ],          "public":null
} ,          "accessIPv4":"","accessIPv6":"","
"configDrive":"","status":"ACTIVE",          "progress":0,
"fault":null,          "tenantId":"00000000000000000000000000000101",
"userId":"artemis-voulkidis",          "keyName":"xifi",
"hostId":"c0d2f2faa797bfb8be05680a6d2f31ba4553f7219e35a13011cb22d1",
"updated":"2014-01-19T12:39:05Z",          "created":"2014-01-
19T12:38:52Z",          "metadata":{          } ,
"powerState":"1",          "vmState":"active",
"host":"gcsic022.ifca.es",          "instanceName":"instance-00000ad2",
"hypervisorHostname":null,          "diskConfig":"MANUAL",
"availabilityZone":null,          "uuid":null,          "adminPass":null
}      ] }
```

Combinatorial requests are also supported. For example, assume that it is of interest to know on which XIFI nodes are instances of CEP and Marketplace GE's simultaneously deployed:

Request

```
curl
http://localhost:8080/dca/servers/ge/multiple?desc1=cep&desc2=marketplace
```

Assuming that in the present state of the XIFI federation, only RegionOne has these two GE's deployed, the answer would have been:

Response

```
[  "RegionOne" ]
```

### Single

## Server

## Deployment Request

In this request we ask for the deployment of a single CEP instance on RegionOne. Necessary information is provided through the POST method. The response is condensed for reasons of brevity.

```
curl http://localhost:8080/dca/servers -X POST -H "Content-Type: application/json" -d '{
    "name": "test-3",
    "imageRef": "90d4865d-5e7b-4d95-af2c-69753e1740d6",
    "flavorRef": "2",
    "keyName": "xifi",
    "securityGroups": [
        {
            "name": "default"
        },
        {
            "name": "cep"
        }
    ],
    "region": "RegionOne"
}'
```

## Response

The response is retrieved and it shows that the server is created.

```
{ "id": "ffae6295-9aba-48a8-908d-4eeeb71bd79c", "name": null,  
"addresses": null, "links": [ { "rel": "self",  
"href": "http://cloud.lab.fi-ware.eu:8774/v2/00000000000000000000000000000101/servers/ffae6295-9aba-48a8-908d-4eeeb71bd79c",  
"type": null } , {  
"rel": "bookmark", "href": "http://cloud.lab.fi-ware.eu:8774/00000000000000000000000000000101/servers/ffae6295-9aba-48a8-908d-4eeeb71bd79c",  
"type": null } ],  
"diskConfig": "MANUAL", "adminPass": "hRmXkm97DRqY" }
```

### Multiple

## Servers

## Deployment Request

In this request we ask for the deployment of a CEP instance, destined for RegionOne, and an Orion CPB instance, destined for RegionTwo.

```
curl http://localhost:8080/dca/multiple -X POST -H "Content-Type: application/json" -d '{  "list": [    {      "name": "test-xifi-proton-r1",      "imageRef": "90d4865d-5e7b-4d95-af2c-69753e1740d6",      "flavorRef": "2",      "keyName": "xifi",      "securityGroups": [        {          "name": "default"        }      ],      "name": "cep"    },    {      "name": "test-xifi-orion-r2",      "imageRef": "02fdb0bc-6b47-4af4-ab13-95508033cdb4",      "flavorRef": "2",      "keyName": "xifi",      "securityGroups": [        {          "name": "default"        }      ],      "region": "RegionTwo"    }  ]  }'
```

## Response

The response is retrieved and proves the creation of the two requested servers (VMs).

```
[ { "id": "a335265d-777e-42fa-8f5a-355160bc93cc",  
  "name": null,  
    "addresses": null,  
      "links": [ {  
        "rel": "self",  
          "href": "http://cloud.lab.fiware.eu:8774/v2/000000000000000000000000000000000101/servers/a335265d-777e-42fa-8f5a-355160bc93cc",  
            "type": null  
          } ,  
            {  
              "rel": "bookmark",  
                "href": "http://cloud.lab.fiware.eu:8774/000000000000000000000000000000000101/servers/a335265d-777e-42fa-8f5a-355160bc93cc",  
                  "type": null  
                } ] ,  
        "diskConfig": "MANUAL",  
          "adminPass": "ZGs4QxAvZc8c"  
        } ,  
          {  
            "id": "29clee5e-fac8-451d-85d9-963ed7ae2386",  
              "name": null,  
                "addresses": null,  
                  "links": [ {  
                    "rel": "self",  
                      "href": "http://130.206.80.11:8774/v2/000000000000000000000000000000000158/servers/29clee5e-fac8-451d-85d9-963ed7ae2386",  
                        "type": null  
                      } ,  
                        {  
                          "rel": "bookmark",  
                            "href": "http://130.206.80.11:8774/000000000000000000000000000000000158/servers/29clee5e-fac8-451d-85d9-963ed7ae2386",  
                              "type": null  
                                } ] ,  
                    "diskConfig": "MANUAL",  
                      "adminPass": "P8inbZ7LErA8"  
                    } ]
```

## Delete

## Server Request

We request the deletion of a specific server.

```
curl -X DELETE http://localhost:8080/dca/servers/d5c22170-38d6-4344-9d62-c9770e550cee
```

## Response

The response is retrieved and show the deletion of the server.

```
{      "deleted": "d5c22170-38d6-4344-9d62-c9770e550cee" }
```

Note that the region of the server is inferred through the DCA persistency layer, described later.

## DCA Persistency Layer

One of the primary goals of DCA is to log the requests made by the users of the federation to the federation, in order to provide meaningful statistics to the infrastructure owners and the federation operators, through a set of predefined queries. Further, DCA could as well operate as a means of *federation cache*, providing quick information on the characteristics of the VM instances deployed in the federation, including Ids and regions of the host XIFI-nodes, the type of the instances etc. The architecture of the DCA persistency layer, currently implemented upon a MySQL server, is depicted in the following figure.



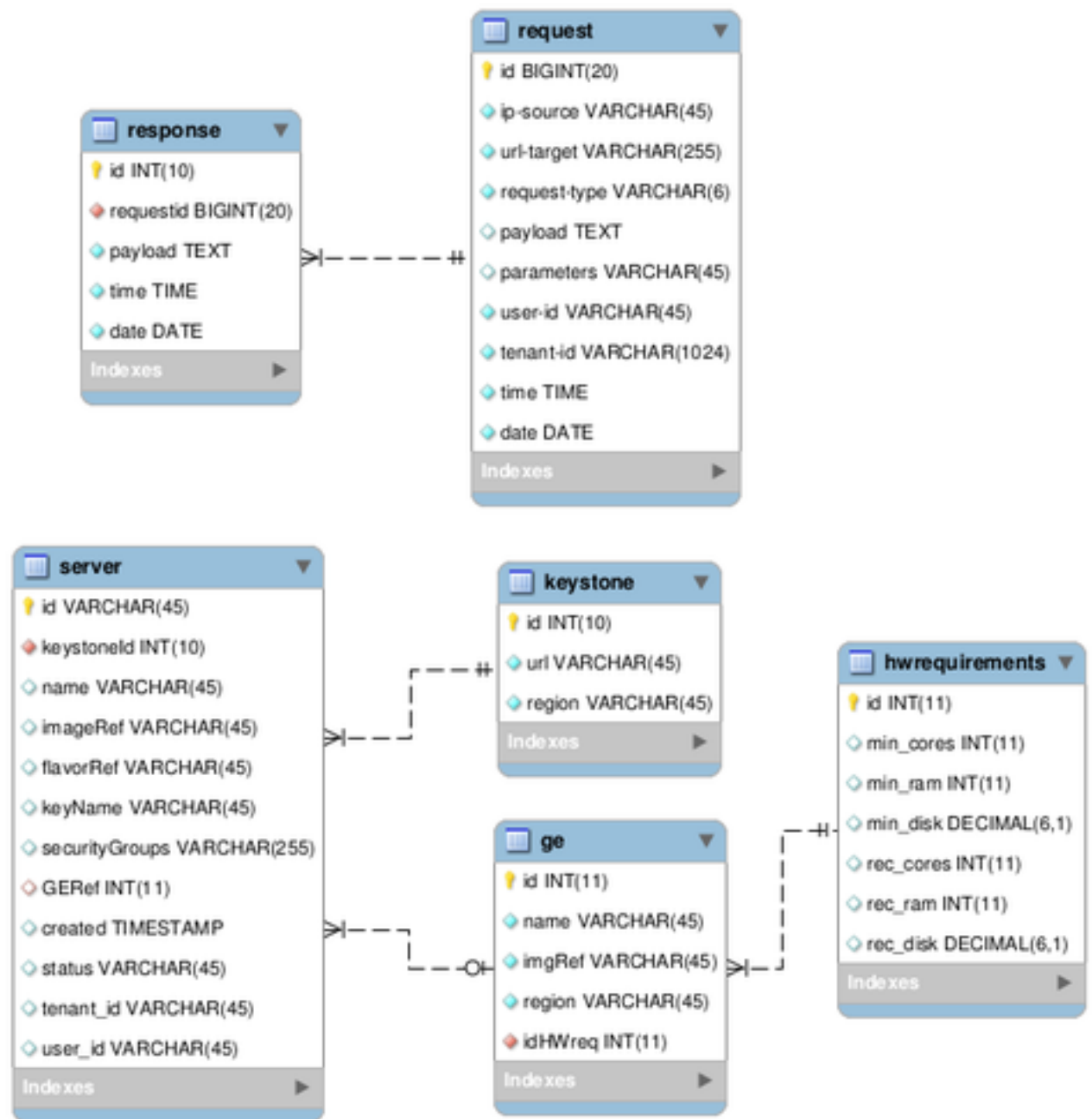


Figure 3: The architecture of the DCA persistency layer

As may be deduced from the above figure, DCA logs the requests made by the federation users and relates them to the federation responses, in an attempt to provide information related to the federation status. This information may be forwarded to the XIFI recommendation tool and the infrastructure owners for further elaboration.

```
mysql> describe dca.request;
```

Field	Type	Null	Key	Default	Extra
id	bigint(20) unsigned	NO	PRI	NULL	
ip-source	varchar(45)	NO			NULL

```

| | url-target | varchar(255) | NO | | NULL
| | request-type | varchar(6) | NO | | NULL
| | payload | text | YES | | NULL
| | parameters | varchar(45) | YES | | NULL
| | user-id | varchar(45) | NO | | NULL
| | tenant-id | varchar(1024) | NO | | NULL
| | time | time | NO | | NULL
| | date | date | NO | | NULL
| +-----+
-----+ mysql> describe dca.response;

```

```

+-----+-----+-----+-----+-----+-----+
+ | Field | Type | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
-- | id | int(10) unsigned | NO | PRI | NULL |
auto_increment | requestid | bigint(20) unsigned | NO | MUL | NULL |
| | payload | text | NO | | NULL |
| | time | time | NO | | NULL |
| | date | date | NO | | NULL |
| +-----+-----+-----+-----+-----+
--

```

Apart from storing the requests made by the users and the related responses, the DCA persistency layer allows for storing information related to the federation nodes DCRM identity services (most likely based on Keystone) in a table called *keystone*.

```
mysql> describe dca.keystone;
```

```

+-----+-----+-----+-----+-----+-----+-----+
Field | Type | Null | Key | Default | Extra | | +-----+
--+-----+-----+-----+-----+-----+-----+ | id |
int(10) unsigned | NO | PRI | NULL | auto_increment | | url |
varchar(45) | NO | UNI | URL | | | region |
varchar(45) | NO | UNI | Region | | +-----+-----+
-----+-----+-----+-----+-----+

```

Also, DCA has built-in information related to the GE images already available through FI-WARE lab and testbed. This information is related to the image reference UUID that can be used to identify the type of the deployed VM (if it constitutes a GEi) when the installation is made directly through image deployment. Further, to ensure proper a priori monitoring of the user quotas and assure minimal/recommended operation of the GEs offered by the FI-WARE environment, DCA has populated another table, called *hwrequirements* that describes the minimum and recommended resources a GE requires for proper operation. This information is coupled to the GE database representation as deduced both by Fig. 3 and the following MySQL output:

```
mysql> describe dca.ge;
```

```

+-----+-----+-----+-----+-----+-----+-----+ | Field
| Type | Null | Key | Default | Extra | | +-----+
--+-----+-----+-----+-----+-----+-----+ | id | int(11) | NO
| PRI | NULL | auto_increment | | name | varchar(45) | NO | |
NULL | | imgRef | varchar(45) | NO | UNI | NULL |
| | region | varchar(45) | NO | | NULL | |
idHWreq | int(11) | NO | MUL | NULL | | +-----+
+-----+-----+-----+-----+-----+ mysql> describe
dca.hwrequirements;

```

```

+-----+-----+-----+-----+-----+-----+-----+ |
Field | Type | Null | Key | Default | Extra | | +-----+
--+-----+-----+-----+-----+-----+-----+ | id
| int(11) | NO | PRI | NULL | auto_increment | | min_cores |
int(11) | YES | | NULL | | | min_ram |

```

int(11)	YES		NULL		min_disk
decimal(6,1)	YES		NULL		rec_cores
int(11)	YES		NULL		rec_ram
int(11)	YES		NULL		rec_disk
decimal(6,1)	YES		NULL		

When a request for a VM deployment in a specific region is issued to the DCA component, the request is logged and the request is forwarded to the controller of the node belonging to the specified region. If the image ID corresponds to a GE image ID, then the respective reference is passed to the database server representation and the VM is onwards considered to be a GEi. DCA also logs the id of the identity service that was contacted to acquire the necessary information to get the VM (GE) deployed, the name of the deployed VM (GE), the flavor that was chosen by the user, the security groups that were identified, as well as other user- and date/time-related information. A representation of the structure of the database representation of a deployed VM (GE) is depicted below.

```
mysql> describe dca.server;
```

Field	Type	Null	Key	Default	Extra	id
varchar(45)	NO	PRI	NULL		keystoneId	
int(10) unsigned	NO	MUL	NULL		name	
varchar(45)	YES		NULL		imageRef	
varchar(45)	YES		NULL		flavorRef	
varchar(45)	YES		NULL		keyName	
varchar(45)	YES		NULL		securityGroups	
varchar(255)	YES		NULL		GERef	
int(11)	YES	MUL	NULL		created	
timestamp	YES		NULL		status	
varchar(45)	YES		NULL		tenant_id	
varchar(45)	YES		NULL		user_id	
varchar(45)	YES		NULL			

## REFERENCES

---

- [1] In Cloud environments based on the high-availability principle, traditional backup services offered by data centres are not more required.
- [2] Transparency, as the capability of inspect a service so as that QoS can be observed and validated is strictly related to the “Measured Service” principle.
- [3] OpenStack Networking Administration Guide, <http://docs.OpenStack.org/grizzly/OpenStack-network/admin/content/connectivity.html>
- [4] OpenStack Operations Guide, [http://docs.OpenStack.org/trunk/OpenStack-ops/content/network\\_design.html](http://docs.OpenStack.org/trunk/OpenStack-ops/content/network_design.html)
- [5] D5.2 Report on XIFI Core Backbone Deployment [https://bscw.fi-xifi.eu/pub/bscw.cgi/d64414/XIFI-D5.2-XIFI\\_Core\\_Backbone.pdf](https://bscw.fi-xifi.eu/pub/bscw.cgi/d64414/XIFI-D5.2-XIFI_Core_Backbone.pdf)
- [6] RFC 1918, <http://tools.ietf.org/html/rfc1918>
- [7] D5.1 PROCEDURES AND PROTOCOLS FOR XIFI FEDERATION [https://bscw.fi-xifi.eu/pub/bscw.cgi/d44719/XIFI-D5.1-Procedures\\_and\\_protocols\\_for\\_XIFI\\_federation.pdf](https://bscw.fi-xifi.eu/pub/bscw.cgi/d44719/XIFI-D5.1-Procedures_and_protocols_for_XIFI_federation.pdf)
- [8] Open source software for building private and public clouds, <http://www.OpenStack.org>
- [9] [9.0](#) [9.1](#) [9.2](#) OpenStack Operations Guide, <http://docs.OpenStack.org/trunk/OpenStack-ops/content/>
- [10] OpenStack Conceptual Architecture, <http://docs.OpenStack.org/grizzly/OpenStack-compute/admin/content/conceptual-architecture.html>
- [11] OpenStack Nova, <https://wiki.OpenStack.org/wiki/Nova>
- [12] OpenStack Neutron, <https://wiki.OpenStack.org/wiki/Neutron>
- [13] Open Virtual Switch, <http://openvswitch.org>
- [14] OpenStack Glance, <https://wiki.OpenStack.org/wiki/Glance>
- [15] [15.0](#) [15.1](#) OpenStack Image Management, [http://docs.OpenStack.org/grizzly/OpenStack-compute/admin/content/ch\\_image\\_mgmt.html](http://docs.OpenStack.org/grizzly/OpenStack-compute/admin/content/ch_image_mgmt.html)
- [16] OpenStack Keystone, <https://wiki.OpenStack.org/wiki/Keystone>
- [17] OpenStack Cinder, <https://wiki.OpenStack.org/wiki/Cinder>
- [18] OpenStack Swift, <https://wiki.OpenStack.org/wiki/Swift>
- [19] OpenStack Horizon, <https://wiki.OpenStack.org/wiki/Horizon>
- [20] Open Network Management Application Platform, <http://www.opennms.org>
- [21] Infrastructure for Network Performance Monitoring, <http://www.perfsonar.net>
- [22] [22.0](#) [22.1](#) [22.2](#) [22.3](#) OpenStack Compute and Image System Requirements, <http://docs.OpenStack.org/grizzly/OpenStack-compute/admin/content/compute-system-requirements.html>
- [23] [23.0](#) [23.1](#) OpenStack Cloud Controller Design, [http://docs.OpenStack.org/grizzly/OpenStack-ops/content/cloud\\_controller\\_design.html](http://docs.OpenStack.org/grizzly/OpenStack-ops/content/cloud_controller_design.html)
- [24] [24.0](#) [24.1](#) OpenStack Instance Storage Solutions, [http://docs.OpenStack.org/grizzly/OpenStack-ops/content/compute\\_nodes.html#instance\\_storage](http://docs.OpenStack.org/grizzly/OpenStack-ops/content/compute_nodes.html#instance_storage)
- [25] [25.0](#) [25.1](#) OpenStack System Requirements, <http://docs.OpenStack.org/grizzly/OpenStack-object-storage/admin/content/object-storage-system-requirements.html>
- [26] Pacemaker, <http://clusterlabs.org/wiki/Pacemaker>

[27] Galera, [http://www.codership.com/wiki/doku.php?id=galera\\_wiki](http://www.codership.com/wiki/doku.php?id=galera_wiki)

OpenStack High Availability Guide, <http://docs.OpenStack.org/high-availability-guide/content/index.html>

[28] OpenStack Block Storage Service Administration Guide, <http://docs.OpenStack.org/grizzly/OpenStack-block-storage/admin/content/index.html>

[29] Refer to The XIFI Consortium Deliverable D4.2 - Baseline Tools v1: [https://bscw.fi-XIFI.eu/pub/bscw.cgi/d58659/XIFI-D4.2-Baseline\\_Tools\\_v1.pdf](https://bscw.fi-XIFI.eu/pub/bscw.cgi/d58659/XIFI-D4.2-Baseline_Tools_v1.pdf)