

D6.1.1

Version	1.0
Author	USV
Dissemination	PU
Date	27/01/2014
Status	Final



D6.1.1: NUBOMEDIA Testbed and simulated load validation v1

Project acronym:	NUBOMEDIA
Project title:	NUBOMEDIA: an elastic Platform as a Service (PaaS) cloud for interactive social multimedia
Project duration:	2014-02-01 to 2016-09-30
Project type:	STREP
Project reference:	610576
Project web page:	http://www.nubimedia.eu
Work package	WP6: Demonstrator
WP leader	Constantin Filote (USV)
Deliverable nature:	Demonstrator
Lead editor:	Alin Calinciu (USV)
Planned delivery date	01/2015
Actual delivery date	27/01/2015
Keywords	Hardware infrastructure, testbed, Gitlab, Jenkins, continuous integration, OpenStack Horizon

The research leading to these results has been funded by the European Union's Seventh Framework Programme (FP7/2007-2013) under grant agreement n° 610576



FP7 ICT-2013.1.6. Connected and Social Media



This is a public deliverable that is provided to the community under a **Creative Commons Attribution-ShareAlike 4.0 International License**
<http://creativecommons.org/licenses/by-sa/4.0/>

You are free to:

Share — copy and redistribute the material in any medium or format

Adapt — remix, transform, and build upon the material
 for any purpose, even commercially.

The licensor cannot revoke these freedoms as long as you follow the license terms.

Under the following terms:

Attribution — You must give appropriate credit, provide a link to the license, and indicate if changes were made. You may do so in any reasonable manner, but not in any way that suggests the licensor endorses you or your use.

ShareAlike — If you remix, transform, or build upon the material, you must distribute your contributions under the same license as the original.

No additional restrictions — You may not apply legal terms or technological measures that legally restrict others from doing anything the license permits.

Notices:

You do not have to comply with the license for elements of the material in the public domain or where your use is permitted by an applicable exception or limitation.

No warranties are given. The license may not give you all of the permissions necessary for your intended use. For example, other rights such as publicity, privacy, or moral rights may limit how you use the material.

For a full description of the license legal terms, please refer to:
<http://creativecommons.org/licenses/by-sa/4.0/legalcode>

Contributors:

Alin Calinciu (USV)
Cristian Spoiala (USV)
Constantin Filote (USV)

Internal Reviewer(s):

Javier Lopez Fernandez (Naevatec)
Luis Lopez (URJC)

Version History

Version	Date	Authors	Comments
0.1	16/07/2014	Alin Calinciuc (USV) Cristian Spoiala (USV) Constantin Filote (USV)	Initial version
0.2	31/07/2014	Alin Calinciuc (USV) Cristian Spoiala (USV) Constantin Filote (USV)	Version bump
0.3	26/09/2014	Constantin Filote (USV) Alin Calinciuc (USV)	Format updates and CI section
0.4	05/11/2014	Cristian Spoiala	Updated version history, license
0.5	15/12/2014	Cristian Spoiala	Final version. Ready for internal review
1.0	07/01/2014	Cristian Spoiala	Added more details about Jenkins jobs, testplan.

Table of contents

1 Executive summary.....	8
2 Hardware infrastructure	8
2.1 Network connectivity	9
2.2 Compute machines	9
2.3 Management machine	10
2.4 Storage system.....	10
2.5 Auxiliary Units	10
2.5.1 <i>UPS</i>	10
2.5.2 <i>Cooling</i>	10
3 Setup testbed	10
3.1 Setup OpenStack with RDO	10
3.1.1 <i>Software prerequisites:</i>	11
3.1.2 <i>Hardware prerequisites:</i>	11
3.1.3 <i>Operating system preparation</i>	11
3.1.4 <i>Install with Packstack</i>	12
4 Software infrastructure	13
4.1 Gitlab.....	13
4.1.1 <i>Features</i>	13
4.1.2 <i>Dashboard</i>	13
4.2 Jenkins.....	14
4.2.1 <i>Features</i>	14
5 Continuous Integration	14
5.1 Jenkins	15
5.2 Packer	15
5.3 Access CI tools.....	15
6 NUBOMEDIA tests	16
6.1 Testing plan.....	16
6.2 Template 1.....	16
6.2.1 <i>Test creation of Template 1</i>	17
6.2.2 <i>Test functionality of Template 1</i>	17
7 Access to the testbed.....	18
7.1 OpenStack Horizon (Web Interface)	18
7.1.1 <i>Access Horizon</i>	18
7.1.2 <i>Create an instance</i>	19
7.1.3 <i>Associate a floating IP</i>	20
7.1.4 <i>Connect to your instance</i>	21
7.1.5 <i>Delete an instance</i>	21
7.2 OpenStack API.....	21
8 References.....	22

List of Figures:

Figure 1 USV Cluster for Testbed.....	9
Figure 2 Gitlab Dashboard	13
Figure 3 NUBOMEDIA Jenkins Dashboard.....	15
Figure 4 Template 1 Topology	16
Figure 5 Architecture of the WebRTC Loopback Test	18
Figure 6 Horizon VM Management	19
Figure 7 Horizon VM Management Launch Instance	20
Figure 8 Horizon VM Management Allocate IP	21
Figure 9 Horizon VM Management Confirm Associate IP	21

Acronyms and abbreviations:

IaaS	Infrastructure as a Service
CI	Continuous Integration
ECC	Error-Correcting Code
SAS	Serial Attached SCSI
UPS	Uninterruptible Power Supply
NTP	Network Time Protocol
API	Application Programming Interface
PESQ	Perceptual Evaluation of Speech Quality

1 Executive summary

This document provides an overview of the NUBOMEDIA Testbed components. Will focus on describing the hardware and software infrastructure. Also will cover the CI (Continuous Integration) and testing planning of the NUBOMEDIA testbed.

2 Hardware infrastructure

This section will describe the current hardware infrastructure for OpenStack IaaS [1] at Stefan cel Mare University of Suceava (USV). This facility will be used for NUBOMEDIA testbed.

Hardware infrastructure is built around multiple units of IBM BladeCenter H on 2 x 42U racks.



Figure 1 USV Cluster for Testbed

2.1 Network connectivity

Internal connectivity inside a BladeCenter is achieved using a two Layer 2 IBM Connectivity Modules with 802.11Q VLAN tagging, and having six external 1GbE ports.

The connectivity with storage, external network and Internet is made using a Layer 2 Managed switch SMC8126L2 with 802.11Q VLAN tagging.

2.2 Compute machines

Each node has the following hardware configuration:

- CPU: 2 x AMD Opteron Quad Core 2376 2.3GHz,
- RAM: 16GB PC2-5300 CL5 ECC DDR2 667MHz.
- LOCAL DISK: IBM 147GB SAS 10K HDD.

- Connectivity: 2 x Gigabit Ethernet cards.

NUBOMEDIA R3 release for production is composed from 5 machines: 1 controller and compute node and 4 compute nodes.

Development instance is composed of 1 controller and 1 compute for R3.

2.3 Management machine

Management machine for NUBOMEDIA that will host Jenkins CI tool, Git code repository with Gitlab:

- CPU: 2 x Intel Xeon Quad Core E5345 2.33GHz.
- RAM: 4GB ECC.
- Local Disk: 2 x IBM 73.4GB SAS HDD.
- Connectivity: 2 x Gigabit Ethernet cards.

2.4 Storage system

Data from OpenStack will be stored on an IBM DS4700 Express Model 70 Storage. Capacity available is 2TB.

2.5 Auxiliary Units

2.5.1 UPS

Each rack has 2 x IBM APC UPS7500XHV UPS.

2.5.2 Cooling

For cooling servers room has 3 x enterprise AC unit 20000 BTU.

3 Setup testbed

This section will focus on allowing partners to replicate the testbed on their premises.

3.1 Setup OpenStack with RDO

To install OpenStack are multiple scenarios and methods available. In this section we'll describe the best method to replicate the testbed.

Deploying RDO is an easy process, setting up an OpenStack cloud takes approximately 15 minutes. It can be as short as 3 steps if you want to deploy it on a single server, but if you want to deploy it to add more nodes it can take more time.

RDO is maintained by RedHat, and it is more suitable for production environments.

The deployment script by RDO is licensed under the [Creative Commons Attribution-ShareAlike 3.0 Unported license](#).

3.1.1 Software prerequisites:

For installing OpenStack RDO, you will need a RHEL-based Linux distribution, such as CentOS, Scientific Linux, or Fedora 20 or later.

3.1.2 Hardware prerequisites:

It is recommended a machine with at least 2GB of RAM, and hardware virtualization extension with at least 1 network adapter for single node deployment. For multi-node deployment, at least two network adapters are needed.

For multi-node deployment, you will also need a Layer 2 Switch that supports 802.11Q VLANs (VLAN tagging).

3.1.3 Operating system preparation

- You will first need to add RDO repositories:

```
yum install -y http://rdo.fedorapeople.org/rdo-release.rpm
```

- You will need to update your current packages using:

```
yum update -y
```

- Then you need to enable ssh key login:

```
cd ~
mkdir .ssh
chmod 700 .ssh
cd .ssh
nano -w authorized_keys # here you should add your
public key
chmod 600 authorized_keys
restorecon -R -v /root/.ssh
```

- Disable selinux or set it in permissive mode (if there is a reason not to have it in enforcing mode).
In file: /etc/selinux/config edit:

```
SELINUX=permissive
```

After this if you do not want to reboot the system you should:

```
setenforce 0
```

If you have previously disabled SELinux, you will need to re-label the filesystem, since when SELinux is disabled, this does not happen for new files, and failing to relabel will likely cause many false positive issues. The easiest way to do that is to do the following as root:

```
touch /.autorelabel
reboot
```

- After this you should install NTP client on all servers because all servers should have date in sync with each other:

```
yum install ntp -y
chkconfig ntpd on
ntpdate pool.ntp.org
/etc/init.d/ntpd start
```

3.1.4 Install with Packstack

- You should first generate the configuration file for the deployment with the following command:

```
packstack --gen-answer-file=icehouse_deployment_vlan.cfg
```

- After this, you should configure the file accordingly with your hardware configuration and also configure the deployment location for every service, if you use multi-node deployment.

```
nano icehouse_deployment_vlan.cfg
```

- Next, you should run packstack to deploy OpenStack RDO to all instances configured:

```
packstack --answer-file=icehouse_deployment_vlan.cfg
```

During this process, you will be required to type the root password for all node that you use in your deployment in order for OpenStack to be able to add its public key to each one of them.

Once the process is complete, you can log in to the OpenStack web interface "Horizon" by going to [http://\\$YOURIP/dashboard](http://$YOURIP/dashboard).

The username is "admin". The password can be found in the file `keystonerc_admin` in the `/root/` directory of the control node.

4 Software infrastructure

This section will describe software solutions used for working on the testbed.

4.1 Gitlab

GitLab Community Edition is an open source software for developers to collaborate on code. It allows creation of projects, repositories, access, and code reviews.

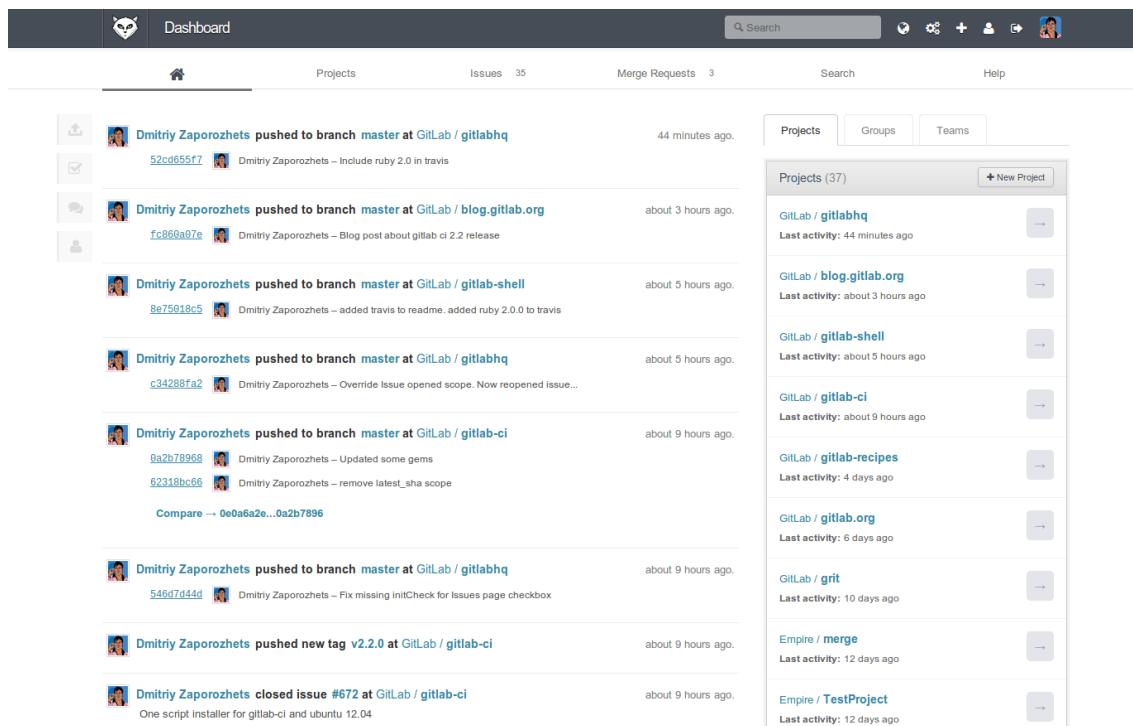
GitLab CE is distributed under MIT license.

4.1.1 Features

- Keep your code secure on your own server
- Manage repositories, users and access permissions
- Perform code review with merge requests
- Extended permission system with 5 access levels and branch protection
- Efficient user management by creating groups of projects and teams of users
- Use the ticketing system included in GitLab or integrate your existing system
- Perform code reviews with merge requests
- Line comments and discussions in merge requests and diffs
- Each project has a wiki backed up by a separate git repository
- Integrations with external systems like JIRA, Redmine, Slack

4.1.2 Dashboard

GitLab CE has a web dashboard for developers to collaborate together.



The screenshot shows the GitLab dashboard interface. At the top, there's a navigation bar with links for 'Dashboard', 'Projects', 'Issues 35', 'Merge Requests 3', 'Search', 'Help', and user profile icons. Below the navigation is a search bar and a toolbar with various icons. The main area is divided into two sections: a 'Recent Activity' feed on the left and a 'Projects' list on the right.

Recent Activity:

- Dmitry Zaporozhets pushed to branch master at GitLab / gitlabhq 44 minutes ago. (Commit 52cd655f7, message: Dmitry Zaporozhets – Include ruby 2.0 in travis)
- Dmitry Zaporozhets pushed to branch master at GitLab / blog.gitlab.org about 3 hours ago. (Commit fc860a07e, message: Dmitry Zaporozhets – Blog post about gitlab ci 2.2 release)
- Dmitry Zaporozhets pushed to branch master at GitLab / gitlab-shell about 5 hours ago. (Commit 8e75018c5, message: Dmitry Zaporozhets – added travis to readme, added ruby 2.0 to travis)
- Dmitry Zaporozhets pushed to branch master at GitLab / gitlabhq about 5 hours ago. (Commit c34288fa2, message: Dmitry Zaporozhets – Override Issue opened scope. Now reopened issue...)
- Dmitry Zaporozhets pushed to branch master at GitLab / gitlab-ci about 9 hours ago. (Commit 0a2b78968, message: Dmitry Zaporozhets – Updated some gems; Commit 62318bc66, message: Dmitry Zaporozhets – remove latest_sha scope)
- Dmitry Zaporozhets pushed to branch master at GitLab / gitlabhq about 9 hours ago. (Commit 546d7d44d, message: Dmitry Zaporozhets – Fix missing initCheck for Issues page checkbox)
- Dmitry Zaporozhets pushed new tag v2.2.0 at GitLab / gitlab-ci about 9 hours ago.
- Dmitry Zaporozhets closed issue #672 at GitLab / gitlab-ci about 9 hours ago. (Message: One script installer for gitlab-ci and ubuntu 12.04)

Projects:

- Projects (37)
 - GitLab / gitlabhq (Last activity: 44 minutes ago)
 - GitLab / blog.gitlab.org (Last activity: about 3 hours ago)
 - GitLab / gitlab-shell (Last activity: about 5 hours ago)
 - GitLab / gitlab-ci (Last activity: about 9 hours ago)
 - GitLab / gitlab-recipes (Last activity: 4 days ago)
 - GitLab / gitlab.org (Last activity: 6 days ago)
 - GitLab / grit (Last activity: 10 days ago)
 - Empire / merge (Last activity: 12 days ago)
 - Empire / TestProject (Last activity: 12 days ago)

Figure 2 Gitlab Dashboard

4.2 Jenkins

Jenkins is a Continuous Integration server that monitors executions of repeated jobs, such as building a software project. In a nutshell, Jenkins provides an easy-to-use system making easy for developers to integrate changes to the project and making it easier for users to obtain a fresh build.

Jenkins was initially a fork from Hudson after disagreements with Oracle who controls Hudson.

Jenkins is distributed under a MIT license.

4.2.1 Features

Jenkins offers the following features:

- Easy installation: Just java -jar jenkins.war, or deploy it in a servlet container. No additional install, no database.
- Easy configuration: Jenkins can be configured entirely from its friendly web GUI with extensive on-the-fly error checks and inline help. There's no need to tweak XML manually anymore, although if you'd like to do so, you can do that, too.
- Change set support: Jenkins can generate a list of changes made into the build from Subversion/CVS. This is also done in a fairly efficient fashion, to reduce the load on the repository.
- Permanent links: Jenkins gives you clean readable URLs for most of its pages, including some permalinks like "latest build"/"latest successful build", so that they can be easily linked from elsewhere.
- RSS/E-mail/IM Integration: Monitor build results by RSS or e-mail to get real-time notifications on failures.
- After-the-fact tagging: Builds can be tagged long after builds are completed.
- JUnit/TestNG test reporting: JUnit test reports can be tabulated, summarized, and displayed with history information, such as when it started breaking, etc. History trend is plotted into a graph.
- Distributed builds: Jenkins can distribute build/test loads to multiple computers. This lets you get the most out of those idle workstations sitting beneath developers' desks.
- File fingerprinting: Jenkins can keep track of which build produced which jars, and which build is using which version of jars, and so on. This works even for jars that are produced outside Jenkins, and is ideal for projects to track dependency.
- Plugin Support: Jenkins can be extended via 3rd party plugins. You can write plugins to make Jenkins support tools/processes that your team uses.

5 Continuous Integration

5.1 Jenkins

In NUBOMEDIA, we use Jenkins for continuous integration. Jenkins is an open source continuous integration tool written in Java.

On Jenkins, we use a plugin named Docker plugin, that aims to provide Jenkins capability to use a Docker host to dynamically provision a slave, run a single build, then tear-down that slave. We configured a Jenkins slave node that hosts all Docker containers, and we created separate jobs to do nightly build images with Docker for each running environment needed in the CI system. When these jobs are done, fresh images are uploaded to Jenkins Docker machine, and new slave nodes with labels are added to the Jenkins master. The advantage of using this architecture is that Jenkins can run jobs on fresh and isolated Docker containers without installing any packages or changing configurations on a live Jenkins node.

5.2 Packer

Another tool that we use for CI is Packer, which creates the virtual machine images from a specific configuration.

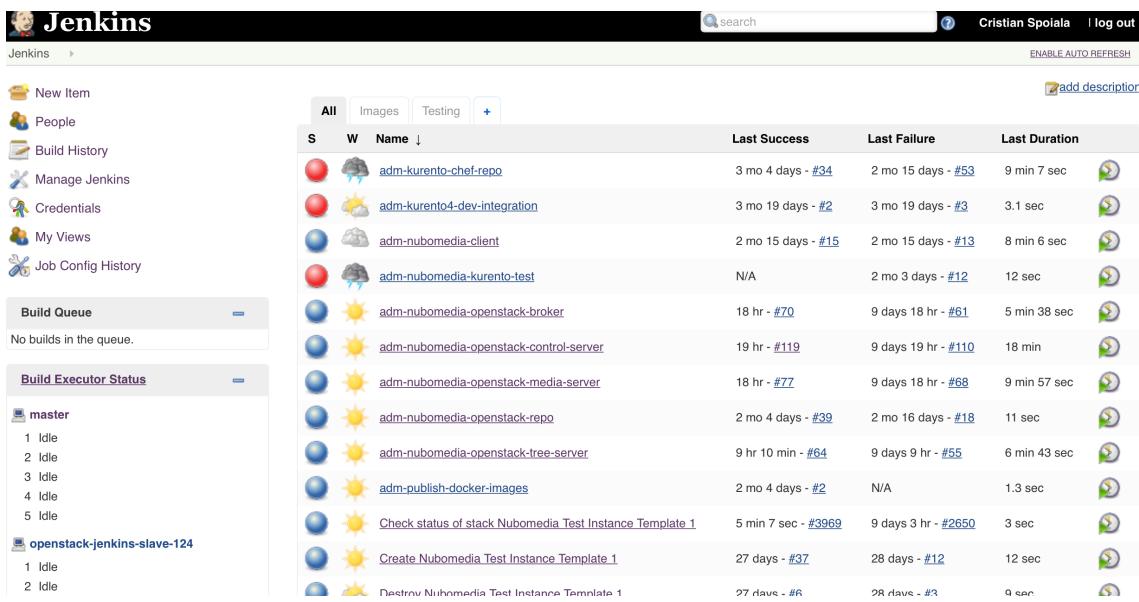
We've setup a Jenkins job to run every night and clone the latest configurations for the OpenStack images of NUBOMEDIA (kurento-broker, kurento-connector, and kurento-media-server), then build a fresh image with latest packages for each of them, and upload the newly created images to Glance, and after that delete the old images.

Source code of packer scripts are on NUBOMEDIA git repository:

- <http://git.nubomedia.eu/usv/adm-nubomedia-openstack-repo/tree/master>

Jenkins jobs configurations are found on:

- http://git.nubomedia.eu/usv/ci/tree/master/jenkins_jobs/images



The screenshot shows the Jenkins dashboard with the following details:

S	W	Name	Last Success	Last Failure	Last Duration
●	rainy	adm-kurento-chef-repo	3 mo 4 days - #34	2 mo 15 days - #53	9 min 7 sec
●	sunny	adm-kurento4-dev-integration	3 mo 19 days - #2	3 mo 19 days - #3	3.1 sec
●	rainy	adm-nubomedia-client	2 mo 15 days - #15	2 mo 15 days - #13	8 min 6 sec
●	rainy	adm-nubomedia-kurento-test	N/A	2 mo 3 days - #12	12 sec
●	sunny	adm-nubomedia-openstack-broker	18 hr - #70	9 days 18 hr - #61	5 min 38 sec
●	sunny	adm-nubomedia-openstack-control-server	19 hr - #119	9 days 19 hr - #110	18 min
●	sunny	adm-nubomedia-openstack-media-server	18 hr - #77	9 days 18 hr - #68	9 min 57 sec
●	sunny	adm-nubomedia-openstack-repo	2 mo 4 days - #39	2 mo 16 days - #18	11 sec
●	sunny	adm-nubomedia-openstack-tree-server	9 hr 10 min - #64	9 days 9 hr - #55	6 min 43 sec
●	sunny	adm-publish-docker-images	2 mo 4 days - #2	N/A	1.3 sec
●	sunny	Check status of stack Nubomedia Test Instance Template 1	5 min 7 sec - #3969	9 days 3 hr - #2650	3 sec
●	sunny	Create Nubomedia Test Instance Template 1	27 days - #37	28 days - #12	12 sec
●	rainy	Destroy Nubomedia Test Instance Template 1	27 days - #8	28 days - #3	9 sec

On the left sidebar, there are links for New Item, People, Build History, Manage Jenkins, Credentials, My Views, and Job Config History. Under Build Queue, it says "No builds in the queue." Under Build Executor Status, it lists "master" with 5 idle executors and "openstack-jenkins-slave-124" with 2 idle executors.

Figure 3 NUBOMEDIA Jenkins Dashboard

5.3 Access CI tools

The NUBOMEDIA implementation of Jenkins can be found at

- <http://jenkins.nubomedia.eu>

All the installations scripts for CI and all configuration files used on the testbed (Openstack deployment) are stored on NUBOMEDIA shared git repository on Gitlab:

- <http://git.nubomedia.eu>

When partners need access to NUBOMEDIA Jenkins or Git repository, they should ask access from filote@eed.usv.ro

6 NUBOMEDIA tests

6.1 Testing plan

Using Jenkins as the main CI tool, was configured a series of tests for the software artefacts developed for NUBOMEDIA project.

Objective of the testing was to make sure that during the development of NUBOMEDIA software quality is kept at high levels. Beside software development tests, the tests are covering also the infrastructure and functionality of OpenStack.

In order to integrate NUBOMEDIA software artefacts on Testbed, was planned a series of templates that included incremental releases. Releases were built on virtual machine images that were deployed on the NUBOMEDIA.

Images are created automatically every day by a Jenkins job using Packer tool mentioned on Section 5.

After the release of the software artefacts on templates, were developed functionality tests that ensure that the release didn't break expected functionality.

6.2 Template 1

Template 1 is the first template to include the first software artefacts developed for NUBOMEDIA. It is consisted from a Control Server, Broker and multiple Media Servers.

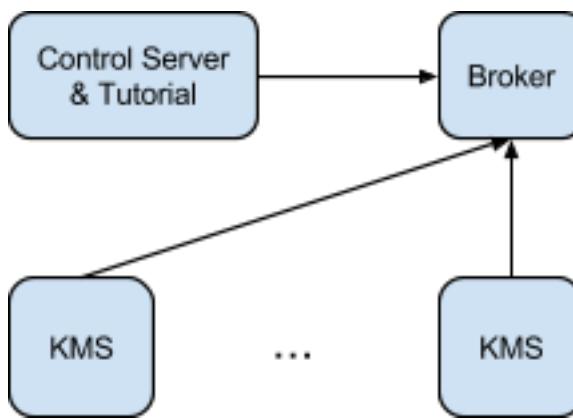


Figure 4 Template 1 Topology

To test the Template 1 functionality were created multiple tests. Tests are running from NUBOMEDIA Jenkins installation.

6.2.1 Test creation of Template 1

First test was for the creation of the Template 1 with a HEAT template developed by TUB.

The will start 3 virtual servers on NUBOMEDIA and configures them according to Template 1 topology.

To work with the HEAT Template, the Jenkins job is installing first the OpenStack API HEAT Client and configures the credentials to connect to the OpenStack API.

Having the HEAT client installed job is able now to create the instance using TUB template fetched from the git repository:

- http://git.nubomedia.eu/giuseppe.a.carella/elastic-media-manager/raw/master/templates/template_1/nubomedia.yaml

To create the instance was used the command:

```
heat stack-create jenkins
```

Jenkins job configuration can be found on the NUBOMEDIA git repository:

- http://git.nubomedia.eu/usv/ci/blob/master/jenkins_jobs/create_nubomedia_template1_instance.xml

6.2.2 Test functionality of Template 1

Second test validates the functionality of a Template 1 instance created by previous Jenkins job.

In order to simulate a browser, the job is using:

- Selenium Chrome Driver [2]
- Chrome Browser
- Xvfb Display Server [3]

Job is downloading and installs a binary of Chrome browser for Linux from:

- https://dl.google.com/linux/direct/google-chrome-stable_current_amd64.deb

Other dependencies of the job are:

- Maven for build management
- Git for source control
- Java Development Kit (JDK)

NAEVATEC partner developed the Java integration tests and we fetch the software artefacts from:

- <https://github.com/Kurento/kurento-java.git>

The following tests are performed:

- WebRtcLoopbackTest - This is a functional test for NUBOMEDIA instance. Media pipeline from Template 1 instance is composed from a single media element (WebRtcEndpoint) and after is received on the server is sent back to the client.

The test will pass if the video stream is received back, play time is as expected and color in the video is as expected.

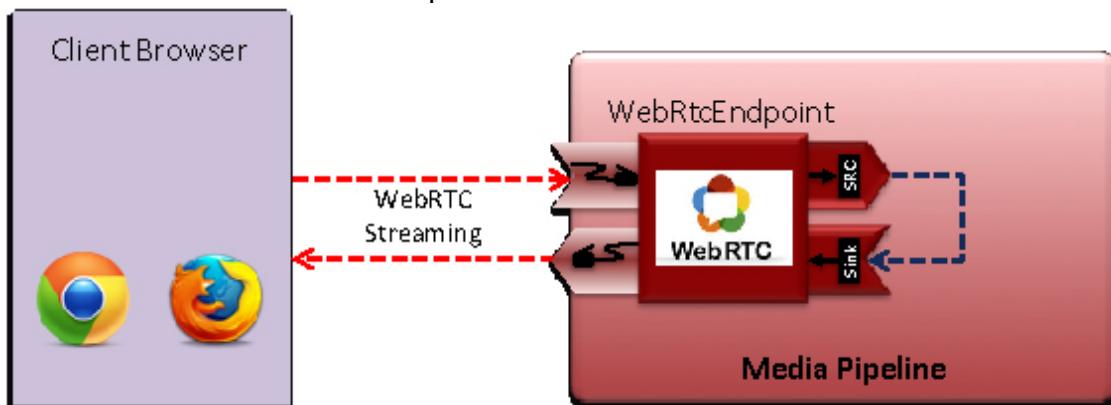


Figure 5 Architecture of the WebRTC Loopback Test

- WebRtcQualityLoopbackTest – Similar with WebRTC Loopback Test, this test is additionally checking the audio quality with PESQ.

The job is running with 4 Jenkins slaves.

Jenkins Job configuration can be found on NUBOMEDIA git repository:

- http://git.nubomedia.eu/usv/ci/blob/master/jenkins_jobs/test_template1_instance.xml

7 Access to the testbed

This section will detail how partners will access testbed instances.

7.1 OpenStack Horizon (Web Interface)

7.1.1 Access Horizon

You need to go to the login link: <http://devconsole.nubomedia.eu> and input the following:

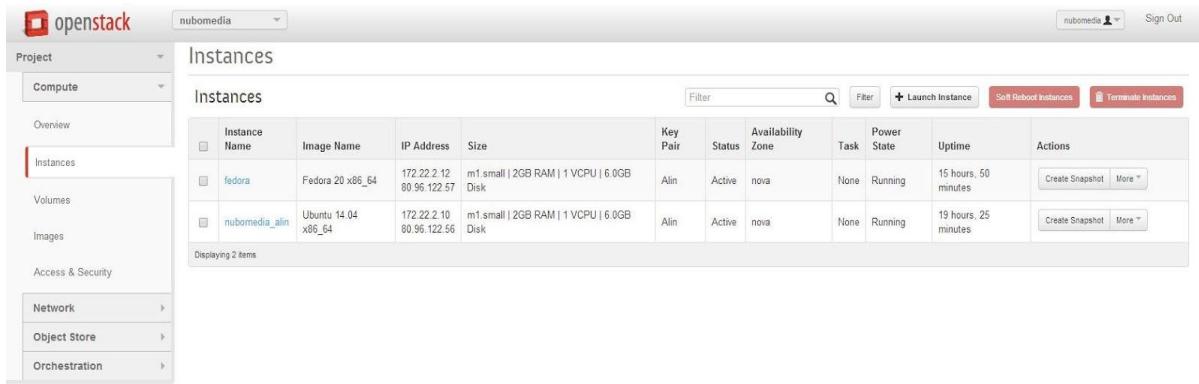
User Name: *user*

Password: *password*

Credentials to access the testbed are found on the wiki:

- https://www.nubomedia.eu/redmine/projects/nubomedia/wiki/Accessing_NUBO MEDIA_testbed

After you are logged in you will see all running instances.



The screenshot shows the OpenStack Horizon interface for managing instances. The left sidebar is titled 'openstack' and includes sections for Project (Compute, Overview, Instances, Volumes, Images), Network, Object Store, and Orchestration. The main area is titled 'Instances' and displays a table of running instances. The table columns are: Instance Name, Image Name, IP Address, Size, Key Pair, Status, Availability Zone, Task, Power State, Uptime, and Actions. Two instances are listed: 'fedora' (Fedora 20 x86_64) and 'nubomedia_aliin' (Ubuntu 14.04 x86_64). Both instances are running, with 'fedora' having been up for 15 hours and 50 minutes, and 'nubomedia_aliin' for 19 hours and 25 minutes.

Instance Name	Image Name	IP Address	Size	Key Pair	Status	Availability Zone	Task	Power State	Uptime	Actions
fedora	Fedora 20 x86_64	172.22.2.12 80.96.122.57	m1.small 2GB RAM 1 VCPU 6.0GB Disk	Aliin	Active	nova	None	Running	15 hours, 50 minutes	Create Snapshot More
nubomedia_aliin	Ubuntu 14.04 x86_64	172.22.2.10 80.96.122.56	m1.small 2GB RAM 1 VCPU 6.0GB Disk	Aliin	Active	nova	None	Running	19 hours, 25 minutes	Create Snapshot More

Figure 6 Horizon VM Management

7.1.2 Create an instance

When you are logged in, you can Launch Instance from the upper right.

7.1.2.1 On the first tab you will need to input the:

- **Instance Name**, which can be whatever name you want
- **Flavor**. A flavor is a virtual hardware template that is defining the size for RAM, disk, number of cores, and so on. After you will define a flavor in the right side you can see the Flavor Details.
- **Instance Count**, the number of instances that will be launched using this template.
- **Instance Boot Source**. Here you can choose to boot from image, boot from a volume or boot from a snapshot. If you want to start a new clean instance you should chose *Boot from image*.
- **Image name**. At this point you choose the Linux distribution you want to run. Currently there are two types: Fedora 20 x86_64 and Ubuntu 14.04 x86_64.

7.1.2.2 On the second tab:

You will need to define your public Key Pair that you will use when you connect to the instance, and the Security Group. For default, the security group will permit connections on all ports of the instance.

When all these fields are completed, you can start to click on Lunch in the right down of the popup.

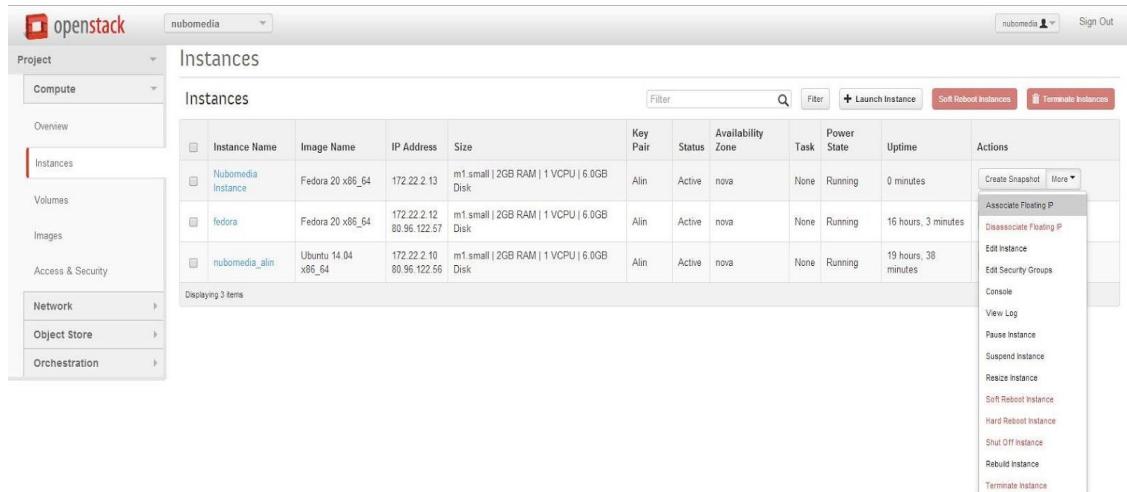
Launch Instance

Details *	Access & Security *	Networking *	Post-Creation	Advanced Options												
Availability Zone nova	Specify the details for launching an instance.															
Instance Name * nubimedia	The chart below shows the resources used by this project in relation to the project's quotas.															
Flavor * m1.small	Flavor Details <table border="1"> <tr><td>Name</td><td>m1.small</td></tr> <tr><td>VCPUs</td><td>1</td></tr> <tr><td>Root Disk</td><td>6 GB</td></tr> <tr><td>Ephemeral Disk</td><td>0 GB</td></tr> <tr><td>Total Disk</td><td>6 GB</td></tr> <tr><td>RAM</td><td>2,048 MB</td></tr> </table>				Name	m1.small	VCPUs	1	Root Disk	6 GB	Ephemeral Disk	0 GB	Total Disk	6 GB	RAM	2,048 MB
Name	m1.small															
VCPUs	1															
Root Disk	6 GB															
Ephemeral Disk	0 GB															
Total Disk	6 GB															
RAM	2,048 MB															
Some flavors not meeting minimum image requirements have been disabled.																
Instance Count * 1	Project Limits <table border="1"> <tr><td>Number of Instances</td><td>3 of 10 Used</td></tr> <tr><td>Number of VCPUs</td><td>3 of 20 Used</td></tr> <tr><td>Total RAM</td><td>6,144 of 51,200 MB Used</td></tr> </table>				Number of Instances	3 of 10 Used	Number of VCPUs	3 of 20 Used	Total RAM	6,144 of 51,200 MB Used						
Number of Instances	3 of 10 Used															
Number of VCPUs	3 of 20 Used															
Total RAM	6,144 of 51,200 MB Used															
Instance Boot Source * Boot from image																
Image Name Fedora 20 x86_64 (201.1 MB)																
<input type="button" value="Cancel"/> <input type="button" value="Launch"/>																

Figure 7 Horizon VM Management Launch Instance

7.1.3 Associate a floating IP

After the instance is in Power State Running, you can associate to it a Floating IP. This means you can add to it a public IP address, because instances initially have only OpenStack internal IP addresses.



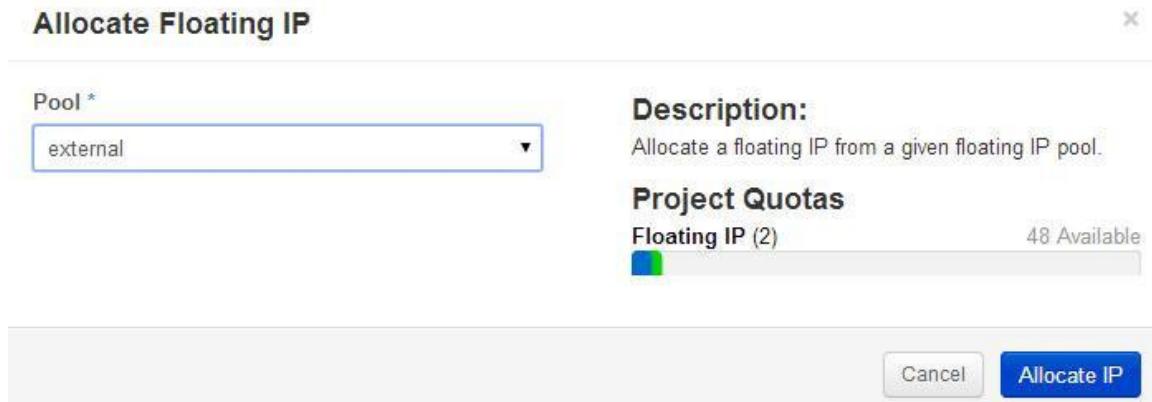
The screenshot shows the Horizon VM Management interface. On the left, there's a navigation sidebar with 'Compute' selected under 'Project'. The main area displays a table of instances:

Instance Name	Image Name	IP Address	Size	Key Pair	Status	Availability Zone	Task	Power State	Uptime	Actions
Nubimedia Instance	Fedora 20 x86_64	172.22.2.13	m1.small 2GB RAM 1 VCPU 6.0GB Disk	Alin	Active	nova	None	Running	0 minutes	<input type="button" value="Create Snapshot"/> <input type="button" value="More"/>
fedora	Fedora 20 x86_64	172.22.2.12	m1.small 2GB RAM 1 VCPU 6.0GB Disk	Alin	Active	nova	None	Running	16 hours, 3 minutes	<input type="button" value="Associate Floating IP"/> <input type="button" value="Disassociate Floating IP"/>
nubimedia_ahn	Ubuntu 14.04 x86_64	172.22.2.10	m1.small 2GB RAM 1 VCPU 6.0GB Disk	Alin	Active	nova	None	Running	19 hours, 38 minutes	<input type="button" value="Edit Instance"/> <input type="button" value="Edit Security Groups"/> <input type="button" value="Console"/> <input type="button" value="View Log"/> <input type="button" value="Pause Instance"/> <input type="button" value="Suspend Instance"/> <input type="button" value="Resize Instance"/> <input type="button" value="Soft Reboot Instance"/> <input type="button" value="Hard Reboot Instance"/> <input type="button" value="Shut Off Instance"/> <input type="button" value="Rebuild Instance"/> <input type="button" value="Terminate Instance"/>

At the bottom of the table, it says 'Displaying 3 items'.

Figure 7 Horizon VM Management Associate IP

Then if you see that No IP addresses available you should click on + and then from the Pool dropdown you should choose external and then Allocate IP.



Allocate Floating IP

Pool *
external

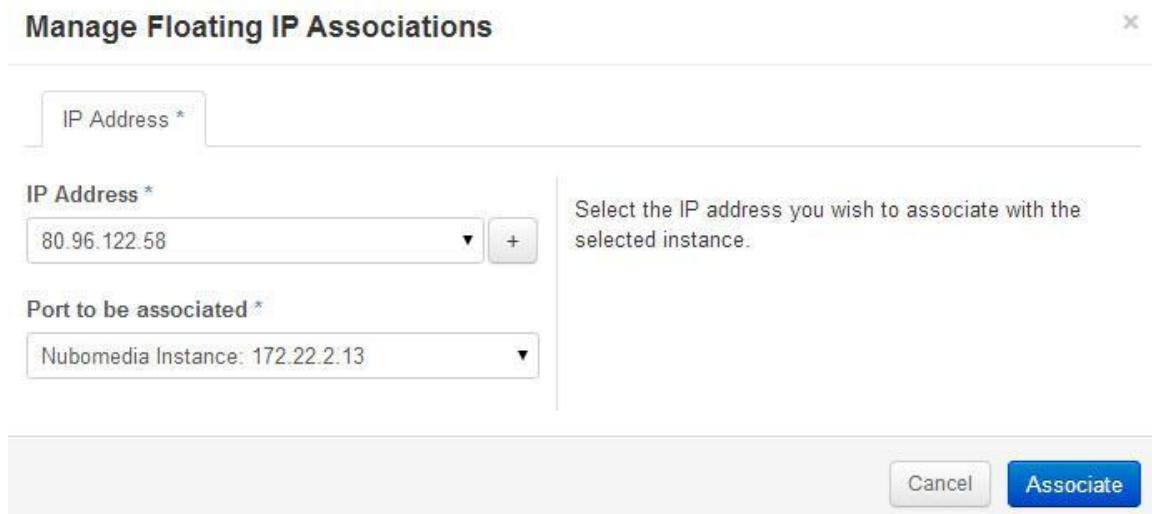
Description:
Allocate a floating IP from a given floating IP pool.

Project Quotas
Floating IP (2) 48 Available

Cancel **Allocate IP**

Figure 8 Horizon VM Management Allocate IP

After the new IP address is allocated you the click on Associate to associate it to your instance.



Manage Floating IP Associations

IP Address *

IP Address *
80.96.122.58

Port to be associated *
Nubomedia Instance: 172.22.2.13

Select the IP address you wish to associate with the selected instance.

Cancel **Associate**

Figure 9 Horizon VM Management Confirm Associate IP

7.1.4 Connect to your instance

Now you can **connect using SSH** and the *private key* associated with the public key you've added to the public IP address associated to your instance.

7.1.5 Delete an instance

If you want to **delete a machine or hard reboot** it, you can use the dropdown More on the Actions tab.

7.2 OpenStack API

To interact with testbed can be used OpenStack API available in WP3. This method of connecting is described in detailed in the WP3 Virtual Infrastructures document.

8 References

- [1] Openstack: Open source software for creating public and private clouds. See <http://www.openstack.org/>.
- [2] SeleniumChromeDriver <https://code.google.com/p/selenium/wiki/ChromeDriver>
- [3] Xvfb is a Display Server that operates in memory without showing any output: <http://www.x.org/releases/X11R7.6/doc/man/man1/Xvfb.1.xhtml>