Grant Agreement No.: 604590
Instrument: Large scale integrating project (IP)
Call Identifier: FP7-2012-ICT-FI

eXperimental Infrastructures for the Future Internet

D3.6: Infrastructures Management Toolkit API v2

Revision: v.3.0

Work package	WP 3
Task	Task 3.3
Due date	31/12/2014
Submission date	08/01/2015
Deliverable lead	SYNELIXIS SOLUTIONS Ltd.
Authors	Th. Zahariadis (SYN), P. Trakadas (SYN), A. Papadakis (SYN), P. Karkazis (SYN), P. Athanasoulis (SYN), F. Facca (CREATE-NET), S. Cretti (CREATE-NET), A. Martellone (CREATE-NET).
Reviewers	E. Power (WIT), A. Alloush (TUB), D. Nehls (TUB)

Abstract	Deliverable D3.6 describes the second, updated version of the infrastructure management tools to be utilised within the XIFI federated infrastructure. These components are: the Infrastructure ToolBox (ITBox) and the Deployment and Configuration Adapter (DCA). The deliverable provides the necessary documentation of each of the components along with a section dedicated on the software updates developed after the submission of D3.3 (first version).
Keywords	Deployment, Configuration, bare metal, Generic Enabler, FIWARE

Document Revision History

Version	Date	Description of change	List of contributor(s)
V1.0	20.11. 2014	Table of Contents and assignments.	Th. Zahariadis (SYN), P. Trakadas (SYN)
V2.0	11.12.2014	First complete version of the deliverable.	Th. Zahariadis (SYN), P. Trakadas (SYN), A. Papadakis (SYN), A. Voulkidis (SYN), P. Karkazis (SYN), P. Athanasoulis (SYN), F. Facca (CREATE-NET), S. Cretti (CREATE-NET), A. Martellone (CREATE-NET)
V3.0	08.01.2015	Comments by the reviewers have been incorporated.	E. Powers (WIT), A. Alloush (TUB), D. Nehls (TUB).

Project co-funded by the European Commission in the 7th Framework Programme (2007-2013)		
Nature of the Deliverable:	P (Prototype)	
Dissemination Level		
PU	Public	✓
PP	Restricted to other programme participants (including the Commission Services)	
RE	Restricted to bodies determined by the XIFI project	
CO	Confidential to XIFI project and Commission Services	

[1] http://creativecommons.org/licenses/by-nc-nd/3.0/deed.en_US

EXECUTIVE SUMMARY

The main objective of XIFI is the design and development of the appropriate software, tools and procedures that will address the capacity building requirements of the pan-European federated cloud infrastructure, following and being compatible with the architecture and development activities of the FI-PPP programme, as a whole, but mostly with FIWARE project, and thus being able to accommodate Use Case projects' requirements. In this perspective, as also stated in the DoW, Task 3.3 provides the means to "*design and develop tools to facilitate the management of the XIFI federation and the FI enablers on top of the infrastructures available at the different nodes in the federation. In order to support deployment on top of infrastructures, we will develop a toolbox that will install a distribution with all the XIFI tools needed for the single node to manage its connection to XIFI federation. Moreover, data on Generic Enablers installed on a single node will be published to the XIFI federation, through APIs, to support add-on federation services developed in WP4*".

In other words, from a technical perspective, Task 3.3 deals with the design and development of the tools and adapters that, on one hand will take care of the processes required for a new infrastructure owner (from bare-metal) to become IaaS/PaaS compatible in XIFI federation level and, on the other hand, to provide/develop tools that will provide information on federation level regarding the deployment and configuration of the Generic Enablers (GEs) and Specific Enablers (SEs). The aforementioned objectives of Task 3.3 (and its accompanying deliverable) have been addressed by the introduction of two components in the XIFI architecture, namely Infrastructure ToolBox (ITBox) and Deployment and Configuration Adapter (DCA). In brief, ITBox is an IaaS deployment tool supporting the automated installation of the main components of a node (including host operating system, hypervisor, Data Centre Resource Manager (DCRM), monitoring and networking adapters) to become compatible with the XIFI federated infrastructure. In parallel to ITBox, Task 3.3 offers DCA, that caters for the management issues related to the deployment and configuration of multiple GEs and SEs in the federated XIFI infrastructure, the design and development of a component that will store all data related to GEs and SEs deployment and configuration and provide data to respective entities/components developed within XIFI, such as the Federation Manager, Resource Catalogue, Cloud Portal and Marketplace.

From a business perspective, this task (and the accompanying deliverable) provides information on the tools that are necessary for an infrastructure owner, in order to join the XIFI federation (from the IaaS/PaaS point of view), but also tools that will help application developers to review, compare and select GEs to deploy innovative services. The developments described herein, focused on the requirement that such processes and tools have to be as automated as possible in order to minimizing time-to-market and assist XIFI sustainability.

Being a "Prototype" in nature, this deliverable provides the source code and all the necessary accompanying documents (installation manual, user manual, developer guide, test cases) that a XIFI user (either an infrastructure owner or an application developer, as mentioned above) has to follow in order to deploy and configure the appropriate tools and GEs/SEs. Following the common branding strategy between FIWARE and XIFI, source code and respective documents for ITBox and DCA will become available through FIWARE Ops, in order to better position the results of XIFI and contribute to the overall success of FI-PPP and its recognition outside the FI-PPP.

Finally, it is important to highlight that this document structure (the text and the developments) as presented in this deliverable has taken into account and addressed the comments received by the reviewers during the XIFI project reviews.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ABBREVIATIONS

API	Application Programming Interface
CEP	Complex Event Processing
DC	Data Centre
DCA	Deployment and Configuration Adapter
DCRM	Data Centre Resource Manager
DEM	Datacentre and Enablers Monitoring
DoW	Description of Work
FI-PPP	Future Internet - Private Public Partnership
GE	Generic Enabler
IaaS	Infrastructure as a Service
IoT	Internet of Things
ITBox	Infrastructure Toolbox
NAM	Network Active Monitoring
NPM	Network Passive Monitoring
PaaS	Platform as a Service
SaaS	Software as a Service
SDC	Software Deployment and Configuration
SE	Specific Enabler
SLA	Service Level Agreement
UC	Use Case
URL	Uniform Resource Locator
VM	Virtual Machine

1 INTRODUCTION

This deliverable provides an in-depth technical analysis and description of the updated version of the ITBox and DCA functionalities, links to their available source code as well as the necessary documentation, including the installation guide, the test cases, the user guide and developer guide.

The motivation for the development of these components stems from the following two reasons:

1. There is a lack of a modular tool that allows for a highly automated installation and configuration of several bare-metal servers belonging to an infrastructure node willing to become part of the XIFI federation. Addressing this challenge, we are developing the Infrastructure Toolbox (ITBox) that offers IaaS FIWARE-compatible, ready-to-use servers (including the installation of the operating system, the hypervisor, the cloud management software and XIFI monitoring and network adaptation tools) through an appropriate API.
2. Management (including installation and monitoring) of several GEs, especially in the federated XIFI landscape, must be simplified as much as possible to assist XIFI (and FI-PPP) sustainability. To this end, the Deployment and Configuration Adapter (DCA) is developed whose main objective is to provide an adaptation layer between the Cloud Portal and other components (such as PaaS and SDC) that are responsible for the deployment and configuration of the Generic Enablers (GEs).

1.1 Purpose

The purpose of this deliverable is to describe the development of (a) the ITBox that aims at simplifying the installation and configuration of appropriate tools for an infrastructure owner to become compliant with XIFI IaaS/PaaS requirements, and (b) DCA that caters for the automated deployment, configuration and management of GEs and SEs to the XIFI federated nodes that will help application providers to easily review, compare and select the proper GEs/SEs to be deployed and configured in order to support the development of innovative services.

1.2 Main updates from deliverable D3.3

Deliverable D3.3 - Infrastructures Management Toolkit API [1] released on month 9 of the project provided a first description of the aforementioned software components in charge of adapting the infrastructure management functionalities that need to be provided in the XIFI federated environment. This first documentation helped the consortium to settle a stable framework, where an initial architecture served as the basis to implement the corresponding software components that have been deployed, upgraded and tested throughout this period.

Though the architecture provided in month 21 of the project maintains the main outlines defined, due to meaningful changes concerning particular and general issues, some important updates have been required and are presented in this document as an upgraded documentation. Nonetheless, thanks to the experience gathered, it is expected that this renewed architecture remains in time as stable, leaving future updates as improvements and additional features.

In particular, compared to the first release:

1. ITBox has been extended in the following ways:
   - ITBox deploys the OpenStack IceHouse release (2014.1.1);
   - All ITBox components (UI, puppet scripts and software packages) have been updated to the Fuel-Mirantis OpenStack 5.1 stable branches. These branches contain all fixes for defects reported by Fuel-Mirantis OpenStack community users and some new OpenStack projects such as Sahara (which provisions a Hadoop cluster), Murano (it introduces an application catalogue to OpenStack, enabling users to publish cloud-ready applications. It deploys these applications

by an external orchestrator tool such as Heat and Ceilometer (the OpenStack metering system, which acquires all the measurements in terms of virtual resources utilisation);

- Enabling of experimental and XIFI features through the configuration file settings.yaml. Experimental features provide functionality that may be useful to some users but that are not already tested in depth. Whereas, the XIFI features provide functionalities related to the project such as Nagios, Context Broker, OpenStackData Collector, and NGSI Adapter.
- Performing of the OpenStack release upgrade on an already deployed environment. ITBox takes advantage of a new feature available from the 5.1 release of Fuel - Mirantis OpenStack. Basically, the user is able to update an existing environment to a newer stable release automatically through the UI.
- Starting from this version, both the user interface and API are protected by the users' credentials.

2. DCA has been extended in the following aspects:
   - DCA has been integrated with Identity Management (IdM) component. Integrating DCA with IdM offers security advantages, as only authorised users can gain access the API provided by DCA. This integration was deemed mandatory since accessing DCA API a user is capable of retrieving information on all nodes consisting XIFI federation.
   - DCA API has been extended to address the reviewers' comments during M12 review of XIFI. In more detail, DCA includes RESTful API methods for 1) returning a list of the most popular GEs in a XIFI node (or across federation), 2) returning a list of active GEs in a specific time period, 3) returning a list of GEs that have been deployed in a specific time period, 4) returning a list of GEs deleted in a specific time period.
   - Apart from collecting, maintaining and correlating information regarding GE deployment through images, DCA has been extended to support GE deployment through blueprints. In this case, VMs launched as part of a blueprint instance are related with metadata items 'nid' and 'region'. This process offers compatibility with the existing one, regarding GE deployment via images and offers the ability to maintain a list of GE instances deployed through blueprints.
   - Moreover, DCA has been extended to support the aforementioned GE functionality for Specific Enablers (SEs). A GET method to Repository component is used to gather data regarding the registered SEs, along with a plethora of information. The information is properly stored in a separate table in DCA, offering newly developed API methods for retrieving information regarding SEs. Thus, DCA is capable of offering valuable data to interested users (Marketplace and SLA Manager components, developed in WP4).
   - Finally, DCA has been extended to include information regarding GEs (and SEs) that are offered as SaaS. This new functionality has been added to support Marketplace component (WP4). Three API methods have been developed, allowing a user to 1) add (POST operation) a new SaaS offering (Add new SaaS), 2) retrieve (GET operation) SaaS offerings per node (or across XIFI federation) and 3) remove (DELETE operation) a SaaS.

1.3 Overview

The structure of this deliverable is as follows: the *Summary* section provides a brief description of the ITBox and DCA components. The *Motivation* section deals with the reasoning for the design and development of these components, while *State of the Art* section provides the rationale for the selection of components to be used, extended and adapted. The next section, *Architecture Design*, presents the functionality that each component provides, along with the dependencies and inter-connections to other components (either FIWARE or XIFI). Finally, the last sections include detailed description of the *Installation Manual*, *Test Cases*, *User Manual* and *Developer Guide* that will

become publicly available through FIWARE Ops and will help infrastructure owners and application developers to deploy and test the functionality of the respective components.

2 COMPONENTS

This section presents in details the two components ITBox and DCA.

2.1 Infrastructure ToolBox (ITBox)

2.1.1 Summary

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. ITBox distribution includes the deployment of the necessary XIFI components, tools and GEs to complete the XIFI node installation. Figure 1 shows the detailed scheme of the XIFI Reference Architecture and the location of the Infrastructure Toolbox.

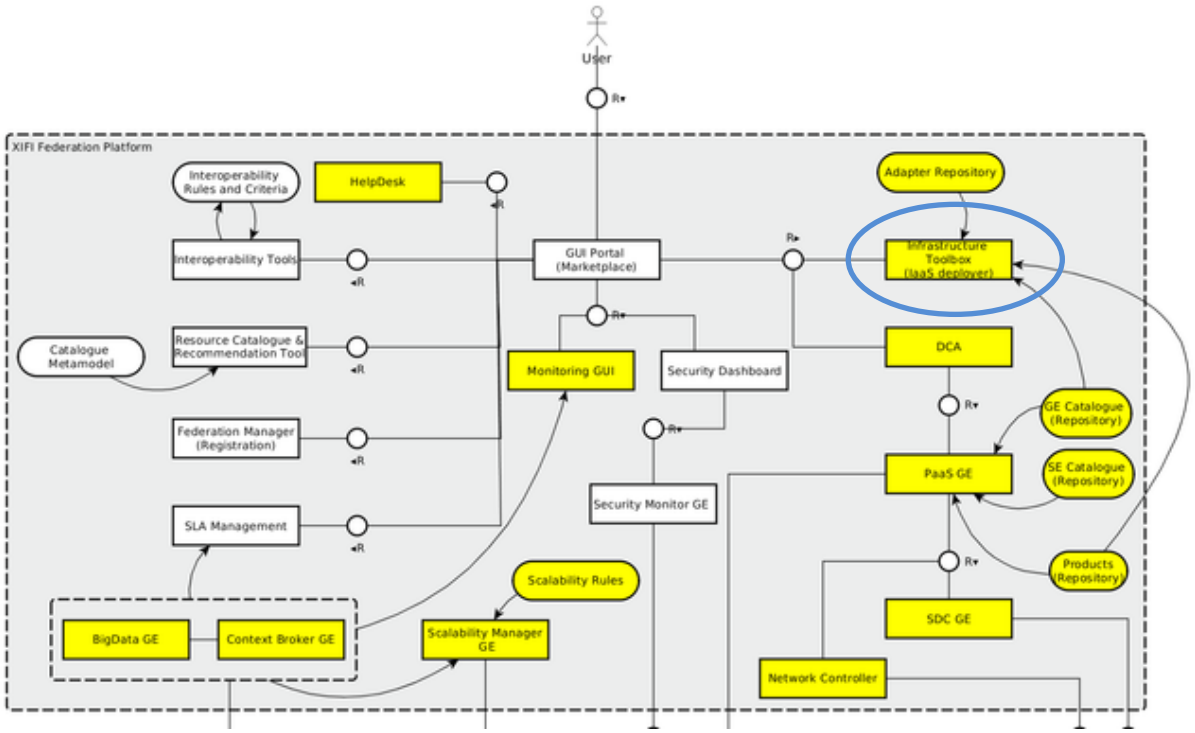

Figure 1: Location of the ITBox in the XIFI Reference Architecture

Reference Scenarios	UC 1 - Joining the Federation
Reference Stakeholders	**Developers** who want to install a private XIFI/FIWARE LAB node. **Infrastructure owners** who want to join XIFI/FIWARE LAB federation or build a private XIFI/FIWARE LAB node.
Type of ownership	Extension
Original tool	Fuel - Mirantis OpenStack Mirantis OpenStack 5.1

Planned OS license	Apache License Version 2.0. As the original tool.
Reference OS community	OpenStack

Table 1: ITBox Context Details

Consist of	• ITBox Interface • ITBox Middleware
Depends on	• Network Active Monitoring (NAM) • Datacentre and Enablers Monitoring (DEM) • Network Passive Monitoring (NPM) • OpenStack Data Collector • Data Centre Resource Management (DCRM) • Context Broker • NGSI Adapter

Table 2: ITBox Dependencies Summary

2.1.2 Component Responsible

Developer	Contact	Partner
Alessandro M. Martellone	alessandro.martellone@create-net.org	CREATE-NET

Table 3: ITBox Reference Details

2.1.3 Motivation

The deployment of a large distributed infrastructure is a complex task that requires automation to scale. The main goal of ITBox is to help the cloud environments administrators to organise and manage a large variety of infrastructures and services, thus obtaining a well-defined deployment and maintenance process. Furthermore, this permits building a more coherent and tested installation within the FIWARE Lab federation guaranteeing as well as a better deployment and a more manageable issues resolution. From the overall scenario, we derived a number of cases that represent requirements to be supported by the tool. Developments occur according to elicited priorities (by infrastructure owners or external adopters) and available resources in the project.

Section 2.1.5, *State of the Art*, provides an overview of available open source tools, which we analysed prior to selecting Fuel - Mirantis OpenStack [3] as the starting point. Fuel - Mirantis OpenStack has been selected mainly for the following reasons:

- it natively supports OpenStack (required to create a new FIWARE Cloud instance);
- the graphical interface is intuitive;
- it is incubated in OpenStack;
- it is a mature and stable solution;
- it natively supports a number of components and projects which are appropriate for the XIFI case;

- Mirantis is an important contributor in the OpenStack community;
- better support in OpenStack project integration with respect to its competitors;
- Mirantis OpenStack uses Puppet which has a large catalog of scripts already developed.

2.1.4 User Stories backlog

In the following the user stories backlog of the ITBox are represented.

Id	User story name	Actors	Description
1	Support Ubuntu LTS 12.04	Infrastructure Administrator (IA), Infrastructure Toolbox (ITBox)	The IA installs Ubuntu LTS 12.04 on the node through the Preboot eXecution Environment (PXE).
2	Include support for configuration without separate cinder node	Infrastructure Administrator (IA), Infrastructure Toolbox (ITBox)	The IA can choose, using user interface (UI), if to install cinder on compute node or in a separate node. If a cinder node is not specified then the compute node acts automatically also as a cinder node. Moreover, the Infrastructure Toolbox verifies, in compliance with DCRM set-up, if the Network File System (NFS) is the current solution for cinder deployment.
3	Deploy DCRM using CLI	Infrastructure Administrator (IA), Infrastructure Toolbox (ITBox)	The IA deploys DCRM using command line interface (CLI), setting node id and the others requirement parameters. Once the installation of DCRM is successfully completed, the IA can select the OpenStack scheduler among default, Pivot, Pulsar.
4	Deploy DCRM using UI	Infrastructure Administrator (IA), Infrastructure Toolbox (ITBox)	The IA can deploy DCRM using UI. Once the installation of DCRM is successfully completed, the IA can select the OpenStack scheduler among default, Pivot, Pulsar.
5	Deploy the Identity Manager (IdM) GE and the Access Control (AC) GE using CLI	Infrastructure Administrator (IA), Infrastructure Toolbox (ITBox)	The IA decides to install, on an already provisioned node, the Identity Manager (IdM) GE and the Access Control (AC) GE using CLI. He launches deploy command, setting node id and the others requirement parameters.
6	Deploy the Identity Manager (IdM) GE and the Access Control (AC) GE using GUI	Infrastructure Administrator (IA), Infrastructure Toolbox (ITBox)	The IA selects a deployed node. The ITBox displays details about it and a list of installable GEs. The IA selects the Identity Manager (IdM) GE and the Access Control (AC) GE and next he starts the installation.
7	Deploy the Monitoring GE and adapters using CLI	Infrastructure Administrator (IA), Infrastructure	The IA decides to install, on an already provisioned node, the Monitoring GE and its related adapters using CLI. He launches update command, setting node id and the

Id	User story name	Actors	Description
		Toolbox (ITBox)	others requirement parameters.
8	Deploy the Monitoring GE and adapters using GUI	Infrastructure Administrator (IA), Infrastructure Toolbox (ITBox)	The IA selects a deployed node. The ITBox displays details about it and a list of installable GEs. The IA selects the Monitoring GE and its related adapters and finally he starts the installation.
9	Send deployment test report to Federation Manager	Infrastructure Administrator (IA), Infrastructure Toolbox (ITBox)	The IA selects the current environment and runs tests on the deployed nodes. When the tests are finished, then ITBox will send a report to Federation Manager.
10	Enable Quantum using GUI	Infrastructure Administrator (IA), Infrastructure Toolbox (ITBox)	The IA selects a provisioned node. The ITBox displays details about it and a list of available options. The IA selects Quantum and he fills its parameter fields (e.g. tenant network type {gre|vlan}, segment range and so on).
11	Include deployment of Object Storage using CLI	Infrastructure Administrator (IA), Infrastructure Toolbox (ITBox)	The IA decides to deploy a new Object Storage node into current infrastructure. He launches deploy command, setting node id and the others requirement parameters.
12	Include (when available) test scenarios from WP2 continuous integration	Infrastructure Administrator (IA), Infrastructure Toolbox (ITBox)	The IA selects the current environment and runs WP2 continuous integration tests on the deployed infrastructure. When the tests are finished, then ITBox will send a report to Federation Manager.
13	Recommender for matching servers with roles according to their capacity	Infrastructure Administrator (IA), Infrastructure Toolbox (ITBox)	The IA selects the current infrastructure. The ITBox shows the list of discovered nodes and in according to their computational capacity it suggests the best role for each node.
14	Upgrade to OpenStack IceHouse release	Infrastructure Administrator (IA), Infrastructure Toolbox (ITBox)	The IA installs a new version of ITBox server.
15	Migration of all components installation from the 1.3.4.1 release to the 2.0 release	-	-
16	Include deployment of NAM using GUI.	Infrastructure Administrator (IA), Infrastructure Toolbox (ITBox)	The IA selects a deployed node. The ITBox displays details about it and a list of installable components. The IA selects the NAM and finally he starts the installation.
17	Enable experimental features through the configuration file	Infrastructure Administrator (IA), Infrastructure Toolbox (ITBox)	The IA configures the master node, enabling or disabling the experimental features.
18	Perform an OpenStack release upgrade on an already deployed	Infrastructure Administrator (IA), Infrastructure	The IA selects an environment and upgrades it choosing the right OpenStack release from a

Id	User story name	Actors	Description
	environment through the GUI.	Toolbox (ITBox)	list of available packages.
19	Protect by user credentials the GUI and the CLI.	Infrastructure Administrator (IA), Infrastructure Toolbox (ITBox)	The IA performs an action only if he is authenticated on the system.

Table 4: ITBox User Stories Backlog

2.1.5 State of the Art

FIWARE developed the SDC tool [2] to support GE installation. However, the deployment of IaaS DCRM and related tools in a similar manner was not supported. This is of primary importance to build the so-called Infrastructure Toolbox. A number of tools exist, to support deployment of IaaS solutions, some are OpenStack focused and other are IaaS solution independent.

In this section we provide a comprehensive coverage of current deployment tools. The tool characteristics that we have evaluated are the following:

- it must be an open source project;
- the community size must be large enough;
- it should be preferably OpenStack oriented

We have selected and analysed the following projects:

- Mirantis OpenStack: a tool focused on OpenStack, developed by Mirantis [3];
- OpenCrowbar: an independent solution that covers the complete lifecycle management tool for physical and virtual servers [4];
- Rackspace Private Cloud: a tool that enables users to deploy a private cloud OpenStack cluster configured according to the recommendations of Rackspace OpenStack specialists [5].

An OpenStack cloud installation consists of many packages from different projects, each with its own requirements, installation procedures and configuration management.

The scope of these tools is to help the cloud environments administrators to organise and manage a large variety of infrastructures and services, thus to have a systematic deployment and maintenance process.

Fuel - Mirantis OpenStack

Fuel - Mirantis OpenStack is an open source deployment tool, designed for deploy an OpenStack environment. Mirantis OpenStack uses Puppet, an IT automation software (it provides, through appropriate scripts, an easy way to automate repetitive tasks). It automates provisioning and deployment of all core OpenStack components including Nova, Glance, Horizon, Swift, Keystone, Neutron/Quantum, Heat, Ceilometer, Savanna, Murano, Ceph and Cinder.

In the last release 5.1, it installs OpenStack IceHouse version both on CentOS and Ubuntu 12.04 LTS. Fuel - Mirantis OpenStack is composed of a Fuel Master server (where all required modules are installed in a Docker container, such as the provisioning agent, the web user interface, the servers' discovery agent) which contains a set of Puppet scripts and all essential OpenStack packages.

When the Master Node is installed, the cloud infrastructure administrator can discover his virtual or physical servers. Through the Mirantis OpenStack UI, the environment administrator can assign a role

(e.g. Controller, Compute, Cinder, Monitoring and so on) to each server, configure storage and network and other options. Finally, the environment administrator can deploy OpenStack on servers.

One of the advantages of Fuel - Mirantis OpenStack is that it comes with a number of pre-built deployment configurations that can be used by users to build their OpenStack cloud infrastructure. These are well-specified configurations of OpenStack, according to the best practices recommended by OpenStack specialists.

OpenCrowbar

OpenCrowbar is a deployment tool developed by Dell under Apache 2 license, which using a set of Chef [10] recipes can deploy OpenStack, Hadoop and Ceph environments. It is the successor of Crowbar Project [11]. As Mirantis OpenStack, also OpenCrowbar requires installing a Master node (based on CentOS 6.5) on a VM or on a physical server. When the installation is finished, the user can deploy a new OpenStack cluster, launching the Admin UI at http://[Admin IP]:3000 using a standard web browser. Once the Crowbar node is running, simply booting VMs or physical nodes on the management network will engage the discovery process (using DHCP and PXE) and they will be added into the system deployment.

When the cluster nodes are all discovered, the user can assign to each of them a specific role (e.g. controller or compute) and to start the deployment.

Rackspace Private Cloud

Rackspace Private Cloud (RPC) has been developed by Rackspace under Apache 2 License. It enables users to quickly deploy a private cloud OpenStack cluster on Ubuntu, CentOS, or RHEL operating systems, according to the recommendations of Rackspace OpenStack specialists.

The latest version of Rackspace Private Cloud (version 9) installs OpenStack IceHouse and the following services: Keystone, Glance, Neutron, Cinder and Heat. Also provides the following infrastructure, monitoring, and logging services to support OpenStack:

- Galera with MariaDB
- RabbitMQ
- Memcached
- Rsyslog

RPC v9 uses a combination of Ansible and Linux Container (LXC) to install and manage an Openstack deployment. Ansible is an IT automation deployment tool which uses “playbooks” in YAML language to orchestrate installation processes.

2.1.6 Architecture Design

In order to support the installation, updating and managing of XIFI nodes, a modular tool has been designed, named Infrastructure Toolbox (ITBox). Its architecture is shown in Figure 2.
The architecture design has been driven by the analysis of existing tools that allow bare-metal deployment of IaaS platforms.

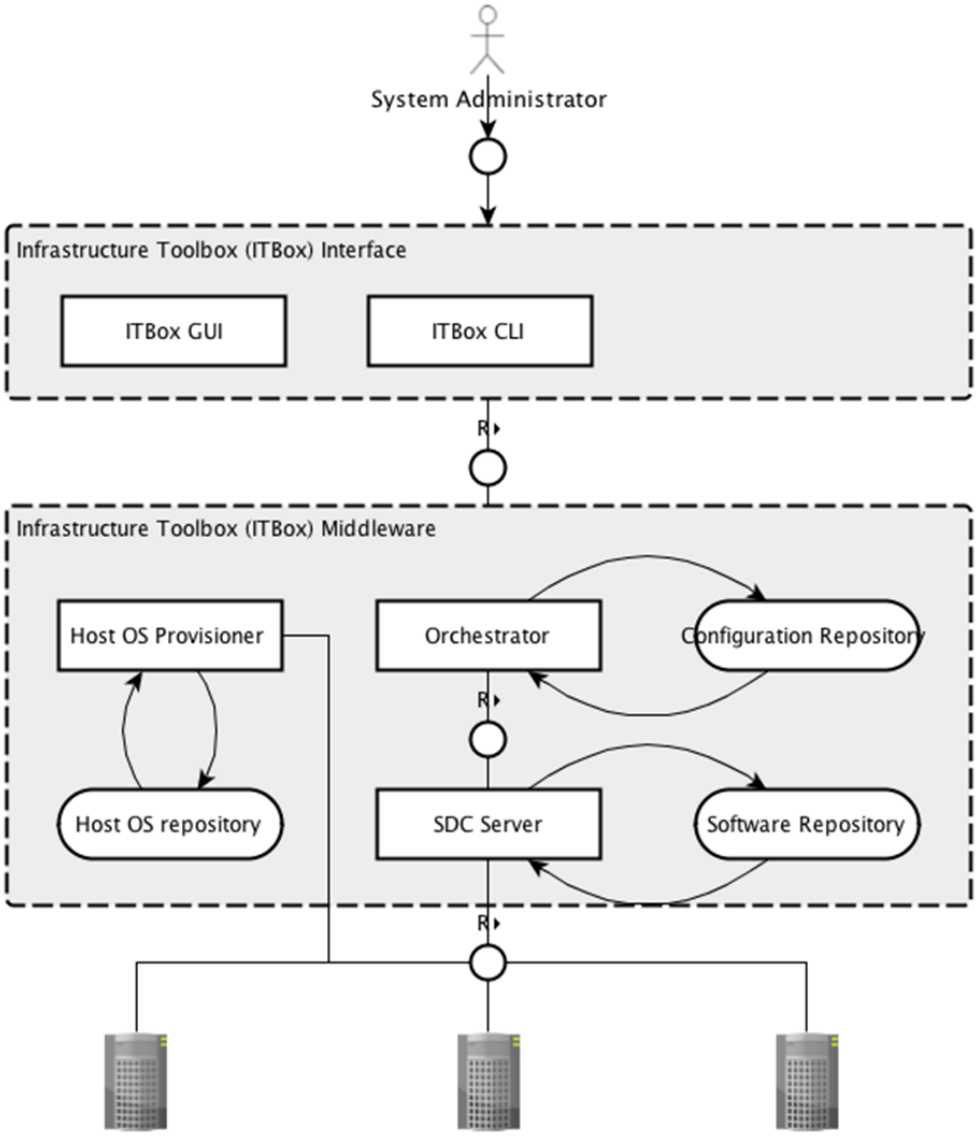

Figure 2: ITBox Reference Architecture

The Infrastructure Toolbox architecture is composed of an interface layer named the Infrastructure Toolbox Interface (shown in the upper box) and of the Infrastructure Toolbox Middleware (lower box).
We now see in detail how these two layers are composed.

- ITBox Interface that allows the System Administrator interfaces to run the set-up and installation of a new XIFI node. In particular the Interface provides:
  - ITBox Graphical User Interface: a web based interface with simplified functionalities for set-up and installation.
  - ITBox Command Line Interface: a shell based interface with advanced functionalities for set-up and installation.
- ITBox Middleware that provides the services that run the actual provision and deployment of OS and services on top of bare-metal infrastructure.
  - The Host Operating System Provisioner: a server that installs via network operating system on node discovered via PXE protocol or similar. It has a repository of host operating systems that can be provisioned on the bare-metal servers.

- The Orchestrator: a server that coordinates the deployment of services on the different servers according to the configuration passed by the ITBox Interface. The orchestration among different servers is needed to ensure proper set-up and configuration of the XIFI node (e.g. knowing when a cloud controller server is ready to register compute servers). The orchestrator relies on a repository of scripts.
- The Software Deployment and Configuration (SDC): a server that provides access to packages and scripts for the installation of services on a single node. It relies on a Software and Scripts repository. (Note: this service has the same role as the FIWARE SDC GE, but a different scope, i.e. it is not meant for the installation of GEs on top of VMs provisioned by the DCRM, but for the installation of DCRM itself and additional services).

2.1.7 Release Plan

Version Id	Milestone	User Stories
1.0	30.11.2013	1,2
1.1	30.12.2013	3,4, 5, 6, 10
1.2	30.04.2014	7,8,9
1.3	30.10.2014	11,12
2.0	30.12.2014	13, 14,15, 16, 17,18,19

Table 5: Release Plan

2.1.8 Test Cases

In order to verify the correct installation of the ITBox, the user can perform the following command:

```
fuel --os-username admin --os-password admin release
```

The output generated by the command is the list of available releases for each Operating System supported:

```
id | name                         | state     | operating_system | version
---|------------------------------|-----------|------------------|-------------
1  | Icehouse on CentOS 6.5       | available | CentOS           | 2014.1.1-5.1
2  | Icehouse on Ubuntu 12.04.4   | available | Ubuntu           | 2014.1.1-5.1
```

2.1.9 Installation Manual

ITBox is distributed as an ISO image that contains an installer for ITBox Master Server. The ISO can be installed in the same way, using a virtualisation 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 [7], all other servers can be powered on, and the user can login into the ITBox User Interface (UI) using the default address http://10.20.0.2:8000/, or they 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.

2.1.10 User Manual

When the user has completed the Master Node installation, they can access ITBox UI, visiting the default URL http://10.20.0.2:8000/, as depicted in Figure 3.

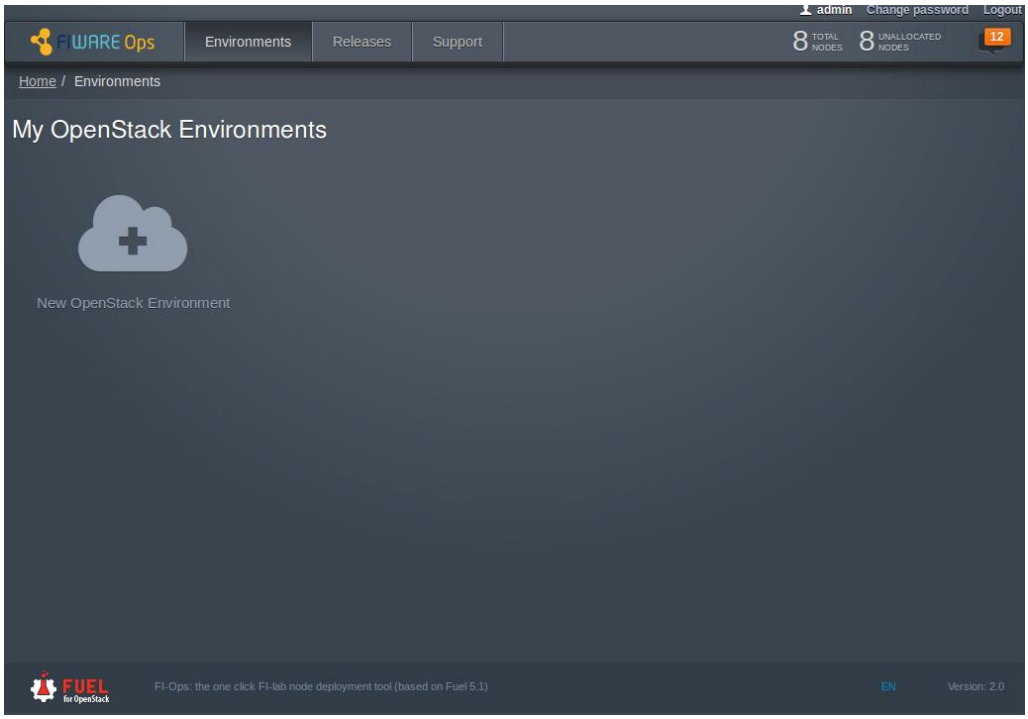

Figure 3: ITBox homepage

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 about discovered nodes on ITBox UI (see Figure 3 in the upper right corner). At this moment, the user could create a new environment.

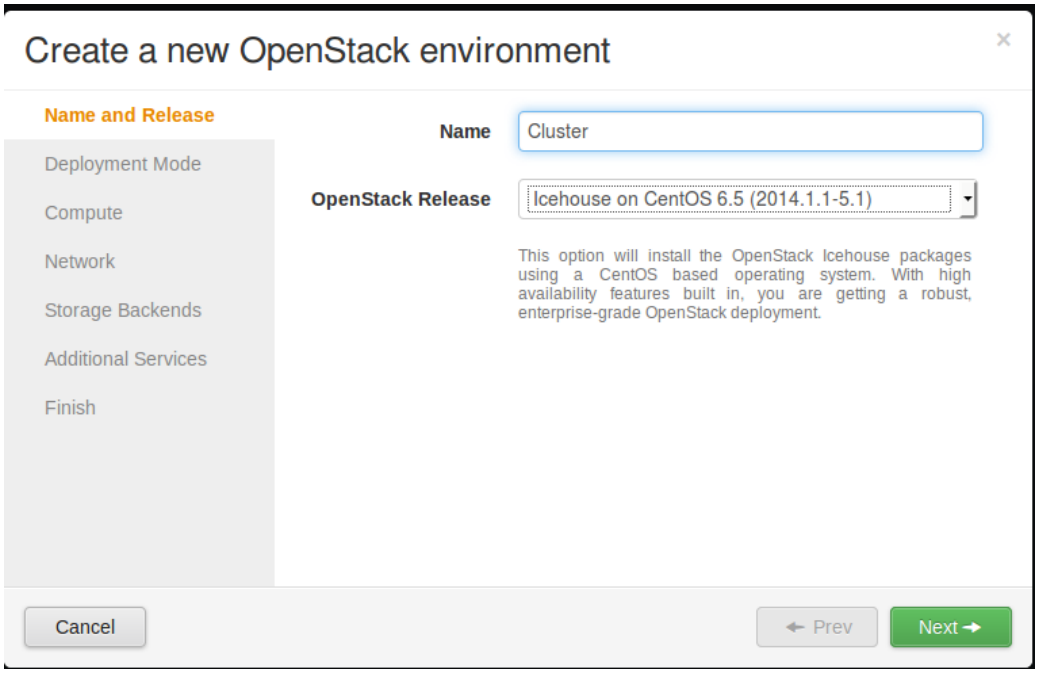

Figure 4: Creation of a new environment

The first step that involves the user is the "New OpenStack Environment" creation (Figure 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, network manager (Nova-Network, Neutron with GRE, Neutron with VLAN, Neutron with VMware NSX plugin) and storage backends (the user could choose as Cinder and Glance backend either their default implementations or that implemented by Ceph – Figure 5).

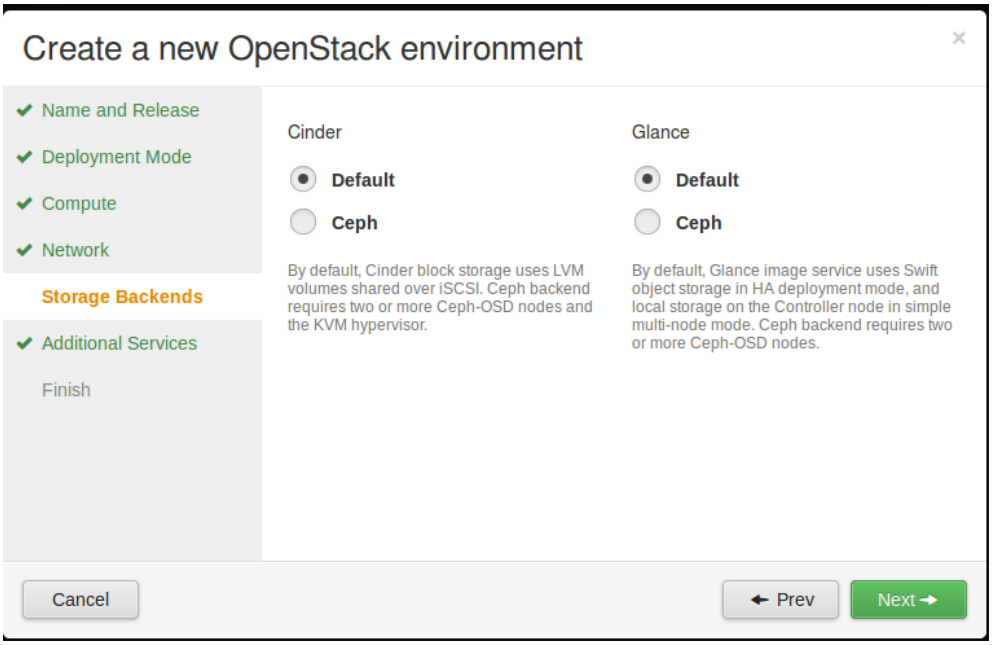

Figure 5: Configuration of storage backends

Finally, the user chooses to install Sahara, Murano or Ceilometer (Figure 6).

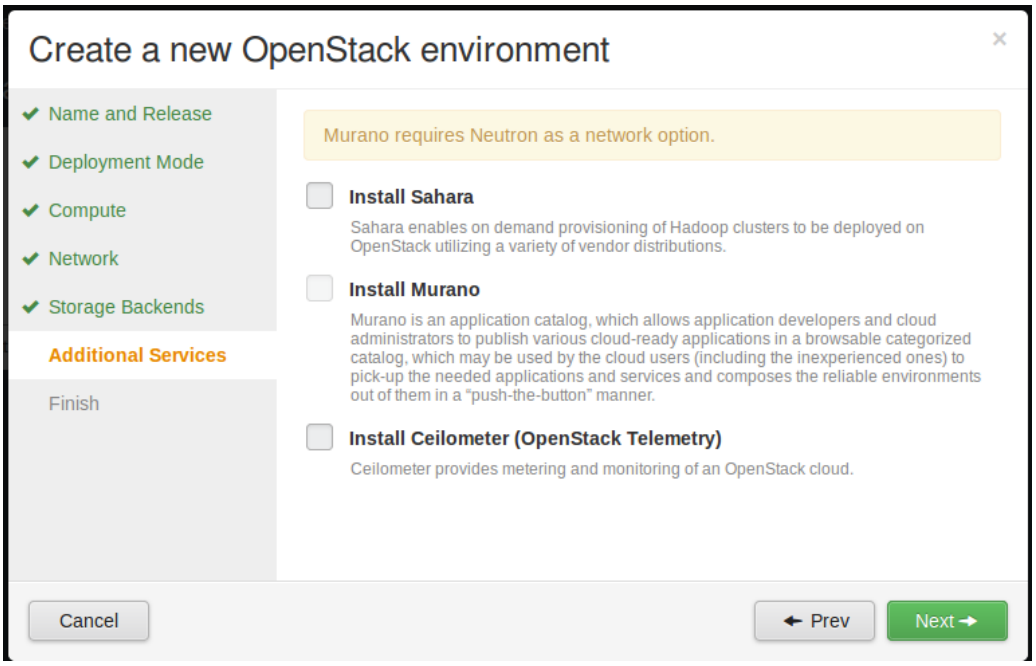

Figure 6: Installation of additional services

Now the environment is ready for deployment (Figure 7, Figure 8).

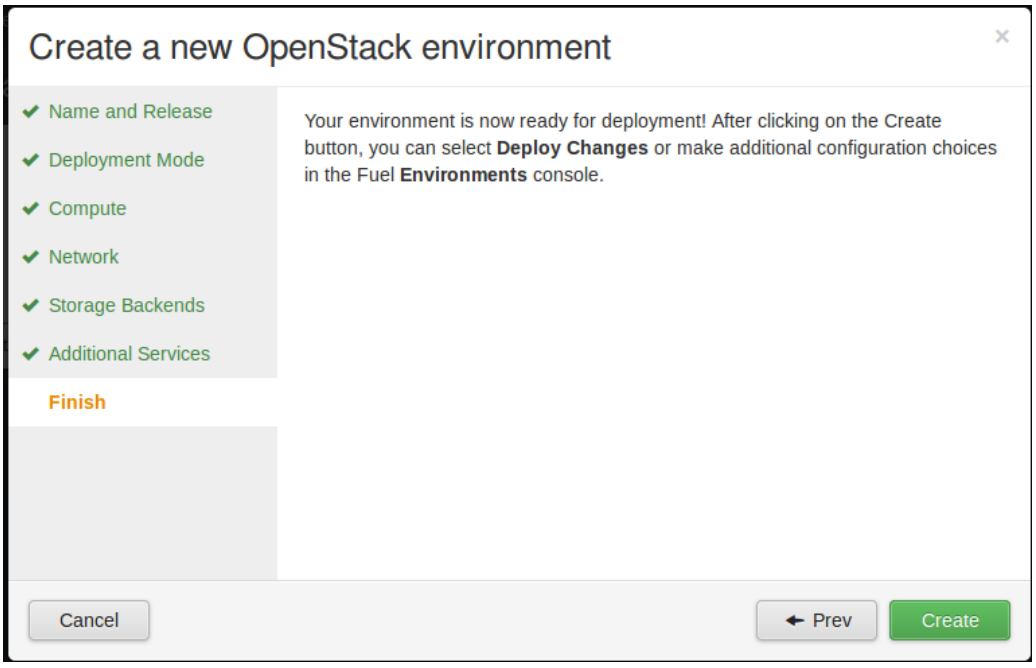

Figure 7: Final creation step

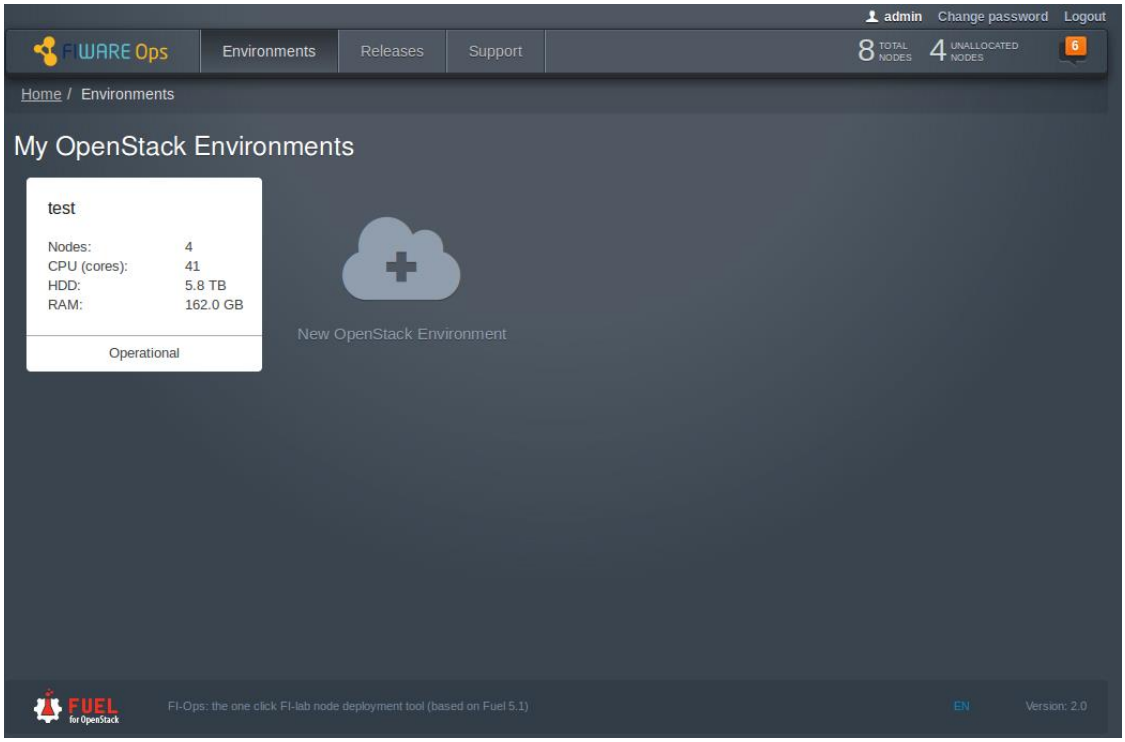

Figure 8: The page of the created environment

In the environment creation process, the user should define the architecture of his cloud infrastructure. The user assigns a role to every server, configures the network, defines the space allocated to hard disks and sets other OpenStack options (Figure 9).

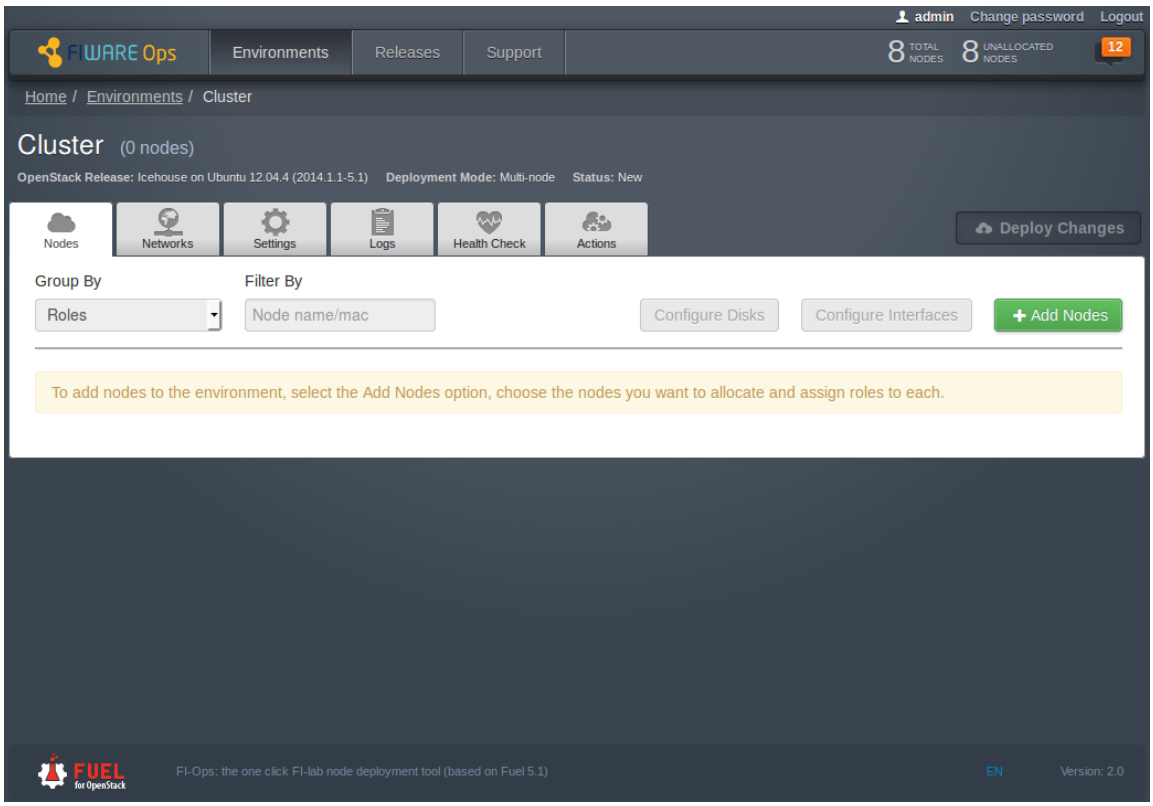

Figure 9: Settings of the environment

Giving roles to servers

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

By clicking on the “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 on the same host, and so on. Finally the user applies the changes (Figure 10).

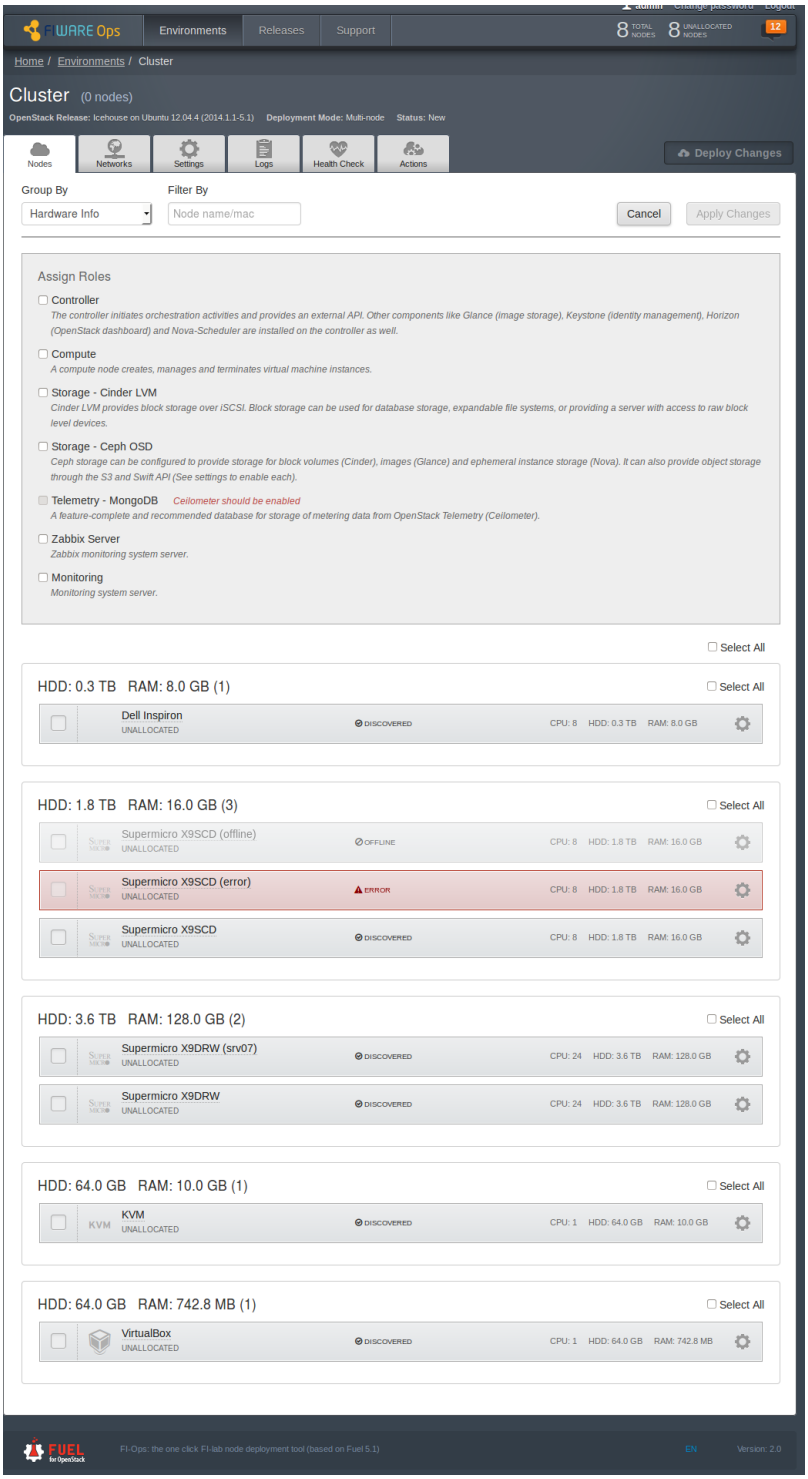

Figure 10: The list of the available servers

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.) (Figure 11, Figure 12, Figure 13).

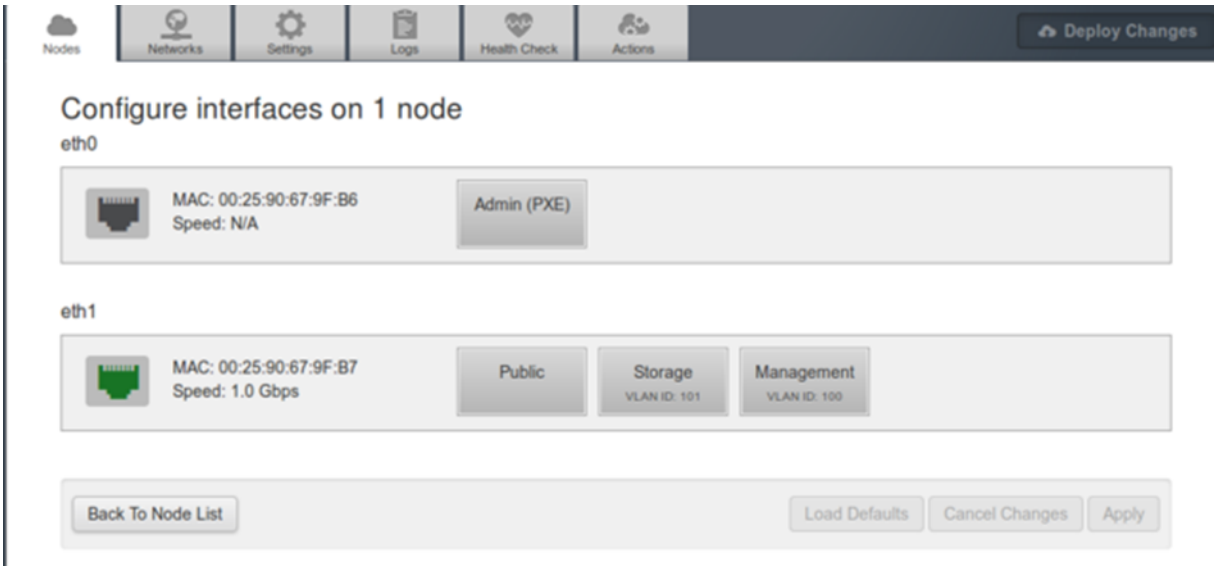

Figure 11: Network interfaces configuration

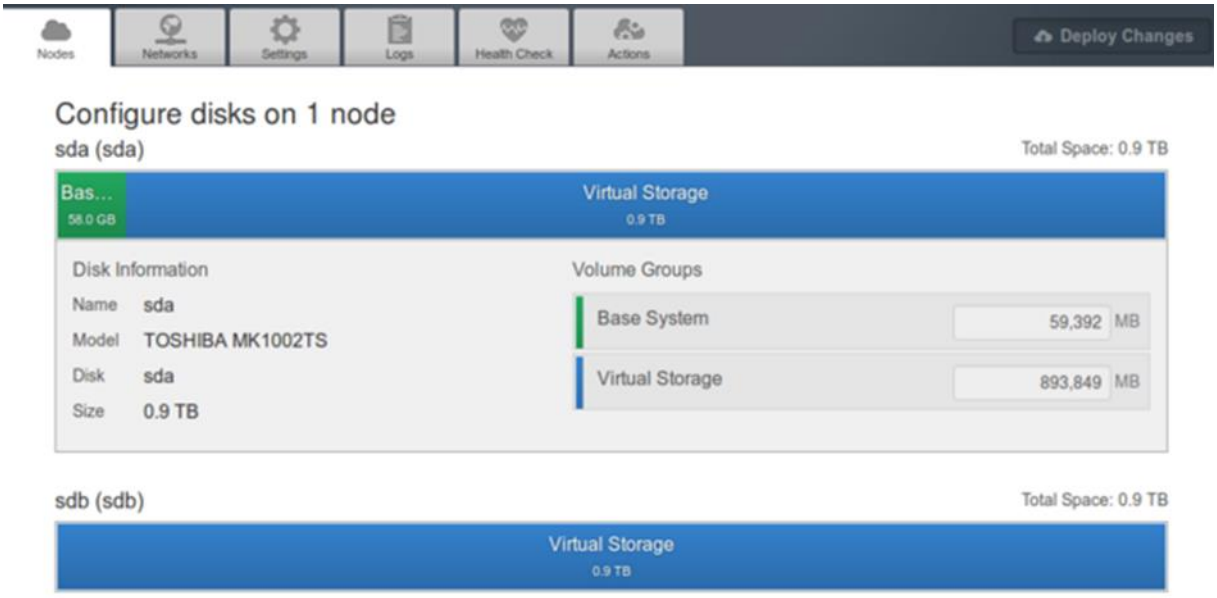

Figure 12: Hard disks configuration

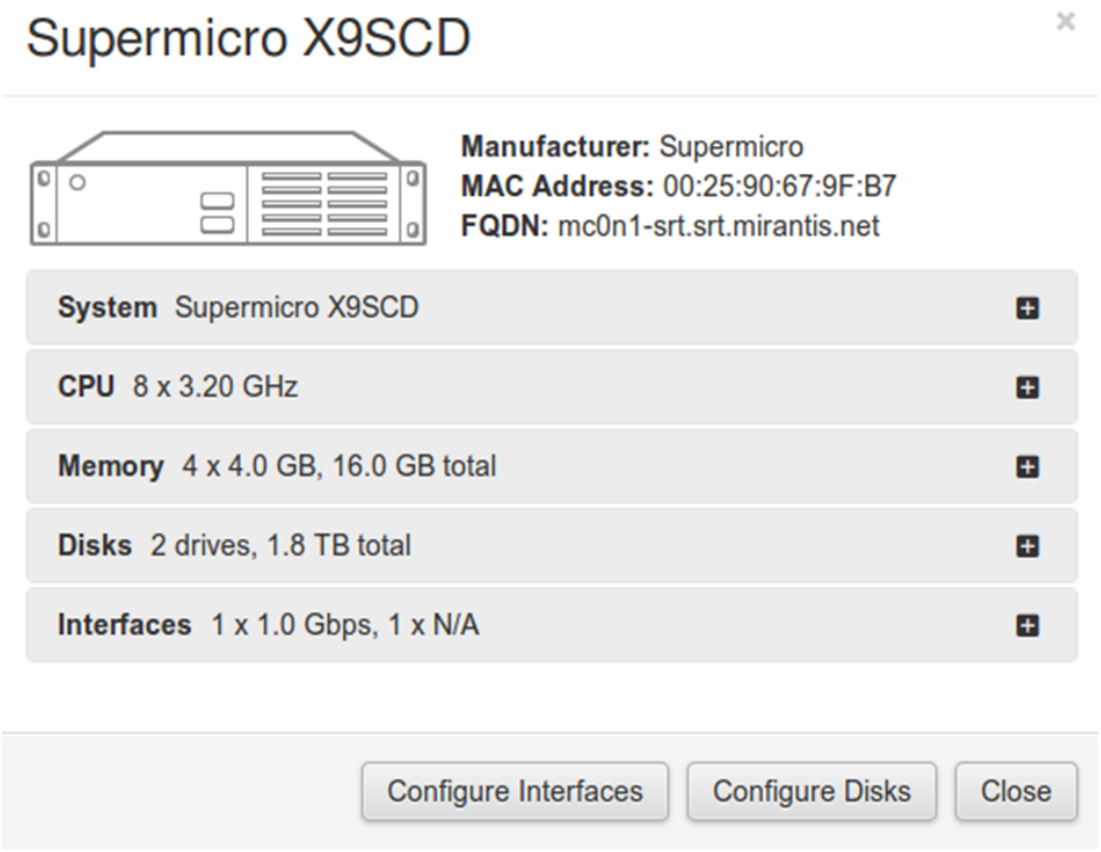

Figure 13: Detailed information about the selected server

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 the user to set the IPs pool and its VLAN tag (Figure 14).

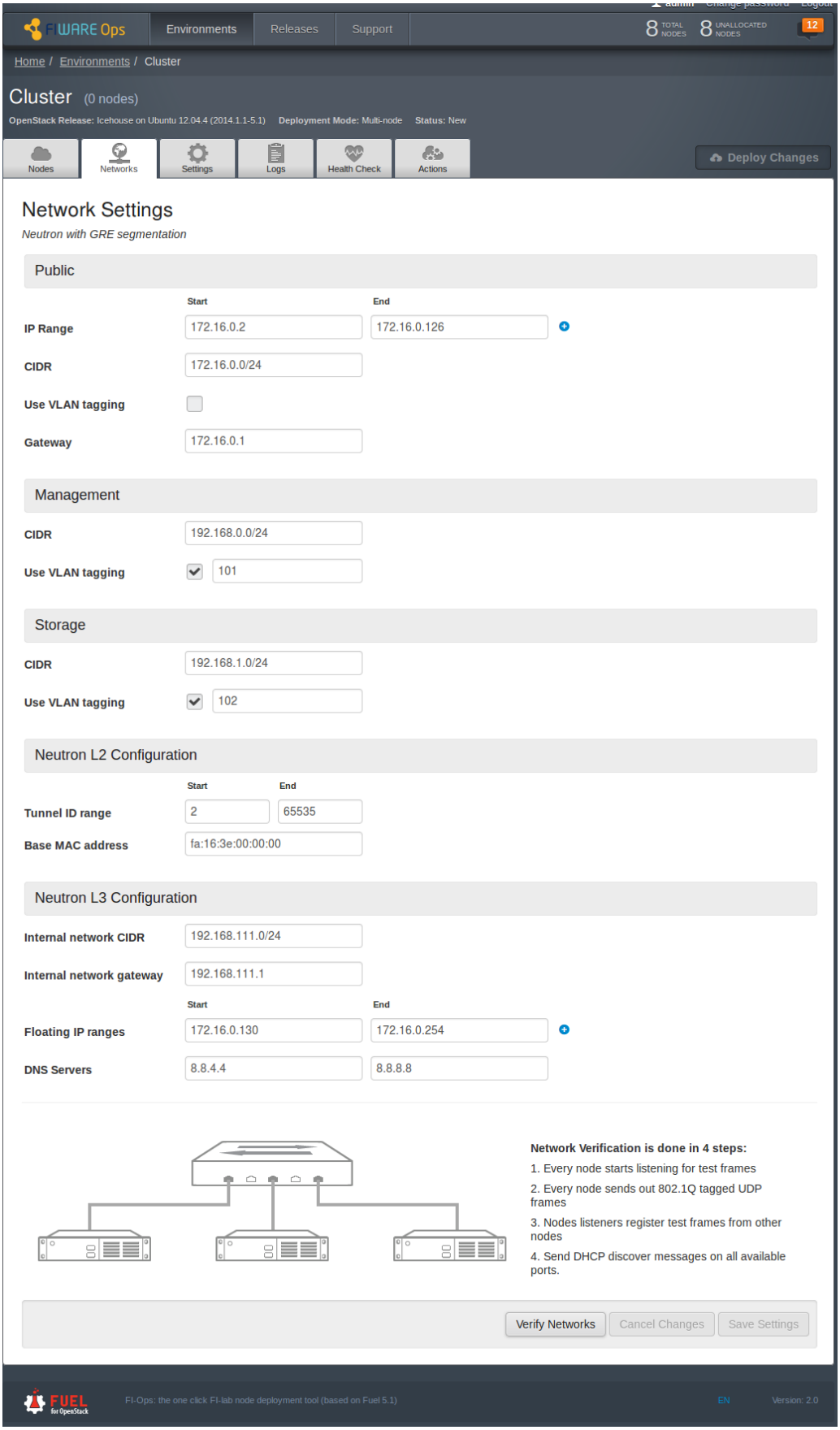

Figure 14: Infrastructure network settings

The ITBox gives the 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 some external DHCP server might interfere with the current deployment (Figure 15).

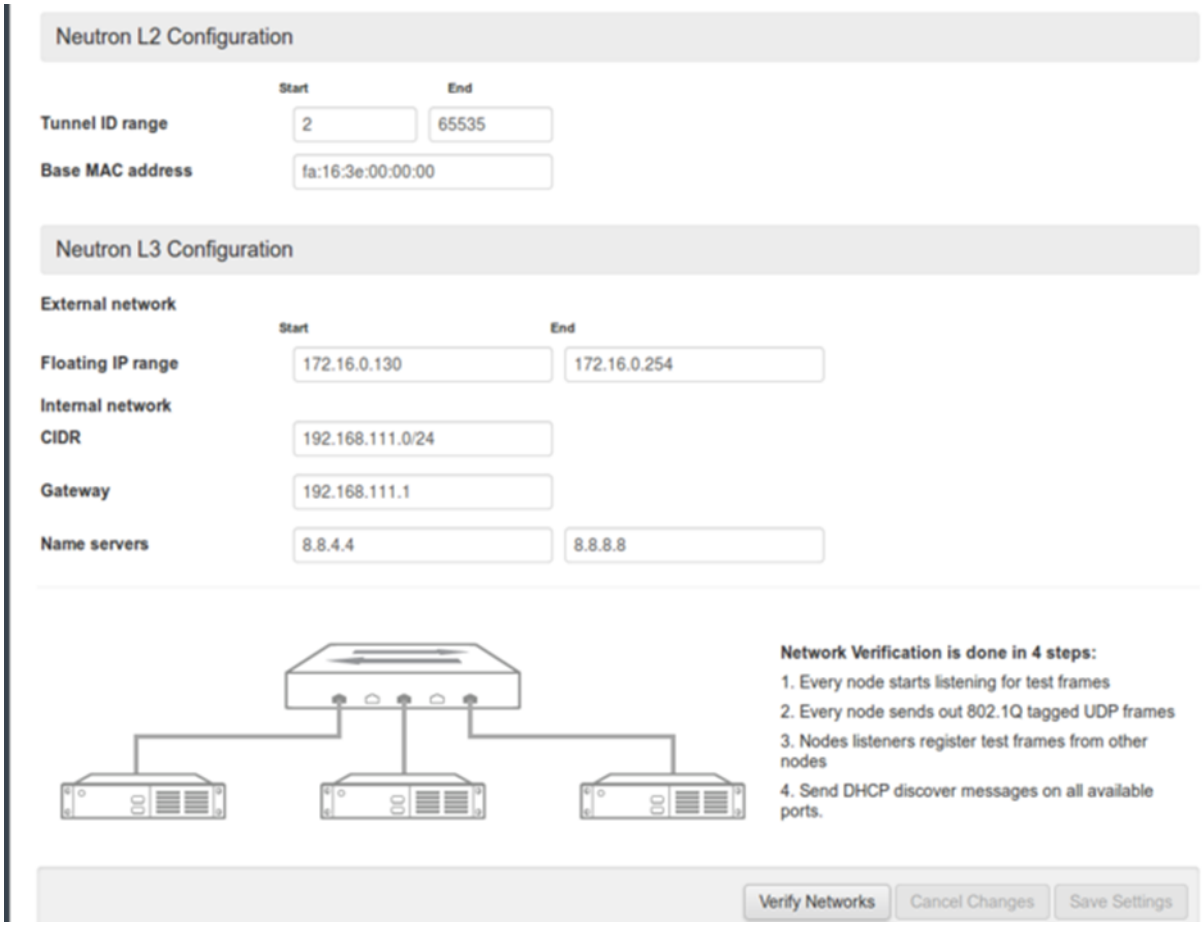

Figure 15: L2/L3 Neutron configuration

General Settings

The "Settings" tab contains useful options for managing the current environment. For example, the user can change the OpenStack admin account, or the hypervisor type, the scheduler driver as well as configure the syslog or define a public key in order to access to the deployed servers. To make changes permanent it is necessary to re-deploy the changes. (Figure 16, Figure 17).

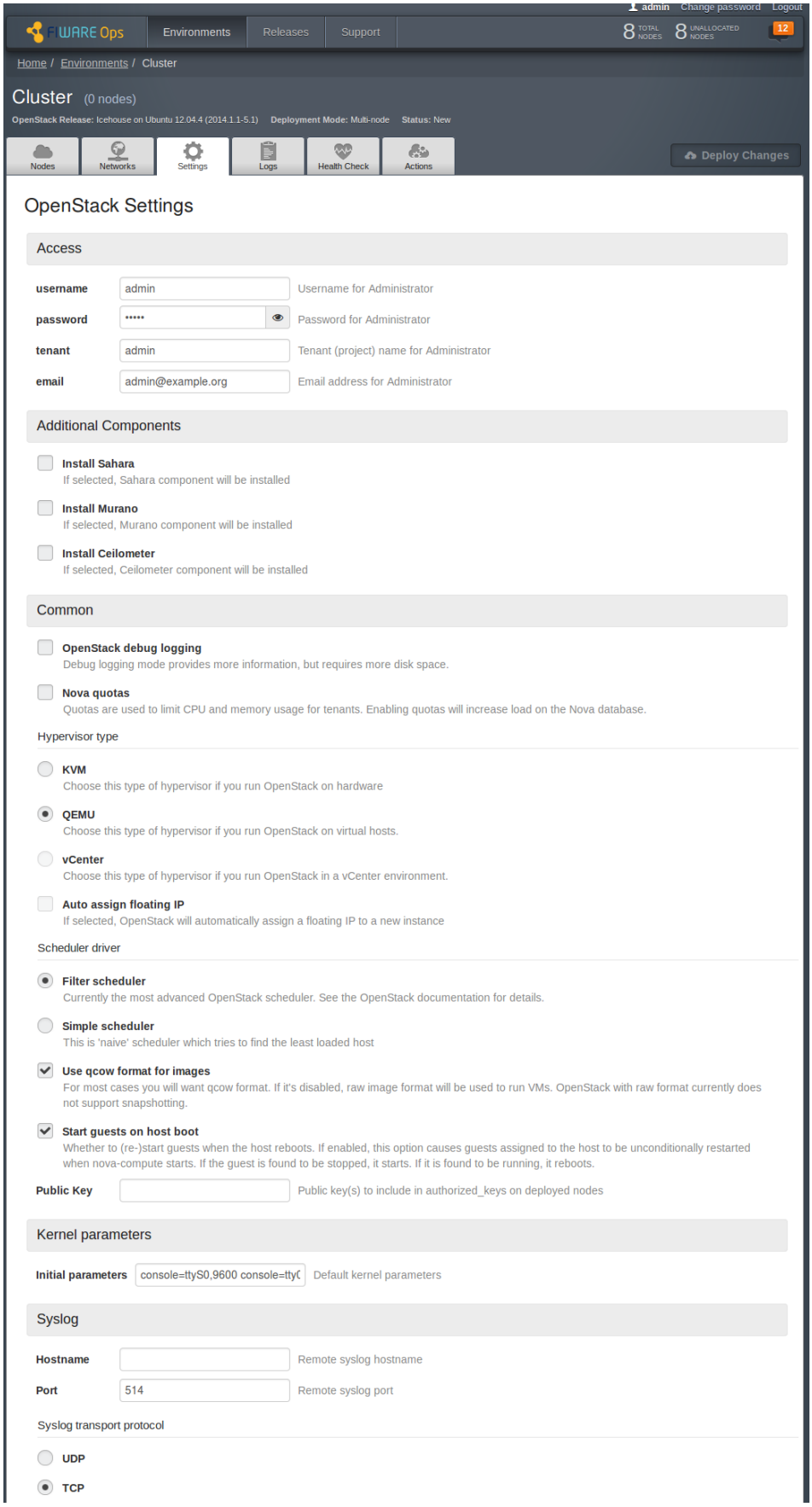

Figure 16: Infrastructure settings (additional components, administrator account, syslog, common)

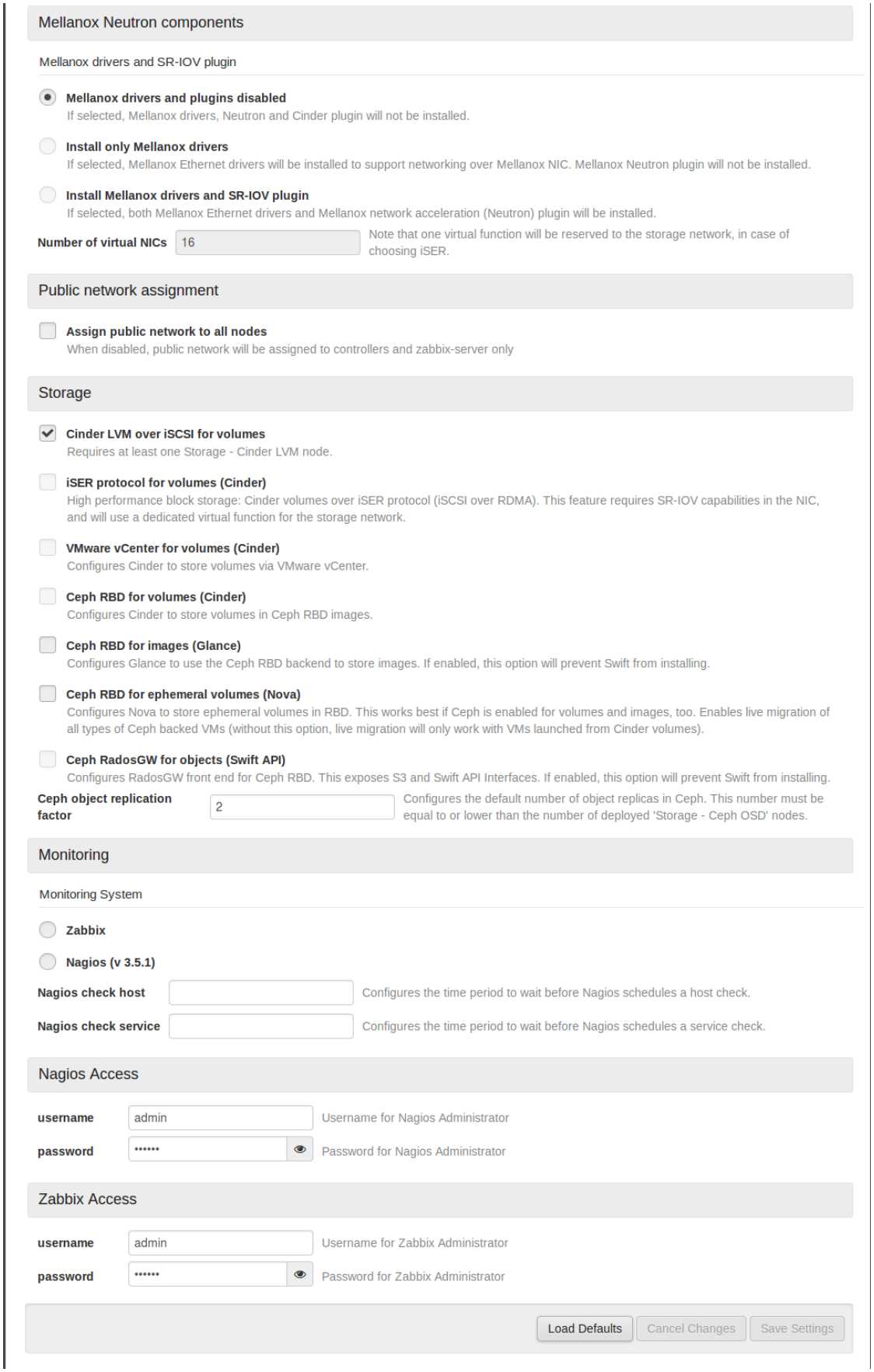

Figure 17: Infrastructure settings (networks, monitoring, and storage)

Logs

The log section is designed to monitor the state of installation and support troubleshooting. The user can select the node to monitor, 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 (Figure 18).

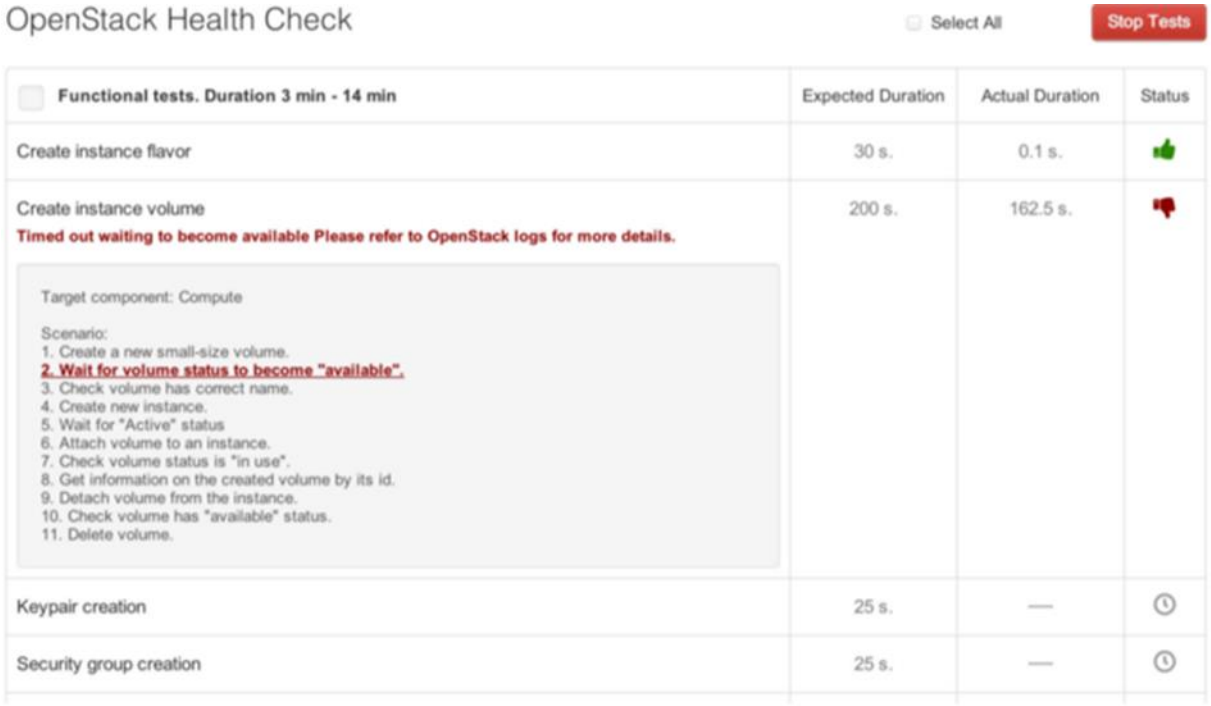

Figure 18: Health check result

Deployment of environment

When the user has finished setting the environment, he can start the deployment process, clicking on "Deploy changes" button (Figure 19). The UI shows the installation progression which can be interrupted by the user in any moment, pushing the button near the progress bar. When the installation is completed, the UI will notify the installation result to the user (Figure 20).

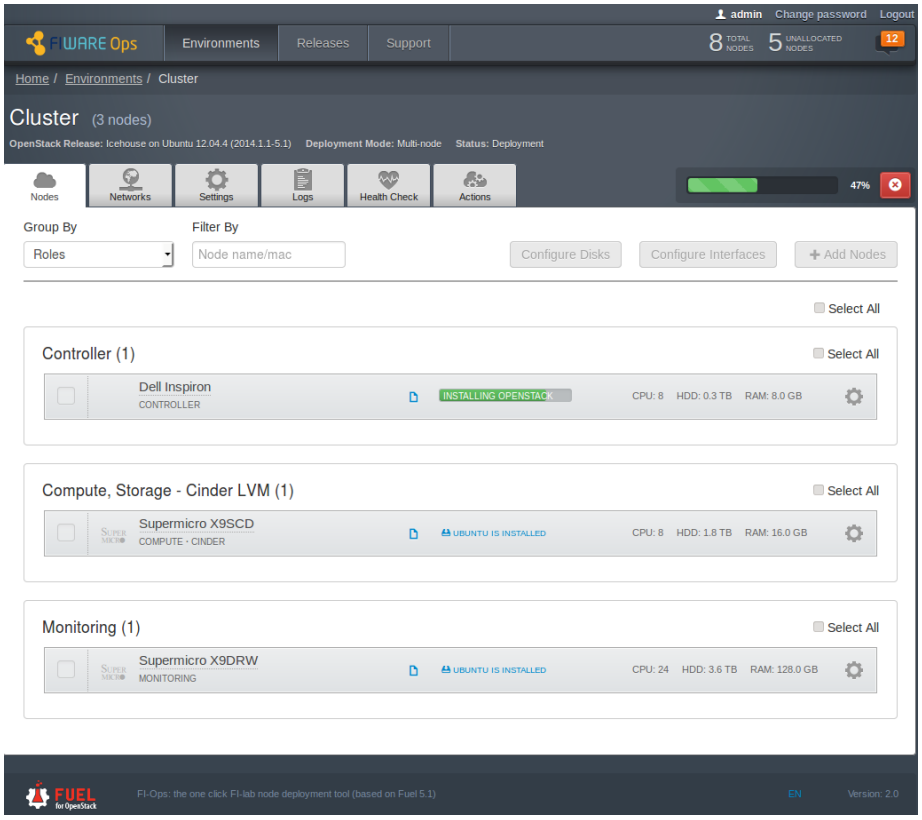

Figure 19: Installation in progress

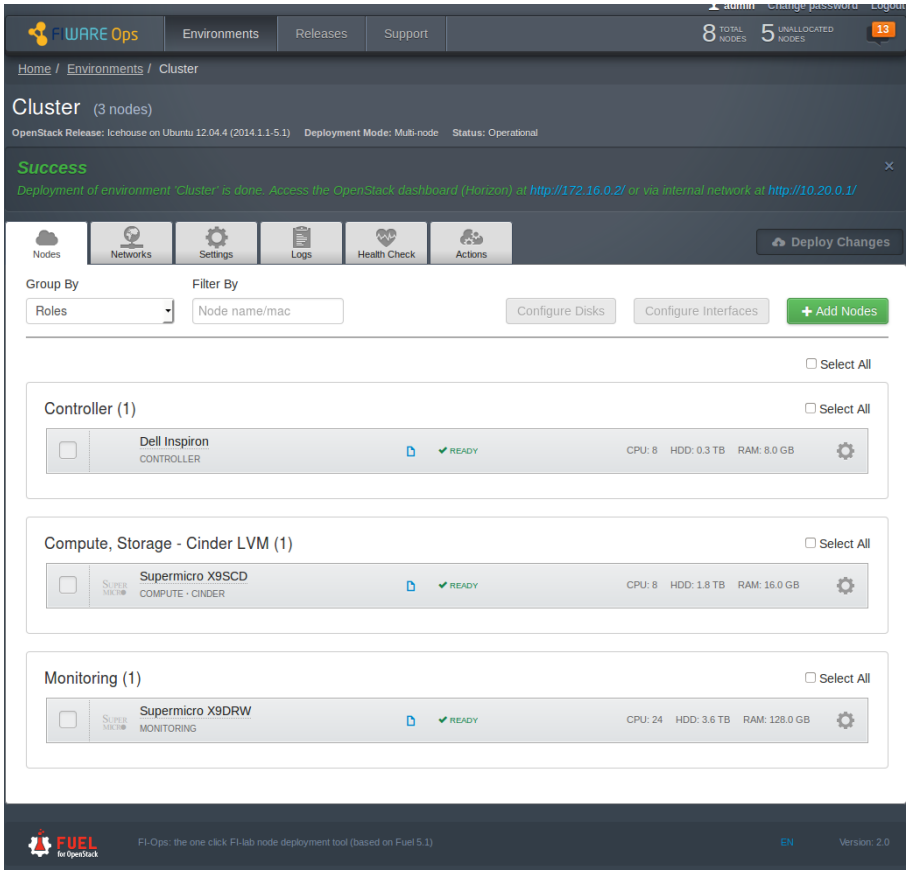

Figure 20: Deployment completed

2.1.11 Developer guide

The ITBox is based on Fuel – Mirantis, OpenStack, released under Apache 2.0 license. The source code of ITBox can be found on https://github.com/SmartInfrastructures/ git repository:

- https://github.com/SmartInfrastructures/fuel-library-dev
- https://github.com/SmartInfrastructures/fuel-web-dev
- https://github.com/SmartInfrastructures/fuel-main-dev

The fuel-main-dev project contains the ISO build scripts so everyone can clone it and build an ITBox ISO. Requirements:

- Ubuntu 12.04 LTS or 14.04 LTS

The steps are:

Install the following required software:

- git clone https://github.com/SmartInfrastructures/itbox-main.git
- make iso

Personalized version of ITBox

A developer might be interested in producing a personalized ITBox version. In order to do so, they should clone (or fork) all repositories (https://github.com/SmartInfrastructures?query=fuel-) and they should know some basic information on the ITBox project structure.

If the developer wants to add a new puppet module, they will make it in a deployment/puppet directory of the project https://github.com/SmartInfrastructures/fuel-library-dev, which contains all puppet scripts.

Finally, the new puppet module will be included in the following files:

- /fuel/deployment/puppet/osnailyfacter/manifests/cluster_simple.pp
- /fuel /deployment/puppet/osnailyfacter/manifests/cluster_ha.pp

For example:

```
include mypuppetscript
```

In order to build a new ISO, the developer will do the steps showed in previous section. Furthermore, they will update _LIB_REPO, NAILGUN_REPO, ASTUTE_REPO and OSTF_REPO fields in /fuel-main/config.mk with developer's repositories URL.

For more information see the Mirantis development documentation: http://docs.mirantis.com/fuel-dev/.

2.2 GE Deployment and Configuration Adapter (DCA)

2.2.1 Summary

The Deployment and Configuration Adapter (DCA) is a 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:

(a) Deployment of multiple GEs upon XIFI infrastructure: The DCA supports the deployment and configuration of multiple GEs in a batch mode (as images, through blueprints or in 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.

(b) Persistence of information related to the deployed GE and SE instances: DCA holds all pertinent information from the whole lifecycle of GE and SE deployment. This includes the requests on behalf of the users (through the cloud portal) and the system responses as related to the GE/SE 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. Moreover, DCA stores the list of GEs/SEs (along with relevant information) provided as Software as a Service by developers and infrastructure owners.

The positioning of DCA in XIFI infrastructure is depicted in Figure 21.

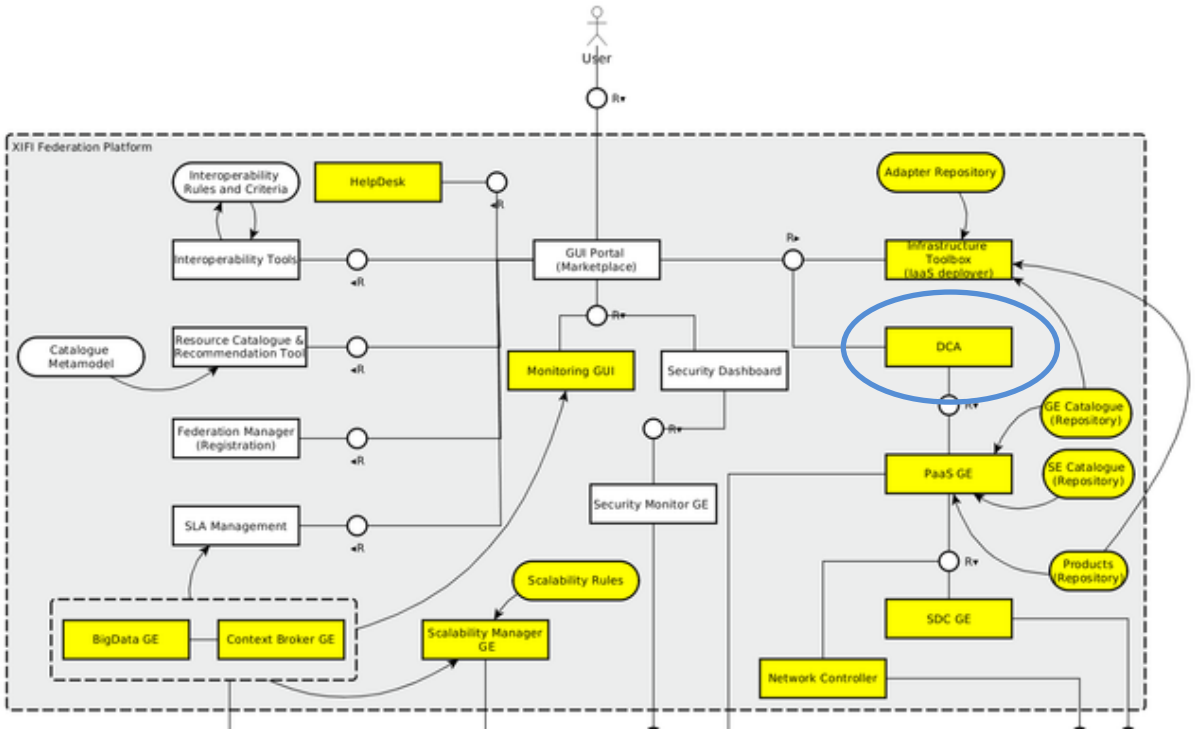

Figure 21: Location of the DCA in the XIFI Reference Architecture

Reference Scenarios	UC-2 - Setup and usage of a development environment. Subscenario: Deployment and Configuration of GE
Reference Stakeholders	**Developers and Infrastructure Owners** who want deploy and configure (individual or groups) GEs upon a node or at a federation level (in a SaaS provision manner) and/or perform queries on the GE instances upon a node or the federation the infrastructure.

Type of ownership	Extension
Original tool	PaaS Manager GE, Cloud Portal
Planned OS license	Apache License Version 2.0
Reference OS community	FIWARE community

Table 6: DCA Context Details

Consist of	DCA is composed of its internal information repository and the set of API Handlers that allows for interconnection with other components: • The internal information repository handles the configuration details for the connection with the nodes and the GEs/SEs repositories. • DCA component is used by the Marketplace, SLA Manager and XIFI Repository. • The API Handlers are responsible for the interconnection with external components.
Depends on	The DCA depends on the following external entities: • The Cloud management platform (DCRM). • The repository of the GEs and SEs. This holds the Generic and Specific Enablers, (a) either as images or (b) as blueprints.

Table 7: DCA Dependencies Summary

2.2.2 Component Responsible

Developer	Contact	Partner
P. Trakadas	ptrak@synelixis.com	SYNELIXIS
A. Papadakis	papadakis@synelixis.com	SYNELIXIS
A. Voulkidis	voulkidis@synelixis.com	SYNELIXIS
P. Karkazis	pkarkazis@synelixis.com	SYNELIXIS
P. Athanasoulis	athanasoulis@synelixis.com	SYNELIXIS

Table 8: DCA Reference Details

2.2.3 Motivation

Since the beginning of the XIFI project, it was identified that the interconnection among the XIFI federated infrastructure, the Use-Case projects and the FIWARE tools should be as smooth as possible, especially in the critical area of GE and SE deployment and configuration. Our starting point has been based on the following assumptions:

- The users (application providers) have to be alleviated from as much overhead as possible when it comes to the deployment of their GEs and SEs, especially considering that deployment and configuration of GEs/SEs in groups is desired by UC projects.
- The federated architecture multiplies possibilities for the users but also poses challenges for the deployment, configuration and management of the GEs/SEs.
- The XIFI project (even from its Description of Work) has identified and described the need for an adaptor in the area of Enablers Deployment and Configuration, that will offer added value services (information) to the upper layer (Cloud Portal, Marketplace, Repository, SLA Manager) of the XIFI infrastructure.

Indeed the Deployment and Configuration Adapter (DCA) has been specified since the beginning of the project (in D3.1 [8]) to fill this space. It also supports mainly the developers and infrastructure owners. The functionality of the DCA is offered to users through the components of WP4, as mentioned above. DCA leverages FIWARE functionality (mainly offered through the cloud hosting GEs) and offers a more flexible and user-friendly approach for special cases of GE/SE deployment, configuration and management. Furthermore it offers a persistency layer for information related to the instances deployed upon the XIFI infrastructure. Finally, DCA maintains the list with the GEs/SEs offered as SaaS through the FIWARE platform.

2.2.4 User Stories backlog

In the following the user stories backlog of the DCA are represented.

Id	User story name	Actors	Description
1	Create VM	User	The user creates a VM in the OpenStack environment. This VM will accommodate the deployed GE(s). The result of the operation is stored locally (in the DCA).
2	Deploy and Configure a Product	User	The user deploys and configures a product (such as Apache Tomcat) in the OpenStack environment. The result of the operation is stored locally (in the DCA).
3	Deploy and Configure a GE using an image	User	The user deploys and configures a GE using an image in the OpenStack environment. The result of the operation is stored locally (in the DCA).
4	Deploy and Configure a group of GEs using images	User	The user deploys and configures GEs sequentially using images in the OpenStack environment. The result of the operation is stored locally (in the DCA).
5	REST interface	User	The information of the deployment and configuration results is offered through the REST interface (1st version of the REST interface).
6	Deploy and Configure a GE using a recipe / blueprint	User	The user deploys and configures a GE using a recipe / blueprint in the OpenStack environment. The result of the operation is stored locally (in the DCA).

Id	User story name	Actors	Description
7	Deploy and Configure a group of GEs using blueprint.	User	The user deploys and configures GEs sequentially using recipes / blueprints in the OpenStack environment. The result of the operation is stored locally (in the DCA).
8	Resource Availability Check on node	User	The user can ask the DCA to check for resource availability for the deployment of a specific GE upon a specific node.
9	Deploy and Configure a GE on multiple nodes	User	The user deploys and configures a GE using an image or a recipe/blueprint on multiple (more or equal to 2) federated XiFi nodes. These nodes have the OpenStack environment installed. The result of the operation is stored locally.
10	GE deployment Registry	User	The DCA holds the deployment and configuration information (successful and unsuccessful requests) in a Registry. The information can be offered through the DCA REST interface (version 2).
11	Query Information per GE and per node	User	The user can ask the DCA for the deployment and configuration requests (both successful and unsuccessful) per GE (or group of GEs) and per node (or group of federated nodes).
12	Complex Queries	User	The user can ask the DCA a set of complex queries, e.g. most frequent (successful, unsuccessful or both) GE deployment and configuration requests in a geographical location, in which nodes is (or is not) the CEP GE installed, which nodes have a combination of GEs, which are the infrastructure owners in a particular node that use a specific GE.
13	Resource Availability Check on federation	User	The user can ask the DCA to check for resource availability for the deployment of a specific GE upon the federation.
14	Full DCA functionality available	User	The full functionality of the DCA is available through the DCA API.

Table 9: DCA User Stories Backlog

2.2.5 State of the Art

Current infrastructures empowered with virtualisation capabilities and comprising multiple nodes call for robust mechanisms for the deployment of the functional elements (specifically the Generic Enablers). In this section, we perform a brief state of the art analysis and explain our selection. We initially refer to an IaaS deployment tool and then to two system configuration tools. We refer to FIWARE pertinent Generic Enabler and then to our point of view.

CloudStack

Apache CloudStack is open source software designed to deploy and manage large networks of virtual machines as an Infrastructure as a Service (IaaS) cloud-computing platform [9]. It includes compute orchestration, Network as a Service, user and account management, resource accounting, an API and UI. It supports VMware, KVM, XenServer and Xen Cloud Platform (XCP) hypervisors, with its API being compatible with AWS EC2 and S3.

Chef

Chef is a system configuration management tool, built on Ruby. It provides system configurations as "recipes" which describe the configuration of a series of resources (products) [10]. It can operate in a client server or standalone fashion. The configuration of Chef is based on Git. It has a stable and well-designed layout and according to [12], it can be natural fit for development-minded admins.

Puppet

Puppet as an open source configuration management tool (with Apache 2.0 license) built on Ruby [6]. Puppet provides similar functionality to Chef. Puppet is a mature solution. Chef and Puppet are the two basic options for software deployment and configuration in FIWARE. According to [12], Puppet is appropriate for heterogeneous environments, with Ansible and Salt [13], [14] a better fit for larger but more homogenous infrastructures.

CloudBees

CloudBees offers a Platform as a Service (PaaS) to build, run, and manage web applications [15]. Its cloud services belong to two main categories: development services and deployment/management services web-based applications. Its orientation towards Java can create restrictions regarding other options.

FIWARE PaaS Manager

According to FIWARE, the PaaS Manager (Platform as a Service Manager) performs the installation and configuration of the whole software stack, taking into account the underlying virtual infrastructure. It enables multiple deployment architectures (based on single or multiple servers). The PaaS Manager wraps the functionality of the SDC (Software Deployment and Configuration) GE that is currently based upon Chef (with plans to support Puppet also).

Our Rationale and Selection

After this brief analysis we consider that the scope of the DCA is not to build from scratch another configuration management tool but to adapt the existing mechanisms to the needs of our users (developers and infrastructure owners and operators) as reflected in relative XIFI requirements. XIFI is tightly bound to FIWARE handling federated, heterogeneous infrastructures. In this view we have opted for full compatibility with the FIWARE and designed DCA as leveraging on PaaS Manager GE functionality with the target to provide additional offerings to the XIFI users (both developers and infrastructure owners).

2.2.6 Architecture Design

As described, DCA provides an adaptation layer between the portal and the components that are responsible for the deployment and configuration of GEs/SEs. In principle, we consider that a GE/SE is deployed on a (single) VM. We use the off-the-shelf (third party) tools that have to be configured prior to the deployment of a GE/SE (such as a DB, web server, etc.). We assume the following:

- The prior deployment and configuration of the cloud hosting GEs (Cloud Portal, PaaS Manager, SDC, DCRM)
- The availability of the recipes/blueprints needed to deploy and configure a GE/SE in an automated manner (as stated in the conditions and instructions of FIWARE)

The interaction of DCA with other XIFI components is depicted in Figure 22:

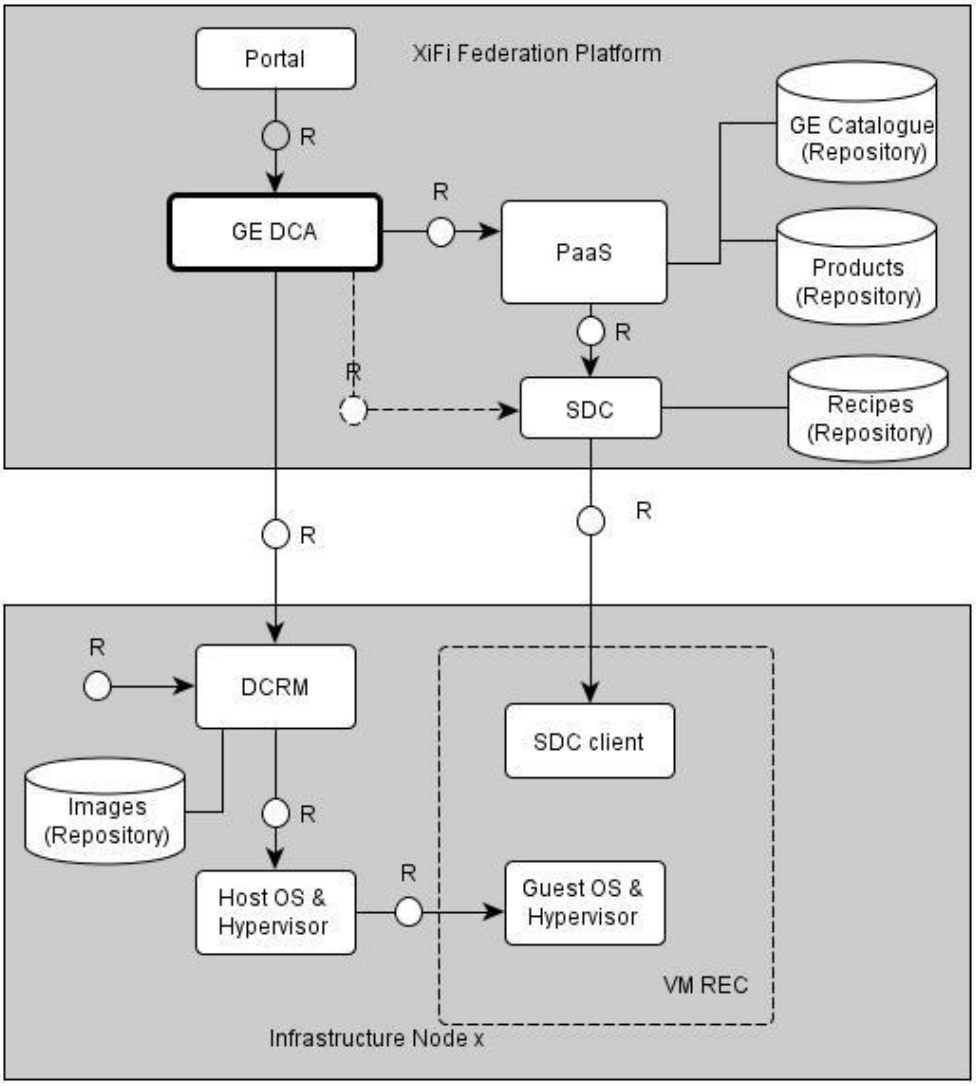

Figure 22: DCA positioning in the XIFI architecture

DCA receives its instructions from the user through the Cloud Portal. These instruction define the (group of) GEs/SEs to be deployed and the region for the deployment. The basis of the GEs/SEs grouping can be thematic (e.g. belonging to the same FIWARE chapter) or of any arbitrary manner according to the application needs (for example an IoT application may need the IoT protocol adapter, the Context Broker and the Big Data Analysis GEs).

DCA also deploys and configures GEs/SEs in a cluster of nodes. Node clustering is at the heart of federation and can be performed based on multiple criteria including the ownership by specific infrastructure owners, authentication / authorisation rights, geographical location, resource availability, properties of the nodes, networking availability existence of GEs/SEs or any other arbitrary reasoning according to the application needs.

DCA is responsible for storing the full set of information related to the exchanged requests and responses and relaying to the user. DCA stores information related to the deployment and/or deletion of GEs/SEs across the FIWARE federation, as well as the list of GEs/SEs offered as SaaS by developers and infrastructure owners.

The internal topology of DCA is depicted in Figure 23:

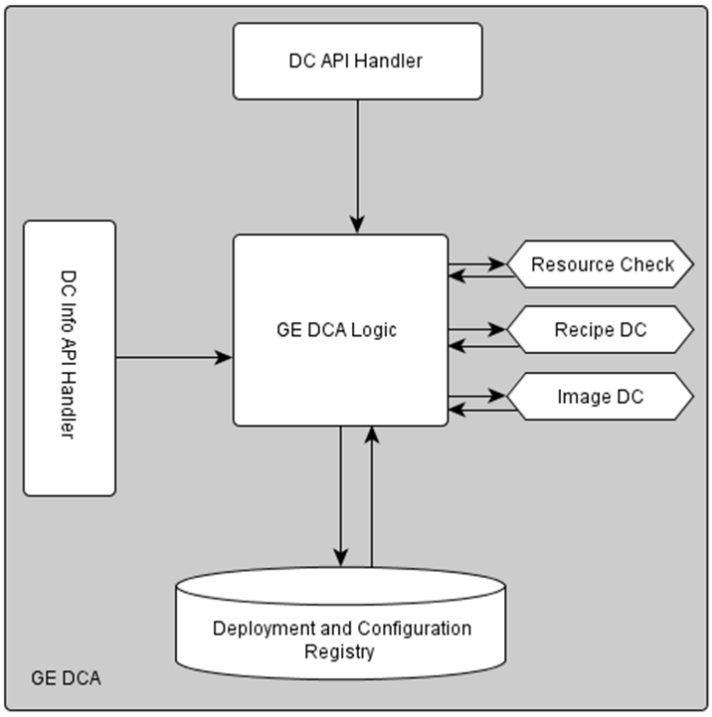

Figure 23: DCA internal topology

The internal architecture consists of the individual functional elements responsible for resource availability checking, and GE/SE deployment and configuration, the API handlers (for deployment and configuration and information retrieval) and the Registry; all of them orchestrated by the internal DCA logic. The Repository holds the deployment and configuration information and exposes a set of queries, indicatively:

- Most popular GE/SE in a geographical location or across federation,
- In which nodes one or more GEs/SEs are installed,
- Number of GEs/SEs deployed or deleted in a given time period,
- Which are the infrastructure owners that use or offer a specific GE/SE,
- Which GEs/SEs are offered as SaaS,
- GEs/SEs deployed by a specific tenant/user.

DCA architecture and internal design provides the following offerings:

- Deploy and configure a GE/SE or combination of GEs/SEs using a blueprint and/or an image at a single node or a cluster of nodes,
- Provide confirmation that the deployment and configuration of a GE/SE has taken place properly,
- Provide information (based on queries) on the GEs/SEs deployment and configuration requests on the VM in the node or node federation.

2.2.7 Release Plan

Version Id	Milestone	User Stories
0.5	30.11.2013	1, 2
1.0	31.12.2013	3, 4, 5
1.5	31.03.2014	6, 7, 8
1.8	30.06.2014	9, 10, 11
2.0	31.12.2014	12, 13, 14

Table 10: Release Plan

2.2.8 Test Cases

This section provides a guide for unit testing regarding the installation and configuration of the DCA and its required software components. This includes:

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

Prerequisites

For the node that will accommodate the DCA the prerequisites are the following:

- 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 persistence server (DCA currently tested with MySQL DB Server)

For the Infrastructure Node:

- The DCRM GE or OpenStack

Test Case for Oracle Java installation

We 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)
```

Test Case for Apache Tomcat installation

We verify that Apache Tomcat has been installed visit from your browser the URL:

```
http://<dca-server-ip>
```

Figure 24 depicts the result of a successful Apache Tomcat installation:

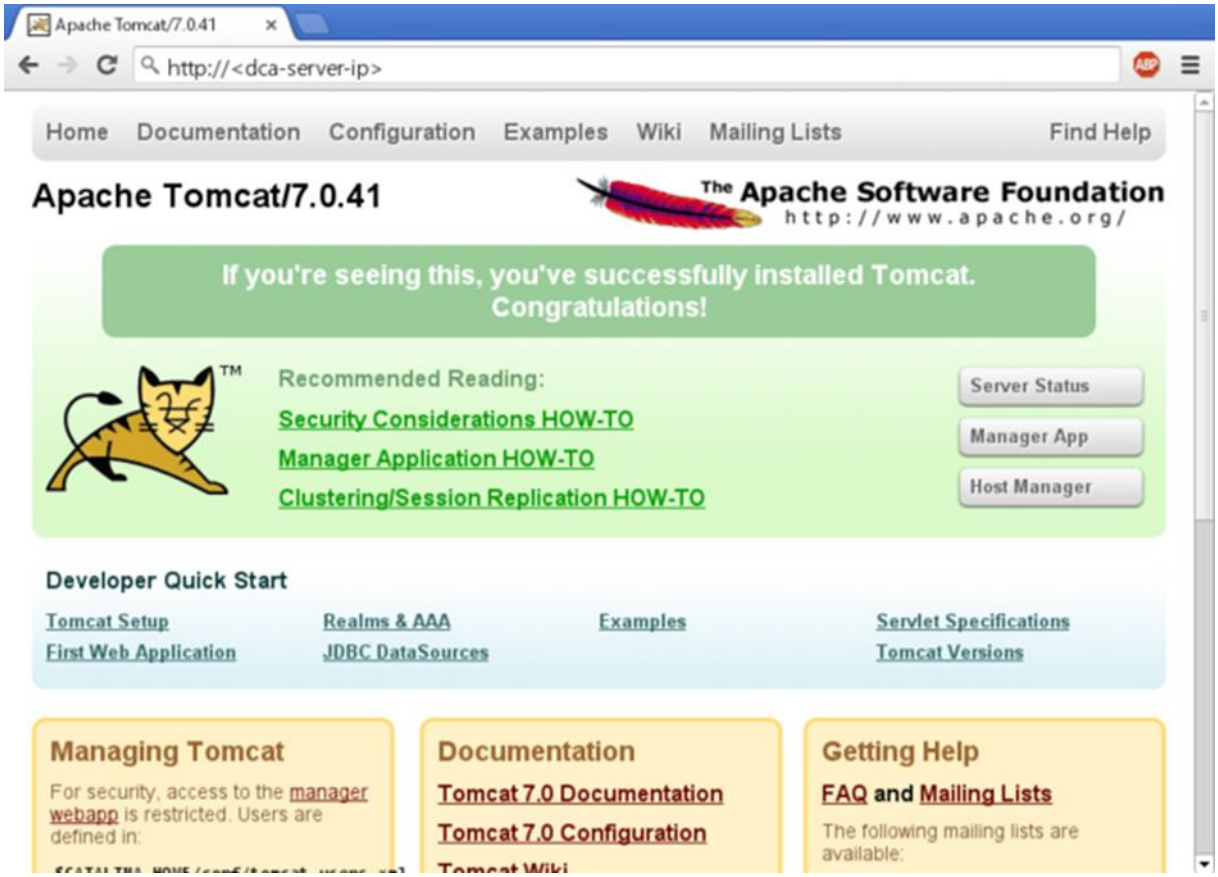

Figure 24: Verification of Apache Tomcat installation

Test case for MySQL and Java JDBC connector installation

We verify that MySQL server and the respective Java JDBC connector have been installed:

```
[root@dca opt]# service mysqld status
mysqld (pid  6557) is running...
[root@dca opt]# mysql_secure_installation
```

To verify that the database creation script has been executed, we execute the following:

```
[root@dca ~]# mysql -uroot -p -e 'show tables' dca
Enter password:
+-----------------+
| Tables_in_dca   |
+-----------------+
| ge              |
| hwrequirements  |
```



```
| keystone        |
| request         |
| response        |
| server          |
+-----------------+
```

We verify that the DCA binaries (dca.war) have been uploaded to the webapps directory of the Apache Tomcat installation and that a keystone instance has been initialized into the DCA platform. Supposing that the DCRM instance of the datacentre 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 should be as follows:

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

Then we verify the smooth operation of DCA invoking its operations as described in the User Manual. For example we list the available flavours in RegionThree as follow:

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

The available flavours are retrieved (tiny, small, medium) as shown in the following response:

```
{
   "flavors":[
      {
         "id":"1",
         "name":"m1.tiny",
         "vcpus":"1",
         "ram":512,
         "disk":"0",
         "swap":"",
         "links":[
            {
               "rel":"self",

"href":"http://cloud.lab.FIWARE.eu:8774/v2/00000000000000000000000000001807/flavors/1",
               "type":null
            },
            {
               "rel":"bookmark",
```

```
"href":"http://cloud.lab.FIWARE.eu:8774/00000000000000000000000000001807/flavors/1",
                "type":null
             }
          ],
          "public":null,
          "OS-FLV-EXT-DATA:ephemeral":0,
          "rxtx_factor":1.0,
          "OS-FLV-DISABLED:disabled":null,
          "rxtx_quota":null,
          "rxtx_cap":null,
          "os-flavor-access:is_public":null
       },
       {
          "id":"2",
          "name":"m1.small",
          "vcpus":"1",
          "ram":2048,
          "disk":"10",
          "swap":"",
          "links":[
             {
                "rel":"self",

"href":"http://cloud.lab.FIWARE.eu:8774/v2/00000000000000000000000000001807/flavors/2",
                "type":null
             },
             {
                "rel":"bookmark",

"href":"http://cloud.lab.FIWARE.eu:8774/00000000000000000000000000001807/flavors/2",
                "type":null
             }
          ],
          "public":null,
          "OS-FLV-EXT-DATA:ephemeral":20,
          "rxtx_factor":1.0,
          "OS-FLV-DISABLED:disabled":null,
          "rxtx_quota":null,
          "rxtx_cap":null,
          "os-flavor-access:is_public":null
       },
       {
          "id":"3",
          "name":"m1.medium",
          "vcpus":"2",
          "ram":4096,
```



```
          "disk":"10",
          "swap":"",
          "links":[
             {
                "rel":"self",

"href":"http://cloud.lab.FIWARE.eu:8774/v2/00000000000000000000000000001807/flavors/3",
                "type":null
             },
             {
                "rel":"bookmark",

"href":"http://cloud.lab.FIWARE.eu:8774/00000000000000000000000000001807/flavors/3",
                "type":null
             }
          ],
          "public":null,
          "OS-FLV-EXT-DATA:ephemeral":40,
          "rxtx_factor":1.0,
          "OS-FLV-DISABLED:disabled":null,
          "rxtx_quota":null,
          "rxtx_cap":null,
          "os-flavor-access:is_public":null
       }
    ]
}
```


2.2.9 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.

Additionally, 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 persistence server (DCA currently tested with MySQL DB Server)

For the Infrastructure Node:

- The DCRM GE or OpenStack Grizzly or OpenStack IceHouse release.

Installation steps

After verifying that the right version of Oracle Java has been installed, we 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
```

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 datacentre 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"}
```


2.2.10 User Manual

The user manual describes the requests and responses of the methods offered by the DCA. A full list of DCA API calls can be seen in http://docs.dca.apiary.io/.

List flavours

We request the flavours available in RegionThree.

Request

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

Response

The available flavours are retrieved (tiny, small, medium).


```
{
   "flavors":[
      {
         "id":"1",
         "name":"m1.tiny",
         "vcpus":"1",
         "ram":512,
         "disk":"0",
         "swap":"",
         "links":[
            {
               "rel":"self",

"href":"http://cloud.lab.FIWARE.eu:8774/v2/00000000000000000000000000001807/flavors/1",
               "type":null
            },
            {
               "rel":"bookmark",

"href":"http://cloud.lab.FIWARE.eu:8774/00000000000000000000000000001807/flavors/1",
               "type":null
            }
         ],
         "public":null,
         "OS-FLV-EXT-DATA:ephemeral":0,
         "rxtx_factor":1.0,
         "OS-FLV-DISABLED:disabled":null,
         "rxtx_quota":null,
         "rxtx_cap":null,
         "os-flavor-access:is_public":null
      },
      {
         "id":"2",
         "name":"m1.small",
```



```
         "vcpus":"1",
         "ram":2048,
         "disk":"10",
         "swap":"",
         "links":[
            {
               "rel":"self",
"href":"http://cloud.lab.FIWARE.eu:8774/v2/00000000000000000000000000001807/flavors/2",
               "type":null
            },
            {
               "rel":"bookmark",
"href":"http://cloud.lab.FIWARE.eu:8774/00000000000000000000000000001807/flavors/2",
               "type":null
            }
         ],
         "public":null,
         "OS-FLV-EXT-DATA:ephemeral":20,
         "rxtx_factor":1.0,
         "OS-FLV-DISABLED:disabled":null,
         "rxtx_quota":null,
         "rxtx_cap":null,
         "os-flavor-access:is_public":null
      },
      {
         "id":"3",
         "name":"m1.medium",
         "vcpus":"2",
         "ram":4096,
         "disk":"10",
         "swap":"",
         "links":[
            {
               "rel":"self",
"href":"http://cloud.lab.FIWARE.eu:8774/v2/00000000000000000000000000001807/flavors/3",
               "type":null
            },
            {
               "rel":"bookmark",
"href":"http://cloud.lab.FIWARE.eu:8774/00000000000000000000000000001807/flavors/3",
               "type":null
            }
         ],
```

```
          "public":null,
          "OS-FLV-EXT-DATA:ephemeral":40,
          "rxtx_factor":1.0,
          "OS-FLV-DISABLED:disabled":null,
          "rxtx_quota":null,
          "rxtx_cap":null,
          "os-flavor-access:is_public":null
       }
   ]
}
```


List the Keypairs

We request the user key pairs on RegionThree.

Request

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

Response

The response is retrieved.


```
{
 "keypairs":[
       {
          "name":"panos",
          "fingerprint":"22:ec:58:0a:c5:f5:c6:d4:7b:91:64:26:5f:17:e9:8b",
          "user_id":null,
          "public_key":"ssh-rsa
AAAAB3NzaC1yc2EAAAADAQABAAAAgQDBgi/eZqP7IKqygkvTn2pkrlPn3LKg57SU8jyRQNxMq37fG6RZUtfTSZsaJs0lnQ
TQvFhfuuzRXs4/9eYQ2CD82BcFOqQr6p2CyhgYzMnEv4xXIz53fUFXGqSjsvUb+YbR6fbSibl3OaXLkNhg4FQH4lGQHDEV
QEABCyagyQTuPQ== nova@gcsic001.ifca.esn",
          "private_key":null
       }
   ]
}
```


List the Security Groups

Request

The security groups available to the user on RegionThree are requested.

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

Response

The security groups are retrieved.


```
{
   "security_groups":[
       {
```



```
        "id":2372,
        "name":"context_broker",
        "description":"context broker security group",
        "rules":[
           {
              "id":4211,
              "name":null,
              "group":{
                 "name":null,
                 "tenant_id":null
              },
              "parent_group_id":2372,
              "from_port":1026,
              "to_port":1026,
              "ip_protocol":"tcp",
              "ip_range":{
                 "cidr":"0.0.0.0/0"
              }
           },
           {
              "id":4212,
              "name":null,
              "group":{
                 "name":null,
                 "tenant_id":null
              },
              "parent_group_id":2372,
              "from_port":8,
              "to_port":0,
              "ip_protocol":"icmp",
              "ip_range":{
                 "cidr":"0.0.0.0/0"
              }
           },
           {
              "id":4213,
              "name":null,
              "group":{
                 "name":null,
                 "tenant_id":null
              },
              "parent_group_id":2372,
              "from_port":0,
              "to_port":0,
              "ip_protocol":"icmp",
```



```
                "ip_range":{
                   "cidr":"0.0.0.0/0"
                }
             }
          ],
          "links":null,
          "tenant_id":"00000000000000000000000000001807"
       },
       {
          "id":2078,
          "name":"default",
          "description":"default",
          "rules":[
             {
                "id":3425,
                "name":null,
                "group":{
                   "name":null,
                   "tenant_id":null
                },
                "parent_group_id":2078,
                "from_port":22,
                "to_port":22,
                "ip_protocol":"tcp",
                "ip_range":{
                   "cidr":"0.0.0.0/0"
                }
             }
          ],
          "links":null,
          "tenant_id":"00000000000000000000000000001807"
       },
       {
          "id":2079,
          "name":"monitoring",
          "description":"For FIWARE monitoring",
          "rules":[
             {
                "id":4137,
                "name":null,
                "group":{
                   "name":null,
                   "tenant_id":null
                },
                "parent_group_id":2079,
```

```
                "from_port":1026,
                "to_port":1026,
                "ip_protocol":"tcp",
                "ip_range":{
                   "cidr":"0.0.0.0/0"
                }
             },
             {
                "id":4141,
                "name":null,
                "group":{
                   "name":null,
                   "tenant_id":null
                },
                "parent_group_id":2079,
                "from_port":80,
                "to_port":80,
                "ip_protocol":"tcp",
                "ip_range":{
                   "cidr":"0.0.0.0/0"
                }
             }
          ],
          "links":null,
          "tenant_id":"00000000000000000000000000001807"
       }
   ]
}
```


List Images

Request

The already available images on RegionThree are requested.

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

Response

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


```
{
   "images":[
      {
         "status":"active",
         "name":"CentOS_6.4",
         "deleted":false,
         "container_format":"bare",
```



```
            "created_at":"2013-11-04T19:13:46",
            "disk_format":"iso",
            "updated_at":"2013-11-04T19:14:01",
            "id":"79e9245c-3a02-48af-8dd2-ada0d893331e",
            "min_disk":0,
            "protected":false,
            "min_ram":0,
            "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":"29c1ee5e-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":"7b001833-5eaa-4a4d-84fa-803e3c59377d",
      "flavorRef":"2",
      "keyName":"xifi",
      "securityGroups":"default,cep",
      "created":1390052553000,
```

```
      "status":null,
      "tenantId":"00000000000000000000000000000158",
      "userId":"artemis-voulkidis",
      "region":"RegionTwo"
   }
]
```


By issuing relevant API calls containing HTTP parameters more complex queries are also supported. 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":"00000000000000000000000000000158",
      "userId":"artemis-voulkidis",
      "region":"RegionTwo"
   }
]
```


Additionally, the DCA component supports 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.FIWARE.eu:8774/v2/00000000000000000000000000000101/servers/a335265d-
777e-42fa-8f5a-355160bc93cc",
                "type":null
             },
             {
                "rel":"bookmark",

"href":"http://cloud.lab.FIWARE.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.FIWARE.eu:8774/00000000000000000000000000000101/images/90d4865d-5e7b-
4d95-af2c-69753e1740d6",
                   "type":null
                }
             ]
          },
          "flavor":{
             "id":"2",
             "links":[
                {
                   "rel":"bookmark",

"href":"http://cloud.lab.FIWARE.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 have instances of CEP and Marketplace GEs 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 GEs 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-4eecb71bd79c",
   "name":null,
   "addresses":null,
   "links":[
      {
         "rel":"self",

"href":"http://cloud.lab.FIWARE.eu:8774/v2/00000000000000000000000000000101/servers/ffae6295-
9aba-48a8-908d-4eecb71bd79c",
         "type":null
      },
      {
         "rel":"bookmark",

"href":"http://cloud.lab.FIWARE.eu:8774/00000000000000000000000000000101/servers/ffae6295-
9aba-48a8-908d-4eecb71bd79c",
         "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"
            }
         ],
         "region":"RegionOne"
      },
      {
         "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/v2/00000000000000000000000000000101/servers/a335265d-
777e-42fa-8f5a-355160bc93cc",
            "type":null
         },
         {
            "rel":"bookmark",

"href":"http://cloud.lab.FIWARE.eu:8774/00000000000000000000000000000101/servers/a335265d-
777e-42fa-8f5a-355160bc93cc",
            "type":null
         }
      ],
      "diskConfig":"MANUAL",
      "adminPass":"ZGs4QxAvZc8c"
   },
   {
      "id":"29c1ee5e-fac8-451d-85d9-963ed7ae2386",
      "name":null,
      "addresses":null,
      "links":[
         {
            "rel":"self",

"href":"http://130.206.80.11:8774/v2/00000000000000000000000000000158/servers/29c1ee5e-fac8-
451d-85d9-963ed7ae2386",
            "type":null
         },
         {
            "rel":"bookmark",

"href":"http://130.206.80.11:8774/00000000000000000000000000000158/servers/29c1ee5e-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 shows 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.

List of popular GEs

Request

We request a list of the most popular GEs in a region.

```
curl http://localhost:8080/dca/servers/ge/popular/region/{region}
```

Response

The response is retrieved and shows the most popular GEs in a region.

```
[
    {
    "name": "Marketplace - SAP RI",
    "nid": "95",
    "deployments": 2
    },
    {
    "name": "Complex Event Processing (CEP) - IBM Proactive Technology Online",
    "nid": "146",
    "deployments": 2
    },
    {
    "name": "Repository - SAP RI",
    "nid": "58",
    "deployments": 1
    },
    {
    "name": "Publish-Subscribe Context Broker - Orion Context Broker",
    "nid": "344",
    "deployments": 1
    }
]
```

Active GEs in a given time period

Request

We request a list of the active GEs in a defined time period.

```
curl
http://localhost:8080/dca/servers/ge/alive/region/{region}/desc/{desc}/time/{time1}/{time2}
```

Response

The response is retrieved and shows the active GEs in that time period.


```
[
    {
    "id": "125bf922-dbb4-4693-9f53-e3c37880d82f",
    "name": "CEP_1",
    "nid": "146",
    "imageRef": "262a9194-f00f-4c0d-8f3e-f331c80070c8",
    "flavorRef": "m1.small (13)",
    "keyName": "user1key",
    "securityGroups": "default",
    "created": 1407231675000,
    "deleted": null,
    "status": "ACTIVE",
    "tenantId": "00000000000000000000000000002782",
    "userId": "user1",
    "region": "region"
    },
    {
    "id": "4bb0ec63-14e3-4448-86a5-62d56b11657f",
    "name": "CEP_2",
    "nid": "146",
    "imageRef": "262a9194-f00f-4c0d-8f3e-f331c80070c8",
    "flavorRef": "m1.small (13)",
    "keyName": "user2key",
    "securityGroups": "default",
    "created": 1408339950000,
    "deleted": null,
    "status": "ACTIVE",
    "tenantId": "00000000000000000000000000000125",
    "userId": "user2",
    "region": "Region"
    }
]
```


List of GEs created during a given time period

Request

We request a list of the GEs deployed in a defined time period.

```
curl
http://localhost:8080/dca/servers/ge/created/alive/region/{region}/desc/{desc}/time/{time1}/{t
ime2}
```

Response

The response is retrieved and shows the GEs deployed in that time period.


```
[
    {
    "id": "125bf922-dbb4-4693-9f53-e3c37880d82f",
    "name": "CEP_1",
    "nid": "146",
    "imageRef": "262a9194-f00f-4c0d-8f3e-f331c80070c8",
    "flavorRef": "m1.small (13)",
    "keyName": "user1key",
    "securityGroups": "default",
    "created": 1407231675000,
    "deleted": null,
    "status": "ACTIVE",
    "tenantId": "00000000000000000000000000002782",
    "userId": "user1",
```

```
    "region": "region"
    },
    {
    "id": "4bb0ec63-14e3-4448-86a5-62d56b11657f",
    "name": "CEP_2",
    "nid": "146",
    "imageRef": "262a9194-f00f-4c0d-8f3e-f331c80070c8",
    "flavorRef": "m1.small (13)",
    "keyName": "user2key",
    "securityGroups": "default",
    "created": 1408339950000,
    "deleted": null,
    "status": "ACTIVE",
    "tenantId": "00000000000000000000000000000125",
    "userId": "user2",
    "region": "Region"
    }
]
```


List of GEs deleted in a given time period

Request

We request a list of the GEs deleted in a defined time period.

```
curl
http://localhost:8080/dca/servers/ge/deleted/region/{region}/desc/{desc}/time/{time1}/{time2}
```

Response

The response is retrieved and shows the GEs deleted in that time period.


```
[
    {
    "id": "125bf922-dbb4-4693-9f53-e3c37880d82f",
    "name": "CEP_1",
    "nid": "146",
    "imageRef": "262a9194-f00f-4c0d-8f3e-f331c80070c8",
    "flavorRef": "m1.small (13)",
    "keyName": "user1key",
    "securityGroups": "default",
    "created": 1407231675000,
    "deleted": null,
    "status": "P_DELETED",
    "tenantId": "00000000000000000000000000002782",
    "userId": "user1",
    "region": "region"
    },
    {
    "id": "4bb0ec63-14e3-4448-86a5-62d56b11657f",
    "name": "CEP_2",
    "nid": "146",
    "imageRef": "262a9194-f00f-4c0d-8f3e-f331c80070c8",
    "flavorRef": "m1.small (13)",
    "keyName": "user2key",
    "securityGroups": "default",
    "created": 1408339950000,
    "deleted": null,
    "status": "P_DELETED",
    "tenantId": "00000000000000000000000000000125",
    "userId": "user2",
    "region": "Region"
    }
]
```


Add new SaaS

Request

In this request we add a new SaaS service in the DCA repository.

```
curl http://localhost:80/dca/saas -s -S --header 'Content-Type: application/json' --header
'Accept: application/json' -d @- | python -mjson.tool) <<EOF
{
"nid":"155",
"uuid":"ilsfgadfdafASDvsvcvvbadfbafdadfbdfbdabdab",
"available":"true",
"name":"service_name",
"description":"service description",
"imgURL":"HTTP://mysite.com/photos/phot1.jpg",
"region":"nodeOne",
"endpoint":"http://121.121.121.121",
"policy_type":"free"
}
EOF
```

Response

The response is retrieved and it shows that the GE/SE has been added as a SaaS.

```
response
{
    "status": "OK"
}
```

List of SaaS

Request

We request a list of the GEs offered as SaaS in all regions.

```
curl http://localhost:80/dca/saas/region/All -s -S --header 'Content-Type: application/json' -
-header 'Accept: application/json' | python -mjson.tool
```

Response

The response is retrieved and shows the GEs offered as SaaS.

```
[
    {
        "available": "true",
        "description": "service description",
        "endpoint": "http://121.121.121.121",
        "imgURL": "HTTP://mysite.com/photos/phot1.jpg",
        "name": "service_name",
        "nid": "155",
        "policy_type": "free",
        "region": "nodeOne",
        "uuid": "ilsfgadfdafASDvsvcvvbadfbafdadfbdfbdabdab"
    }
]
```

Delete a SaaS

Request

We request the deletion of a SaaS.

```
curl http://localhost:80/dca/saas/uuid/ilsfgadfdafASDvsvcvvbadfbafdadfbdfbdabdab/nid/155 -s -S
```

```
--header 'Content-Type: application/json' -X DELETE --header 'Accept: application/json' |
python -mjson.tool
```

Response

The response is retrieved and shows the deletion of the SaaS.

```
{
    "deleted ": "155@ilsfgadfdafASDvsvcvvbadfbafdadfbdfbdabdab"
}
```

2.2.11 Developer Guide

The code of DCA is available in the SVN repository of the project in the WP3, DCA software trunk (https://xifisvn.res.eng.it/wp3/software/DCA).

DCA API is documented here: http://docs.dca.apiary.io/

The project can be built and managed using the Apache Maven software project management tool (http://maven.apache.org/index.html). The steps are:

- The developer retrieves the POM file that describes the build procedure of OpenStack Java SDK and places it in the parent directory of DCA. The POM is retrieved from: https://raw.github.com/woorea/openstack-java-sdk/master/pom.xml
- Then uses maven for building the project.

The commands are the following:

- wget https://raw.github.com/woorea/openstack-java-sdk/master/pom.xml
- cd dca/
- mvn --also-make --projects dca install

The dca.war file is then created in the target directory.

3 CONCLUSIONS

This document described the second and final version of the design and development of tools and adapters to facilitate the management of the XIFI federation and FIWARE enablers.

On one hand, the first component, Infrastructure ToolBox (ITBox), provides a toolbox that eases and automatizes the process of deploying the appropriate tools and GEs to a bare-metal infrastructure, becoming fully compliant to the XIFI federation requirements. On the other hand, the second component, DCA, provides a management layer that caters for the deployment of multiple GEs to be utilised by application developers, offering collected data to the federation and thus the interested users.

These components have been developed based on the tools which have been described in the sections of this deliverable and offered as autonomous components to interested XIFI partners, infrastructure owners and application developers through FIWARE Ops. Apart from the source code, the components described herein provide extended documentation for the installation, testing and extension of provided APIs in the XIFI repository.

As a conclusion, the achievements of this deliverable can be summarised as follows:

- ITBox significantly assists XIFI sustainability by offering a toolbox that takes care of the processes required for a new (bare-metal) infrastructure owner to become IaaS/PaaS compatible with the XIFI federation infrastructure.
- DCA supports the extensibility of XIFI infrastructure and simplify the management of GEs deployment and configuration by providing APIs and data on federation level that is of high interest of application developers.

The final documentation of ITBox and DCA can be found at:

- http://wiki.fi-xifi.eu/Xifi:Wp3:Components:InfrastructureToolbox
- http://wiki.fi-xifi.eu/Xifi:Wp3:Components:DCA.

It is noted that if extensions or modifications to the ITBox and DCA components are deemed necessary, e.g. support FI-PPP Accelerator Projects, until the end of XIFI project, these changes will be reflected to the aforementioned wiki-pages of the components.

REFERENCES

[1] XIFI Deliverable D3.3, “Infrastructures Management Toolkit API”, http://wiki.fi-xifi.eu/Main_Page#Technical_Deliverables

[2] Software Deployment and Configuration (SDC) GE - https://forge.FIWARE.eu/plugins/mediawiki/wiki/fiware/index.php/FIWARE.OpenSpecification.Cloud.SDC

[3] Fuel - Mirantis OpenStack by Mirantis - http://fuel.mirantis.com/

[4] OpenCrowbar, https://github.com/opencrowbar/core/

[5] Rackspace Private Cloud - http://www.rackspace.com/cloud/private/script/

[6] Puppet Labs - http://puppetlabs.com/puppet/what-is-puppet

[7] Fuel - Mirantis OpenStack command line - http://docs.mirantis.com/Fuel - Mirantis OpenStack /Fuel - Mirantis OpenStack -3.2.1/user-guide.html#understanding-environment-deployment-with-Fuel - Mirantis OpenStack -cli

[8] XIFI Deliverable D3.1, "XIFI infrastructure adaptation components API open specification", http://wiki.fi-XIFI.eu/Public:D3.1

[9] Apache Cloudstack, http://cloudstack.apache.org/

[10] Chef configuration management tool, http://www.getchef.com/chef/

[11] Crowbar Project: http://crowbar.github.io/home.html

[12] P. Venezia "Review: Puppet vs. Chef vs. Ansible vs. Salt", http://www.infoworld.com/d/data-center/review-puppet-vs-chef-vs-ansible-vs-salt-231308

[13] Ansible configuration management tool, http://www.ansibleworks.com/

[14] Salt Open Source Software Project, http://www.saltstack.com/

[15] CloudBees Platform as a Service, http://www.cloudbees.com

