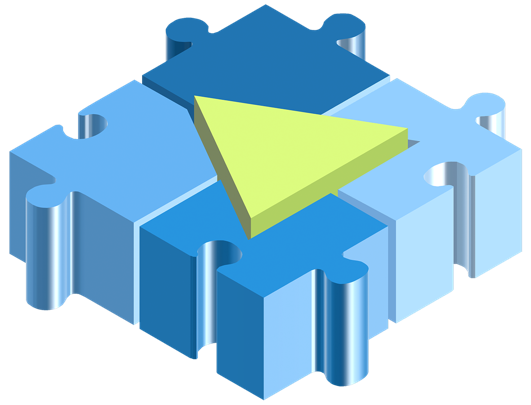
****

Openstack Installation

Basic Tests

Indice degli Argomenti

1. Verify the Identity Service (Keystone) 3

2. Verify the Image Service (Glance) 9

3. Verify the Compute Service (Nova) 10

4. Verify the Block Storage Service (Cinder) 16

5. Verify the Networking Service (Neutron) 21

6. Verify the services on the Compute Nodes 24

7. End-to-End test: VM instantiation 26

# Verify the Identity Service (Keystone)

The following steps allow to verify the correct installation of the Openstack Identity Service:

1. if set, clear OS\_SERVICE\_TOKEN and OS\_SERVICE\_ENDPOINT environment variables:

$ unset OS\_SERVICE\_TOKEN OS\_SERVICE\_ENDPOINT

1. Request an authentication token by using the admin user and the password you chose for that user:

$ keystone --os-username=admin --os-password=$ADMIN\_PASS --os-auth-url=http://<controller\_ip>:35357/v2.0 token-get

In response, you should receive a token paired with your **user ID**. This verifies that the Identity Service is running on the expected endpoint and that your user account is established with the expected credentials.

The expected output is shown hereafter:

+----------+------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------+

| Property | Value |

+----------+------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------+

| expires | 2014-11-11T17:12:03Z |

| **id** | -BVX5OwflHp+AYa3ROfl6T9XxWgesKL7DcMmDem1Uq5w9OR3682m-n4NYAcewkiDPPoj+NRkEmpRZdQVDm9vtitp-RPMcySXW7KSiWzSOD1AbqK7Ug5mh1PSSGAb7LRg8fQ-w8+ZQMnVpks+uqObNc7MuFqhlIK1PTe1oFLS2YwCp4BtrbWD8xu0dVRchiLpFzcFwBVNnXuC47KA== |

| **user\_id** | 769b7fafd6c145ccb8680e398702f14f |  
+----------+------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------+

1. Verify that authorization behaves as expected. To do so, request authorization on a tenant:

$ keystone --os-username=admin --os-password=$ADMIN\_PASS --os-tenant-name=admin --os-auth-url=http:// <controller\_ip>:35357/v2.0 token-get

In response, you should receive a token that includes the ID of the tenant that you specified. This verifies that your user account has an explicitly defined role on the specified tenant and the tenant exists as expected.

The expected output is shown hereafter:

+----------+------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------+

| Property | Value |

+----------+------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------+

| expires | 2014-11-18T03:04:05Z |

| **id** |  |

| **tenant\_id** | afb4978796f5422d9acdec64da6aaf5f |

| **user\_id** | 13b300f990ec4826a23cd252089e6cad |

+----------+------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------+

You can also set your --os-\* variables in your environment to simplify command-line usage.

For the following tests, create an admin-openrc.sh file in your home directory, with the following content:

export OS\_USERNAME=admin

export OS\_PASSWORD=$ADMIN\_PASS

export OS\_TENANT\_NAME=admin

export OS\_AUTH\_URL=http://<controller\_ip>:35357/v2.0

# Verify the Image Service (Glance)

*Preparation*:

Set your --os-\* variables in your environment:

$ source admin-openrc.sh

The following steps allow to verify the correct installation of the Openstack Image Service:

1. Verify that the Glance API is correctly configured and working:

$ glance image-list

If you have not uploaded any image yet, the output of this command will show an empty table; otherwise you will get the list of registered images.

1. Verify that the Glance registry process is working correctly by importing the Cirros image available online:

$ wget http://cdn.download.cirros-cloud.net/0.3.2/cirros-0.3.2-x86\_64-disk.img

$ glance image-create --name "cirros-0.3.2-x86\_64" --disk-format qcow2 --container-format bare --is-public True --progress < cirros-0.3.2-x86\_64-disk.img

1. Check the status of the image using the ID returned by the previous command:

$ glance image-show <image\_id>

# Verify the Compute Service (Nova)

*Preparation*:

Set your --os-\* variables in your environment:

$ source admin-openrc.sh

The following steps allow to verify the correct installation of the Openstack Compute Service:

1. Verify that all nova processes are up and running:

$ for s in /etc/init.d/nova-\*; do status $(basename $s); done

The expected output is like the following:

nova-api start/running, process 26601

nova-cert start/running, process 26615

nova-conductor start/running, process 26625

nova-consoleauth start/running, process 26659

nova-novncproxy start/running, process 26669

nova-scheduler start/running, process 26643

If one or more of the above processes are stopped, look at the log files in the folder /var/log/nova in order to find the problem.

1. To verify your configuration, list available images:

$ nova image-list

The output should be like this:

+--------------------------------------+---------------------+--------+--------+

| ID | Name | Status | Server |   
+--------------------------------------+---------------------+--------+--------+  
 | acafc7c0-40aa-4026-9673-b879898e1fc2 | cirros-0.3.2-x86\_64 | ACTIVE | |   
+--------------------------------------+---------------------+--------+--------+

1. Verify the process “nova-cert” is running and connected to the database and messaging server (AMQP):
   1. Verify the connection with the database (replace the DB\_PORT value with the correct value depending on your installation. Default is 3306)

# export DB\_PORT=3306

# netstat -punt | grep -s $DB\_PORT | grep -s $(pgrep nova-cert)

The expected output shows the ESTABILISHED connection:

tcp 0 0 90.147.75.205:39263 212.189.205.99:33306 **ESTABLISHED** 23311/python

* 1. Verify the connection with the messaging server (replace the AMQP\_PORT value with the correct value depending on your installation. Default is 5672)

# export AMQP\_PORT=5672

# netstat -punt | grep -s $AMQP\_PORT | grep -s $(pgrep nova-cert)

The expected output shows the ESTABILISHED connection:

tcp 0 0 90.147.75.205:52498 90.147.75.68:5672 **ESTABLISHED** 23311/python

1. Verify the process “nova-scheduler” is running and connected to the database and messaging server (AMQP):
   1. Verify the connection with the database (replace the DB\_PORT value with the correct value depending on your installation. Default is 3306)

# export DB\_PORT=3306

# netstat -punt | grep -s $DB\_PORT | grep -s $(pgrep nova-scheduler)

The expected output shows the ESTABILISHED connections:

tcp 0 0 90.147.75.205:39303 212.189.205.99:33306 **ESTABLISHED** 23943/python

tcp 0 0 90.147.75.205:39355 212.189.205.99:33306 **ESTABLISHED** 23943/python

* 1. Verify the connection with the messaging server (replace the AMQP\_PORT value with the correct value depending on your installation. Default is 5672)

# export AMQP\_PORT=5672

# netstat -punt | grep -s $AMQP\_PORT | grep -s $(pgrep nova-scheduler)

The expected output shows the ESTABILISHED connections:

tcp 0 0 90.147.75.205:52528 90.147.75.68:5672 **ESTABLISHED** 23943/python

tcp 0 0 90.147.75.205:53220 90.147.75.68:5672 **ESTABLISHED** 23943/python

1. Verify the process “nova-consoleauth” is running and connected to the database and messaging server (AMQP):
   1. Verify the connection with the database (replace the DB\_PORT value with the correct value depending on your installation. Default is 3306)

# export DB\_PORT=3306

# netstat -punt | grep -s $DB\_PORT | grep -s $(pgrep -f /usr/bin/nova-consoleauth)

The expected output shows the ESTABILISHED connections:

tcp 0 0 90.147.75.205:39257 212.189.205.99:33306 **ESTABLISHED** 23532/python

* 1. Verify the connection with the messaging server (replace the AMQP\_PORT value with the correct value depending on your installation. Default is 5672)

# export AMQP\_PORT=5672

# netstat -punt | grep -s $AMQP\_PORT | grep -s $(pgrep -f /usr/bin/nova-consoleauth)

The expected output shows the ESTABILISHED connections:

tcp 0 0 90.147.75.205:43638 90.147.75.68:5672 **ESTABLISHED** 23532/python

tcp 0 0 90.147.75.205:52524 90.147.75.68:5672 **ESTABLISHED** 23532/python

tcp 0 0 90.147.75.205:52696 90.147.75.68:5672 **ESTABLISHED** 23532/python

tcp 0 0 90.147.75.205:53833 90.147.75.68:5672 **ESTABLISHED** 23532/python

1. Verify the process “nova-novnc” is running and listening on its port (default is 6080):

# netstat -a | grep -s 6080

The expected output is like the following:

tcp 0 0 preprod-01.ba.infn:**6080** \*:\* **LISTEN**

# Verify the Block Storage Service (Cinder)

*Preparation*:

Set your --os-\* variables in your environment:

$ source admin-openrc.sh

The following steps allow to verify the correct installation of the Openstack Block Storage Service:

1. Verify that all cinder processes are up and running:

$ for s in /etc/init.d/cinder-\*; do status $(basename $s); done

The expected output is like the following:

cinder-api start/running, process 30050

cinder-scheduler start/running, process 30087

cinder-volume start/running, process 30127

If one or more of the above processes are stopped, look at the log files in the folder /var/log/cinder in order to find the problem.  
Note: depending on your installation, the cinder-volume process may not be running on the controller node (if you have decided to deploy it on a dedicated host, you should check its status there: ssh <cinder-host> service cinder-volume status).

1. Verify the process “cinder-scheduler” is running and connected to the database and messaging server (AMQP):
   1. Verify the connection with the database (replace the DB\_PORT value with the correct value depending on your installation. Default is 3306)

# export DB\_PORT=3306

# netstat -punt | grep -s $DB\_PORT | grep -s $(pgrep -f /usr/bin/cinder-scheduler)

The expected output shows the ESTABILISHED connections:

tcp 0 0 90.147.75.205:45718 212.189.205.99:33306 **ESTABLISHED** 23943/python

tcp 0 0 90.147.75.205:45812 212.189.205.99:33306 **ESTABLISHED** 23943/python

* 1. Verify the connection with the messaging server (replace the AMQP\_PORT value with the correct value depending on your installation. Default is 5672)

# export AMQP\_PORT=5672

# netstat -punt | grep -s $AMQP\_PORT | grep -s $(pgrep -f /usr/bin/cinder-scheduler)

The expected output shows the ESTABILISHED connections:

tcp 0 0 90.147.75.205:52528 90.147.75.68:**5672** **ESTABLISHED** 23943/python

tcp 0 0 90.147.75.205:53220 90.147.75.68:**5672** **ESTABLISHED** 23943/python

1. Finally, to verify that cinder is configured properly, create a new volume:

$ cinder create --display-name test 1

The expected output is like the following:

+---------------------+--------------------------------------+

| Property | Value |

+---------------------+--------------------------------------+

| attachments | [] |

| availability\_zone | nova |

| bootable | false |

| created\_at | 2014-06-22T01:14:02.705154 |

| display\_description | None |

| display\_name | test |

| encrypted | False |

| id | ad2f9004-3939-4b1c-a234-8ab26b8fe961 |

| metadata | {} |

| size | 1 |

| snapshot\_id | None |

| source\_volid | None |

| status | creating |

| volume\_type | None |

+---------------------+--------------------------------------+

1. Check the volume status using the command “cinder list”. The status should pass from “creating” to “available”:

$ cinder list

The expected output is like the following:

+--------------------------------------+-----------+--------------+------+-------------+----------+-------------+

| ID | Status | Display Name | Size | Volume Type | Bootable | Attached to |

+--------------------------------------+-----------+--------------+------+-------------+----------+-------------+

| ad2f9004-3939-4b1c-a234-8ab26b8fe961 | available | test | 1 | None | false | |

| cfe55712-5933-42fe-b9a2-aacaa8620cd6 | creating | test | 1 | None | false | |

+--------------------------------------+-----------+--------------+------+-------------+----------+-------------+

If the status value is not *available*, the volume creation failed. Check the log files in the /var/log/cinder/ directory on the controller and volume nodes to get information about the failure.

# Verify the Networking Service (Neutron)

*Preparation*:

Set your --os-\* variables in your environment:

$ source admin-openrc.sh

The following steps allow to verify the correct installation of the Openstack Networking Service. In this guide we assume that the networking services have been deployed onto a dedicated node (network node); therefore the following commands should be issued on the network node.

1. Verify that all neutron processes are up and running:

$ for s in /etc/init.d/neutron-\*; do status $(basename $s); done

The expected output is like the following:

neutron-dhcp-agent start/running, process 7515

neutron-l3-agent start/running, process 7529

neutron-metadata-agent start/running, process 7537

neutron-ovs-cleanup start/running

neutron-plugin-openvswitch-agent start/running, process 7812

neutron-server start/running, process 7820

If one or more of the above processes are stopped, look at the log files in the folder /var/log/neutron in order to find the problem.

1. Query the neutron API to get the list of networks:

$ neutron net-list

The expected output shows the list of the available networks (if any).

1. Verify the process “neutron-dhcp-agent” is running and connected to the messaging server (AMQP):
   1. replace the AMQP\_PORT value with the correct value depending on your installation. Default is 5672

# export AMQP\_PORT=5672

# netstat -punt | grep -s $AMQP\_PORT | grep -s $(pgrep -f /usr/bin/neutron-dhcp-agent)

The expected output shows the ESTABILISHED connections:

tcp 0 0 90.147.75.218:47911 90.147.75.68:**5672** **ESTABLISHED** 14058/python

tcp 0 0 90.147.75.218:47912 90.147.75.68:**5672** ESTABLISHED 14058/python

tcp 0 0 90.147.75.218:47910 90.147.75.68:**5672** **ESTABLISHED** 14058/python

1. Verify the process “neutron-l3-agent” is running and connected to the messaging server (AMQP):
   1. replace the AMQP\_PORT value with the correct value depending on your installation. Default is 5672

# export AMQP\_PORT=5672

# netstat -punt | grep -s $AMQP\_PORT | grep -s $(pgrep -f /usr/bin/neutron-l3-agent)

The expected output shows the ESTABILISHED connections:

tcp 0 0 90.147.75.218:46892 90.147.75.69:5672 **ESTABLISHED** 9986/python

tcp 0 0 90.147.75.218:47843 90.147.75.68:5672 **ESTABLISHED** 9986/python

tcp 0 0 90.147.75.218:46891 90.147.75.69:5672 **ESTABLISHED** 9986/python

# Verify the services on the Compute Nodes

Check that the compute and networking agents are up and running and able to communicate with the controller. Use the commands “nova service-list” and “neutron agent-list” (they can be issued from the controller node).

1. Load the admin credentials:

$ source admin-openrc.sh

1. verify that all the compute nodes are up:

$ nova service-list | grep nova-compute

| nova-compute | preprod-05 | nova | enabled | **up** | 2014-11-10T12:30:30.000000 | None |

| nova-compute | preprod-03 | nova | enabled | **up** | 2014-11-17T15:49:45.000000 | None |

| nova-compute | preprod-04 | nova | enabled | **up** | 2014-11-17T22:48:51.000000 | None

1. verify that the neutron open-vswitch agent is running on the compute nodes:

$ neutron agent-list | grep vSwitch

The output shows “:-)” if the service is working fine or “xxx” if there are problems

| 156c39d5-f685-450a-8fef-1bf9ca3c1e0f | Open vSwitch agent | preprod-05 | **:-)** | True |

| 30facdbc-7ae5-4f84-b11c-b8908544c7af | Open vSwitch agent | preprod-04 | **:-)** | True |

| df7f0820-24d1-41e6-82db-fec1e3296af5 | Open vSwitch agent | preprod-03 | **:-)** | True |

| ed2a74b3-f246-44dd-a3f4-d1170e30b3c0 | Open vSwitch agent | preprod-02 | **:-)** | True |

# End-to-End test: VM instantiation

The following bash script can be used to check that the installed Openstack infrastructure is able to provide running virtual machines.

To execute the script you must fill the variables at the beginning of the file with proper values depending on your installation.

*IMAGE\_ID*

To set the IMAGE\_ID variable you can use the command “glance image-list” to list the available image ids.

*FLAVOR\_ID*

To set the FLAVOR variable you can use the command “nova flavor-list” to list the available flavors (both the flavor name and id can be used).

*KEY\_NAME*

To set the KEY\_NAME variable you can use the command “nova keypair-list” to list the available keypairs

*NET\_ID*

To set the NET\_ID variable you can use the command “neutron net-list” to list the available networks.

#!/bin/bash

########################################

# BEFORE RUNNING THIS SCRIPT FILL THE

# FOLLOWING VARIABLES WITH PROPER VALUES

########################################

export OS\_USERNAME=admin

export OS\_PASSWORD=<password>

export OS\_TENANT\_NAME=admin

export OS\_AUTH\_URL=http://<controller\_ip>:35357/v2.0

export IMAGE\_ID=<image id>

export FLAVOR=<flavor id or name>

export KEY\_NAME=<key name>

export NET\_ID=<network id>

#########################################

LOOP\_THRESH=5

WAIT\_TIMEOUT=30

wait\_vm\_active()

{

typeset vmid=$1

let i=0

status=

while [ $i -lt $LOOP\_THRESH -a "$status" != active ]

do

let i++

status=$(nova show $vmid | awk '/OS-EXT-STS:vm\_state/{print $4}')

echo "VM status is <$status>"

[ "$status" = active ] && continue

sleep 30

done

[ "$status" = active ]

}

######################

### MAIN

######################

if [ $# -ne 0 ]; then

echo "Usage: ./`basename $0`"

echo -e "\nThis probe tries to create a new VM.\nExit code: 0 - probe successfully run (vm is active); 1 - probe failed"

exit 1

fi

#create the test VM

VM\_ID=$(nova boot --image $IMAGE\_ID --key-name $KEY\_NAME --flavor $FLAVOR --nic net-id=$NET\_ID test-vm | sed -n 's/^|\ \+id\ \+|\ \+\([^\ ].\*\)\ \+|/\1/p')

echo "VM id is <$VM\_ID>"

# wait for vm to become active

wait\_vm\_active $VM\_ID

# check status

if [ $? -ne 0 ]; then

echo "Error: instance not running after $LOOP\_THRESH x $WAIT\_TIMEOUT [sec]"

exit\_code=1

else

echo "OK. VM creation was successful"

exit\_code=0

fi

# terminate instance

nova delete $VM\_ID

#return 0 if test is ok, 1 otherwise

exit $exit\_code