05 2015

Brocade VCS Plugin Deployment Guide

In Mirantis OpenStack v6.0 Environment

Notice: This document is for informational purposes only and does not set forth any warranty, expressed or implied, concerning any equipment, equipment feature, or service offered or to be offered by Brocade. Brocade reserves the right to make changes to this document at any time, without notice, and assumes no responsibility for its use. This informational document describes features that may not be currently available. Contact a Brocade sales office for information on feature and product availability. Export of technical data contained in this document may require an export license from the United States government.

The authors and Brocade Communications Systems, Inc. shall have no liability or responsibility to any person or entity with respect to any loss, cost, liability, or damages arising from the information contained in this book or the computer programs that accompany it.

Preface

This document is a deployment guide for implementing a Brocade VCS Plugin, including the key features and options supported NOS device. It is written for technology decision-makers, architects, systems engineers, NOC engineers and other experts responsible for network upgrades and migration.

This document provides step-by-step examples to prepare, perform, and verify the deployment of a Brocade VCS Plugin. It is assumed that the reader is familiar with establishing console access and entering commands using the Brocade CLI.

Overview

Brocade provides a network service to OpenStack community by providing plugin to communicate with VCS/VDX devices. This Plugin supports management of L2 network and L3 router on VCS/VDX devices. Plugin provides the services provided by OpenvSwitch to create the VLAN bridges on Compute Nodes. This plugin deploys VLAN configuration and AMPP (Automatic migration of Port Profiles) configurations in the VCS device.

SVI support will be implemented using Openstack L3 extension API. The l3 service framework will be used to create the router, add the subnet to VLAN.

Document History

Date	Version	Description
2015-05-07	1.0	Initial Release

About Brocade

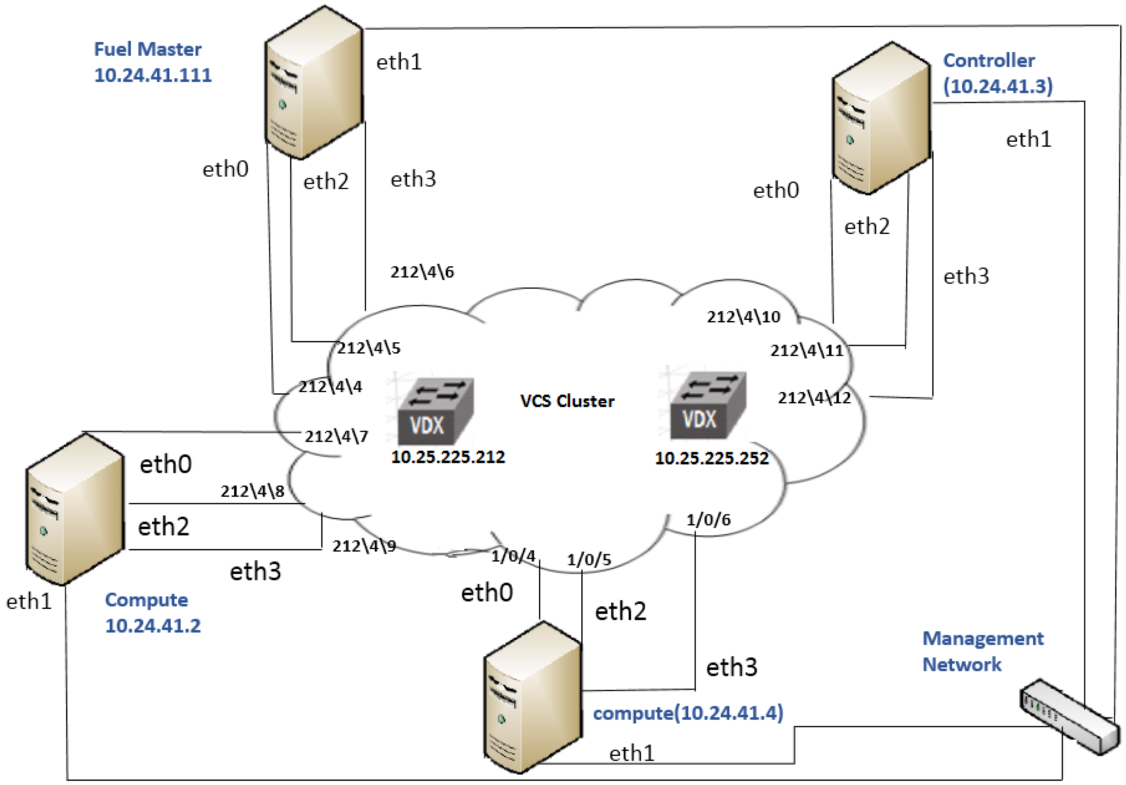
Brocade® (NASDAQ: BRCD) networking solutions help the world's leading organizations transition smoothly to a world where applications and information reside anywhere. This vision is designed to deliver key business benefits such as unmatched simplicity, non-stop networking, application optimization, and investment protection.

Innovative Ethernet and storage networking solutions for datacenter, campus, and service provider networks help reduce complexity and cost while enabling virtualization and cloud computing to increase business agility.

To help ensure a complete solution, Brocade partners with world-class IT companies and provides comprehensive education, support, and professional services offerings. (www.brocade.com)

Table of Contents

Network Topology



Prerequisites

1. A user can download MOS version with or without purchasing a support contract (subscription) from Mirantis http://software.mirantis.com/
2. More details about System/Hardware requirements, recommended configurations and setting up the environment are available at http://docs.mirantis.com/openstack/fuel/fuel-6.0/

Support Matrix

Environment	Description
Supported VCS Devices	VDX6710, VDX6720, VDX6730 VDX8770, VDX6740, VDX6740T

Supported NOS Versions	NOS4.0.x /NOS5.0.x
Supported VCS Cluster Mode	Logical Chassis
Supported OS Platform	Ubuntu 12.04 LTS
Upstream OpenStack Versions	2014.2 (JUNO)
VLAN Limit/Range	1 - 4096
Mirantis OpenStack Version	6.x

APIs serviced by Brocade VCS L2 Plugin:

List of API calls that are serviced by Brocade VCS plugin are –

Create_Network -> Using this API, Brocade Plugin will create a VLAN on VCS device
Create_Port -> Using this API, Brocade Plugin will create a Virtual machine on the compute node corresponding to the VLAN (selected network) on the VCS device
Delete_Network -> Using this API, Brocade Plugin will delete the VLAN on the VCS device corresponding to that Network
Delete_Port -> Using this API, Brocade Plugin will delete the Virtual machine on the compute node and clear the port association for the VLAN (selected network) on the VCS device

APIs serviced by Brocade VCS L3 Plugin:

List of API calls that are serviced by Brocade VCS L3 plugin are –

Create_router, Update_router -> create_router and update_router methods will set the admin_state state as down
add_interface_to_router -> Openstack subnet_id is provided in the request parameters. Using this API, Brocade Plugin will create ve interface and assign gateway ip of the subnet to the VCS device
Remove_interface_from_router -> Using this API, Brocade Plugin will remove ve interface along with assigned gateway ip of the subnet from VCS device
Delete_router -> This method is used to clean up the NETCONF, db connection and any caches.

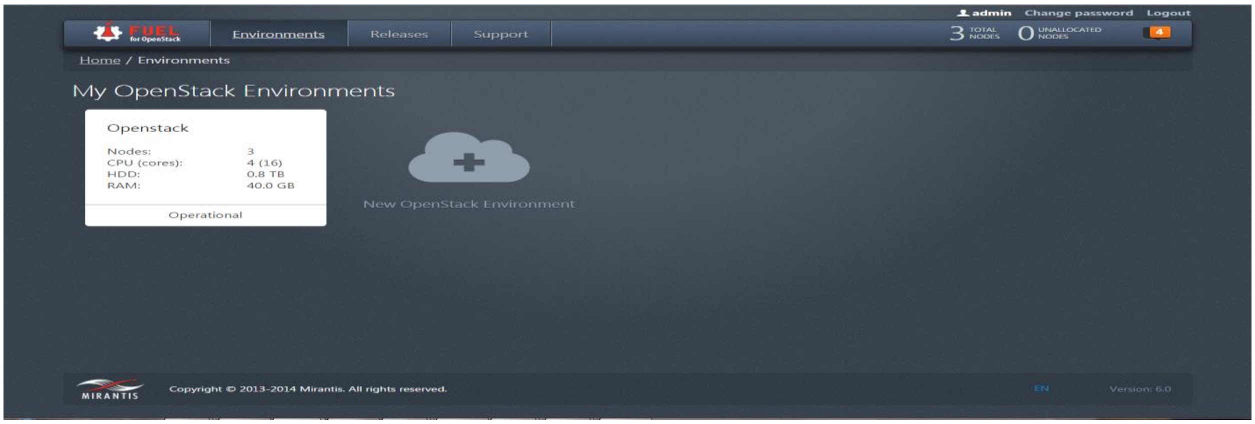
Master Node Configuration:

Master Node is the primary node, which manages all the Controller and Compute nodes and is responsible for

- Deploying the OS (Operating System) in the Controller and Compute nodes using PXE boot
- Deploying the Controller and Compute node configuration easily through Fuel UI
- Checking the status of the Controller and Compute nodes regularly

OpenStack deployment is made easier through intuitive UI called Fuel UI which is hosted in the Master Node. Fuel UI can also provide options for Network settings, Health Check and Logs tracking. Below are the step by step procedures for configuration and troubleshooting Controller and Compute nodes using Fuel UI.

1. Create Environment – click on the ‘New OpenStack Environment’ cloud icon. Environment can be able to Reset or Delete using ‘Actions’ tab



My OpenStack Environments

Openstack

Nodes: 3
CPU (cores): 4 (16)
HDD: 0.8 TB
RAM: 40.0 GB

Operational

Copyright © 2013-2014 Mirantis. All rights reserved.

Configuration Check – Brocade VCS Device Configurations:

Login to the switch 10.25.225.212 with username and password as admin/password.

Associate the vlan in the interfaces as follows,

VDX device connection to Fuel Master Node:

```
sw0(config)# interface TenGigabitEthernet 212/4/4
sw0(conf-if-te-212/4/4)# switchport
sw0(conf-if-te-212/4/4)# switchport mode access
sw0(conf-if-te-212/4/4)# switchport access vlan 100
sw0(conf-if-te-212/4/4)#no shutdown
sw0(conf-if-te-212/4/4)#exit

sw0(config)# interface TenGigabitEthernet 212/4/5
sw0(conf-if-te-212/4/5)# switchport
sw0(conf-if-te-212/4/5)#switchport mode trunk
sw0(conf-if-te-212/4/5)# switchport trunk allowed vlan add 101-102
sw0(conf-if-te-212/4/5)# switchport trunk tag native-vlan
sw0(conf-if-te-212/4/5)#no shutdown
sw0(conf-if-te-212/4/5)#exit

sw0(config)#interface TenGigabitEthernet 212/4/6
sw0(conf-if-te-212/4/6)#port-profile-port
sw0(conf-if-te-212/4/6)#no shutdown
sw0(conf-if-te-212/4/6)#exit
```

VDX device connection to Slave Node 1:

```
sw0(config)# interface TenGigabitEthernet 212/4/10
sw0(conf-if-te-212/4/10)# switchport
sw0(conf-if-te-212/4/10)# switchport mode access
sw0(conf-if-te-212/4/10)# switchport access vlan 100
sw0(conf-if-te-212/4/10)#no shutdown
sw0(conf-if-te-212/4/10)#exit

sw0(config)# interface TenGigabitEthernet 212/4/11
sw0(conf-if-te-212/4/11)# switchport
sw0(conf-if-te-212/4/11)#switchport mode trunk
sw0(conf-if-te-212/4/11)# switchport trunk allowed vlan add 101-102
sw0(conf-if-te-212/4/11)# switchport trunk tag native-vlan
sw0(conf-if-te-212/4/11)#no shutdown
```

```
sw0(conf-if-te-212/4/11)#exit

sw0(config)#interface TenGigabitEthernet 212/4/12
sw0(conf-if-te-212/4/12)#port-profile-port
sw0(conf-if-te-212/4/12)#no shutdown
sw0(conf-if-te-212/4/12)#exit
```

VDX device connection to Slave Node 2:

```
sw0(config)# interface TenGigabitEthernet 212/4/7
sw0(conf-if-te-212/4/7)# switchport
sw0(conf-if-te-212/4/7)# switchport mode access
sw0(conf-if-te-212/4/7)# switchport access vlan 100
sw0(conf-if-te-212/4/7)#no shutdown
sw0(conf-if-te-212/4/7)#exit

sw0(config)# interface TenGigabitEthernet 212/4/8
sw0(conf-if-te-212/4/8)# switchport
sw0(conf-if-te-212/4/8)#switchport mode trunk
sw0(conf-if-te-212/4/8)# switchport trunk allowed vlan add 101-102
sw0(conf-if-te-212/4/8)# switchport trunk tag native-vlan
sw0(conf-if-te-212/4/8)#no shutdown
sw0(conf-if-te-212/4/8)#exit

sw0(config)#interface TenGigabitEthernet 212/4/9
sw0(conf-if-te-212/4/9)#port-profile-port
sw0(conf-if-te-212/4/9)#no shutdown
sw0(conf-if-te-212/4/9)#exit
```

Eth0 Configurations:

VCS ports which are connected to eth0 (Admin PXE) of the servers should be configured with Access mode configurations as below –verify it by the command

sw0#show running-config interface TenGigabitEthernet 212/4/4

```
interface TenGigabitEthernet 212/4/4
 fabric isl enable
 fabric trunk enable
 switchport
 switchport mode access
 switchport access vlan 100
 spanning-tree shutdown
 no shutdown
```

Eth2 Configurations:

VCS ports which are connected to eth2 (Private, Management and Storage networks) of the servers should be configured with Trunk mode configurations as below – verify it by the command

sw0#show running-config interface TenGigabitEthernet 212/4/5

```
interface TenGigabitEthernet 212/4/5
 fabric isl enable
 fabric trunk enable
 switchport
 switchport mode trunk
 switchport trunk allowed vlan add 101-102
 switchport trunk tag native-vlan
 spanning-tree shutdown
 no shutdown
```

Eth3 Configurations:

VCS ports which are connected to eth3 of the servers should be configured with Profile mode and this is the Brocade plugin's functionality- verify it by the command

sw0#show running-config interface TenGigabitEthernet 212/4/6

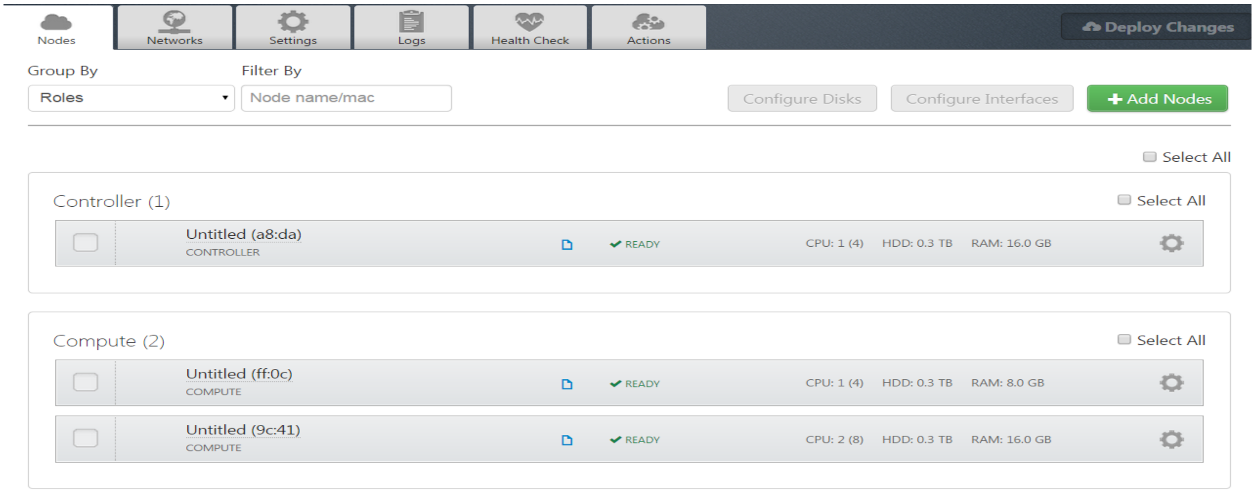
```
interface TenGigabitEthernet 212/4/6
 fabric isl enable
 fabric trunk enable
 port-profile-port
 no shutdown
```

Other Brocade VCS Commands:

Below are the VCS commands to verify the VLAN and Port Profile configurations

```
#show vlan brief
#show port-profile status
#show mac-address-table
```

2. Make sure nodes are detected in Master Node. Click on 'Add Nodes', to assign the Roles to the discovered nodes.



Group By: Roles

Filter By: Node name/mac

Controller (1) □ Select All

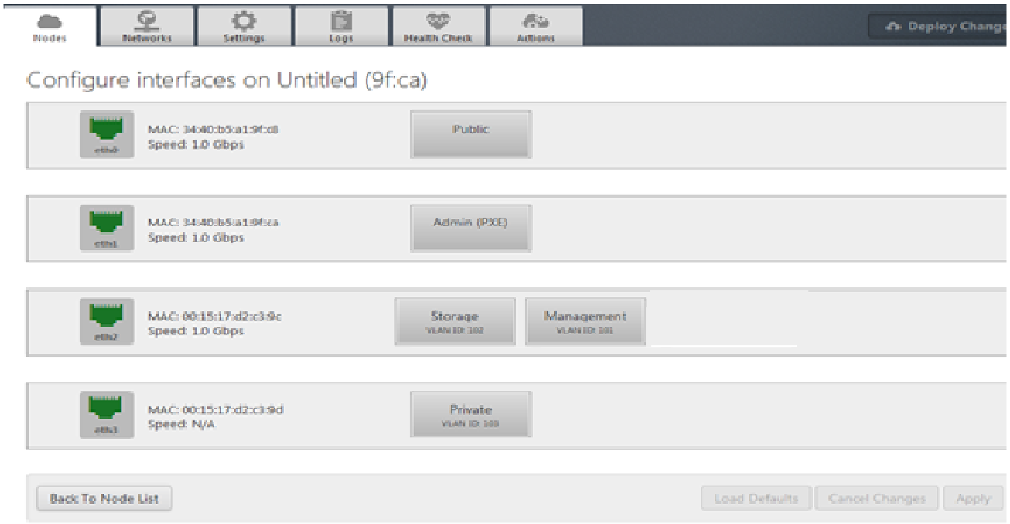
□ Untitled (a8:da) CONTROLLER ✓ READY CPU: 1 (4) HDD: 0.3 TB RAM: 16.0 GB

Compute (2) □ Select All

□ Untitled (ff:0c) COMPUTE ✓ READY CPU: 1 (4) HDD: 0.3 TB RAM: 8.0 GB

□ Untitled (9c:41) COMPUTE ✓ READY CPU: 2 (8) HDD: 0.3 TB RAM: 16.0 GB

3. Public, Admin, Private, Storage and Management networks can be associated to their respective Ethernet interfaces using 'Configure Interfaces' option.



Configure interfaces on Untitled (9f:ca)

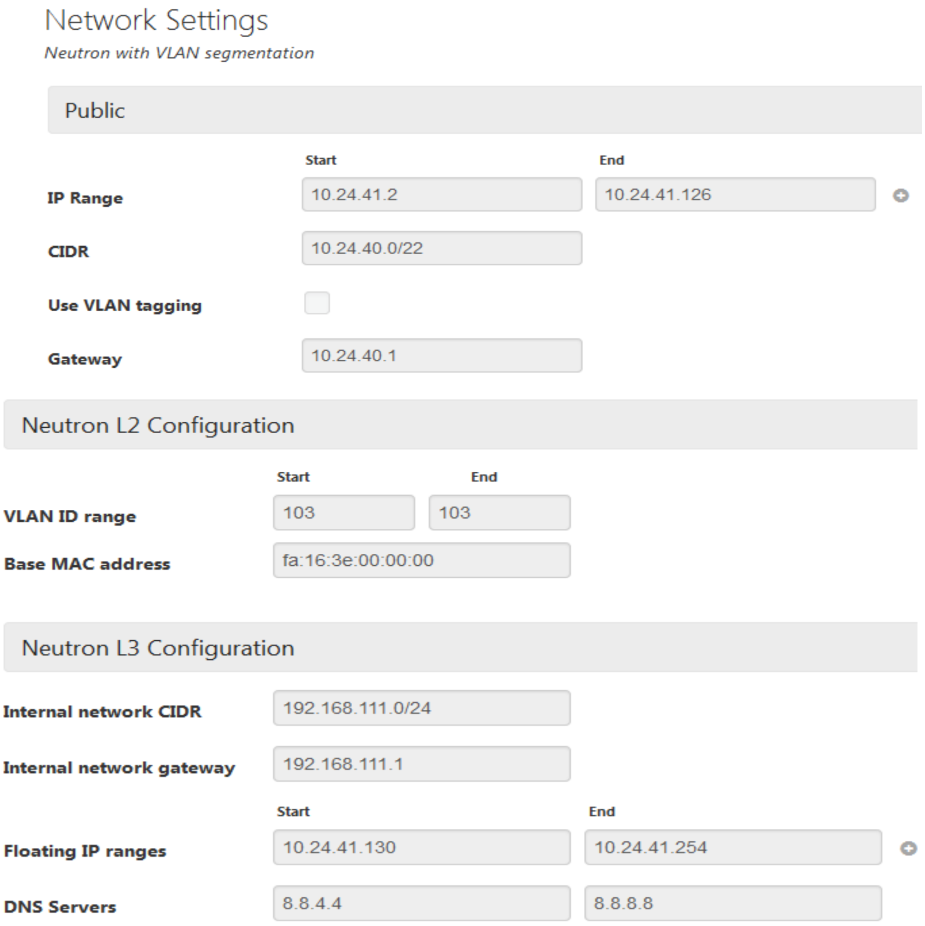
eth0
MAC: 34:40:b5:a1:9f:c8
Speed: 1.0 Gbps
Public

eth1
MAC: 34:40:b5:a1:9f:ca
Speed: 1.0 Gbps
Admin (PXE)

eth2
MAC: 00:15:17:d2:c3:9c
Speed: 1.0 Gbps
Storage VLAN ID: 102
Management VLAN ID: 101

eth3
MAC: 00:15:17:d2:c3:9d
Speed: N/A
Private VLAN ID: 103

4. Verify Networks tab is populated with correct values



Network Settings

Neutron with VLAN segmentation

Public

	Start	End
IP Range	10.24.41.2	10.24.41.126
CIDR	10.24.40.0/22	
Use VLAN tagging	□	
Gateway	10.24.40.1	

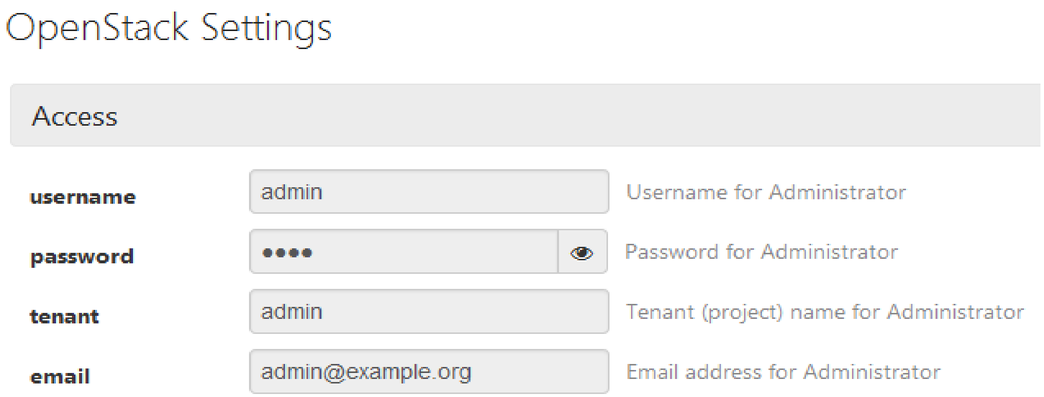
Neutron L2 Configuration

	Start	End
VLAN ID range	103	103
Base MAC address	fa:16:3e:00:00:00	

Neutron L3 Configuration

	Start	End
Internal network CIDR	192.168.111.0/24	
Internal network gateway	192.168.111.1	
Floating IP ranges	10.24.41.130	10.24.41.254
DNS Servers	8.8.4.4	8.8.8.8

5. Horizon credentials can be configured in Master node as –


OpenStack Settings

Access

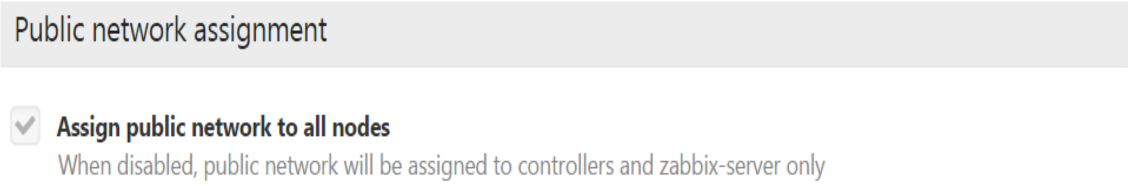
username: admin — Username for Administrator

password: •••• — Password for Administrator

tenant: admin — Tenant (project) name for Administrator

email: admin@example.org — Email address for Administrator

6. Public network can be assigned to all node as


Public network assignment

☑ Assign public network to all nodes

When disabled, public network will be assigned to controllers and zabbix-server only

7. Click on 'Deploy Changes' button

Configuration Check – Brocade VCS Plugin code and ncclient:

1. Verify the presence of Brocade VCS ML2 plugin code at location /usr/lib/python2.7/dist-packages/neutron/plugins/ml2/drivers/brocade and compare the files with Ice House upstream.
   Make sure the below code is available at nosdriver.py file
   /usr/lib/python2.7/dist-packages/neutron/plugins/ml2/drivers/brocade/nos/nosdriver.py
2. On the controller node, install the netconf client (ncclient) which is required to communicate with the Brocade VCS cluster

```
# apt-get install git
# git clone https://code.grnet.gr/git/ncclient
# cd ncclient && python setup.py install
```

Configuration Check – Neutron Configuration Files:

Make sure below mentioned configurations are done in respective configuration files in Controller and Compute nodes

/etc/neutron/neutron.conf

```
[database]
connection =
mysql://neutron:password@192.168.0.3:3306/neutron_ml2?read_timeout=60
```

/etc/neutron/plugins/openvswitch/ovs_neutron_plugin.ini (Only on the compute nodes)

```
[ovs]
tenant_network_type = vlan
network_vlan_ranges =physnet1:400:600
integration_bridge = br-int
bridge_mappings =physnet1:br-eth3
```

/etc/neutron/plugins/ml2/ml2_conf.ini

```
[ml2]
tenant_network_types = vlan,flat,local
type_drivers = vlan,flat,local
mechanism_drivers = openvswitch,brocade
[ml2_brocade]
#VCS cluster credential
username = admin
password = password
address  = 10.25.225.133
ostype = NOS
physical_networks = physnet1
```

ovs configuration

Make sure ovs-vsctl lists the required bridge configurations as below -

```
root@node-18:~# ovs-vsctl show
583bdc4f-52d0-493a-8d51-a613a4da6c9a
    Bridge "br-eth2"
        Port "br-eth2"
            Interface "br-eth2"
                type: internal
        Port "br-eth2--br-storage"
            tag: 102
            Interface "br-eth2--br-storage"
                type: patch
                options: {peer="br-storage--br-eth2"}
        Port "br-eth2--br-mgmt"
            tag: 101
            Interface "br-eth2--br-mgmt"
                type: patch
                options: {peer="br-mgmt--br-eth2"}
        Port "eth2"
            Interface "eth2"
        Port "br-eth2--br-prv"
            Interface "br-eth2--br-prv"
                type: patch
                options: {peer="br-prv--br-eth2"}
    Bridge br-mgmt
        Port "br-mgmt--br-eth2"
```

```
            Interface "br-mgmt--br-eth2"
                type: patch
                options: {peer="br-eth2--br-mgmt"}
        Port br-mgmt
            Interface br-mgmt
                type: internal
    Bridge "br-eth0"
        Port "br-eth0"
            Interface "br-eth0"
                type: internal
        Port "br-eth0--br-ex"
            trunks: [0]
            Interface "br-eth0--br-ex"
                type: patch
                options: {peer="br-ex--br-eth0"}
        Port "eth0"
            Interface "eth0"
    Bridge "br-eth1"
        Port "br-eth1--br-fw-admin"
            trunks: [0]
            Interface "br-eth1--br-fw-admin"
                type: patch
                options: {peer="br-fw-admin--br-eth1"}
        Port "eth1"
            Interface "eth1"
        Port "br-eth1"
            Interface "br-eth1"
                type: internal
    Bridge br-ex
        Port "br-ex--br-eth0"
            trunks: [0]
            Interface "br-ex--br-eth0"
                type: patch
                options: {peer="br-eth0--br-ex"}
        Port br-ex
            Interface br-ex
                type: internal
        Port "qg-83437e93-e0"
            Interface "qg-83437e93-e0"
                type: internal
        Port phy-br-ex
            Interface phy-br-ex
    Bridge br-int
        Port "int-br-eth3"
            Interface "int-br-eth3"
        Port int-br-prv
            Interface int-br-prv
        Port br-int
            Interface br-int
                type: internal
        Port int-br-ex
            Interface int-br-ex
        Port "tap373e4404-77"
            tag: 6
            Interface "tap373e4404-77"
                type: internal
    Bridge br-storage
        Port br-storage
            Interface br-storage
                type: internal
        Port "br-storage--br-eth2"
            Interface "br-storage--br-eth2"
                type: patch
                options: {peer="br-eth2--br-storage"}
    Bridge br-prv
        Port phy-br-prv
            Interface phy-br-prv
        Port "br-prv--br-eth2"
            Interface "br-prv--br-eth2"
                type: patch
                options: {peer="br-eth2--br-prv"}
        Port br-prv
            Interface br-prv
                type: internal
    Bridge br-fw-admin
        Port "br-fw-admin--br-eth1"
            trunks: [0]
            Interface "br-fw-admin--br-eth1"
                type: patch
                options: {peer="br-eth1--br-fw-admin"}
        Port br-fw-admin
            Interface br-fw-admin
                type: internal
    Bridge "br-eth3"
        Port "phy-br-eth3"
            Interface "phy-br-eth3"
        Port "eth3"
            Interface "eth3"
        Port "br-eth3"
            Interface "br-eth3"
                type: internal
    ovs_version: "1.10.1"
```

SVI - L3 Networking driver.

This section describes how SVI feature can be leveraged to provide internetworking between networks configured using OpenStack.

Edit /etc/neutron/neutron.conf

```
service_plugins =
neutron.services.l3_router.brocade.l3_router_plugin.BrocadeSVIPlugin
```

A new field has been added to the existing fields in brocade.ini file.

Add the below configuration in both /etc/neutron/plugins/ml2/ml2_conf.ini and /etc/neutron/plugins/ml2/ml2_conf_brocade.ini

```
rbridge_id = <rbridge id of vcs device>
```

This field indicates the Rbridge on which the SVI interfaces would get created.

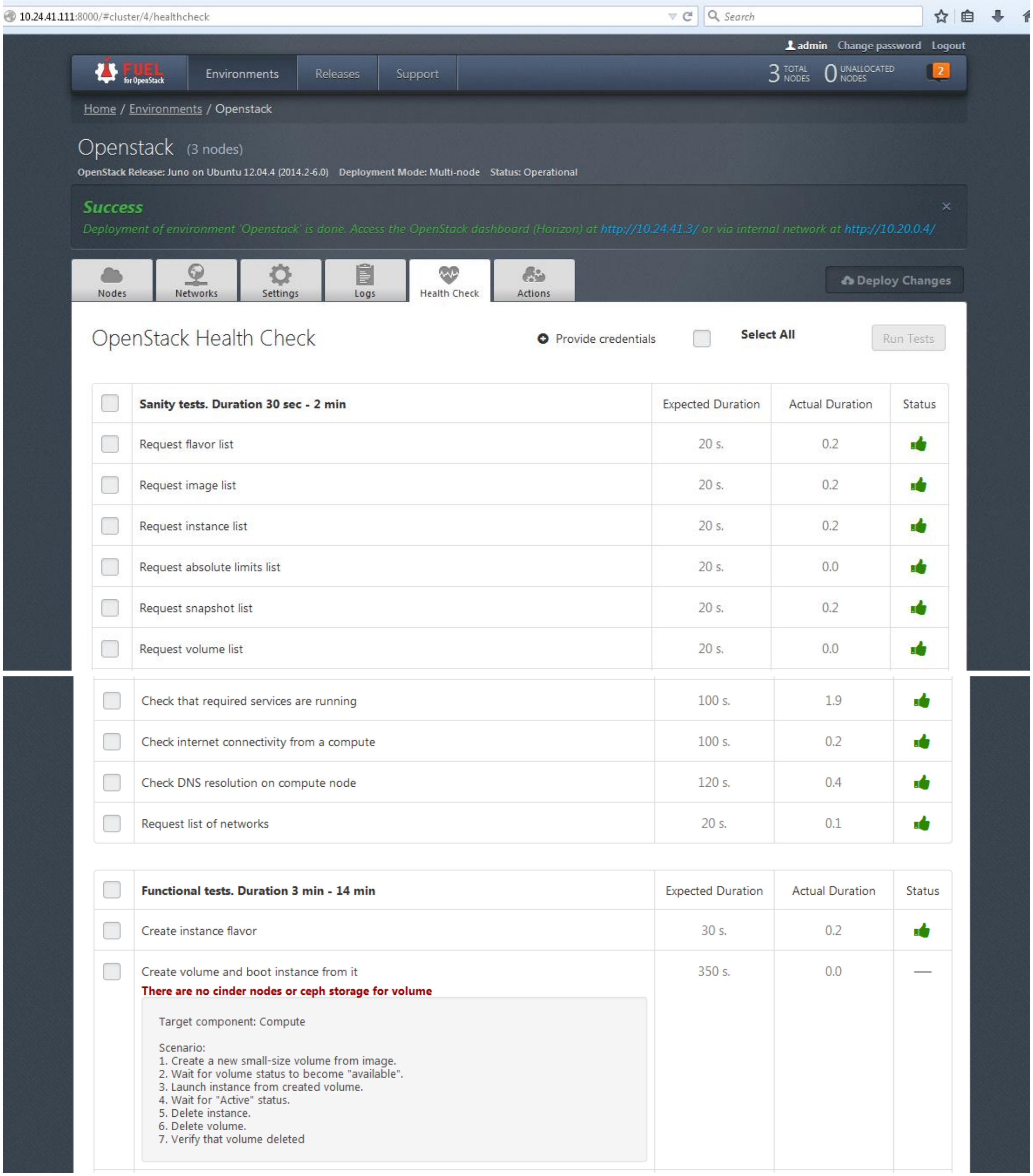
Test Report

Below are the functionality test cases that are tested as part of Brocade VCS Plugin in the Mirantis OpenStack Environment.

S.No	Test Case Title	Test Case Description	Result
1	Create Network	Verify create network is successful and VLAN is created on VCS device	Pass
2	Create Multiple Networks	Verify multiple VLANs are created on the VCS device	Pass
3	Delete Network	Verify the VLAN is deleted from the VCS device	Pass
4	Delete Multiple Networks	Verify multiple VLANs are deleted from the VCS device	Pass
5	Create Network when VLAN range exceeds	Verify VLANs are not created on VCS devices	Pass
6	Launch instance	Verify Create Instance is successful • Verify the MAC address assigned to the VM by launching VM console • Verify the same MAC address is listing in MAC address table	Pass
7	Create Multiple Instances	Verify Create Instances are successful • Verify the MAC addresses assigned to the VMs by launching VM console • Verify the mulitple MAC addresses are listing in MAC address table	Pass
8	Delete instance	Verify the MAC address of the deleted VM is not listing in MAC address table	Pass
9	Delete multiple instances	Verify the MAC addresses of the deleted VMs are not listing in MAC address table	Pass
10	Reboot of VCS Device	Verify even after the reboot of VCS device, created network (VLAN) and MAC addresses are listing properly	Pass
11	Ping between the VMs on same host.	Verify ping between the VMs present in same host is successful	Pass
12	Ping between the VMs on different host	Verify ping between the VMs present in different hosts is successful	Pass
13	Ping between the VMs on another Network.	Verify ping should not happen between VMs present in different networks	Pass
14	Create Duplicate Network with the existing subnet	Verify VLAN is not created on VCS devices	Pass
15	Disturb switch to server connection when Ping is happening between VMs.	Verify the following - • Verify the packet is getting dropped and ping is not successful • Verify the ping is getting successful again	Pass
16	Create Network and Create Port when switch is not reachable.	Verify the creation is getting failed	Pass

Health Check Results

Health Check has been run after configuring the Brocade VCS Plugin in the Mirantis Open Stack environment and below are the snapshots -



Home / Environments / Openstack

Openstack (3 nodes)

OpenStack Release: Juno on Ubuntu 12.04.4 (2014.2-6.0) Deployment Mode: Multi-node Status: Operational

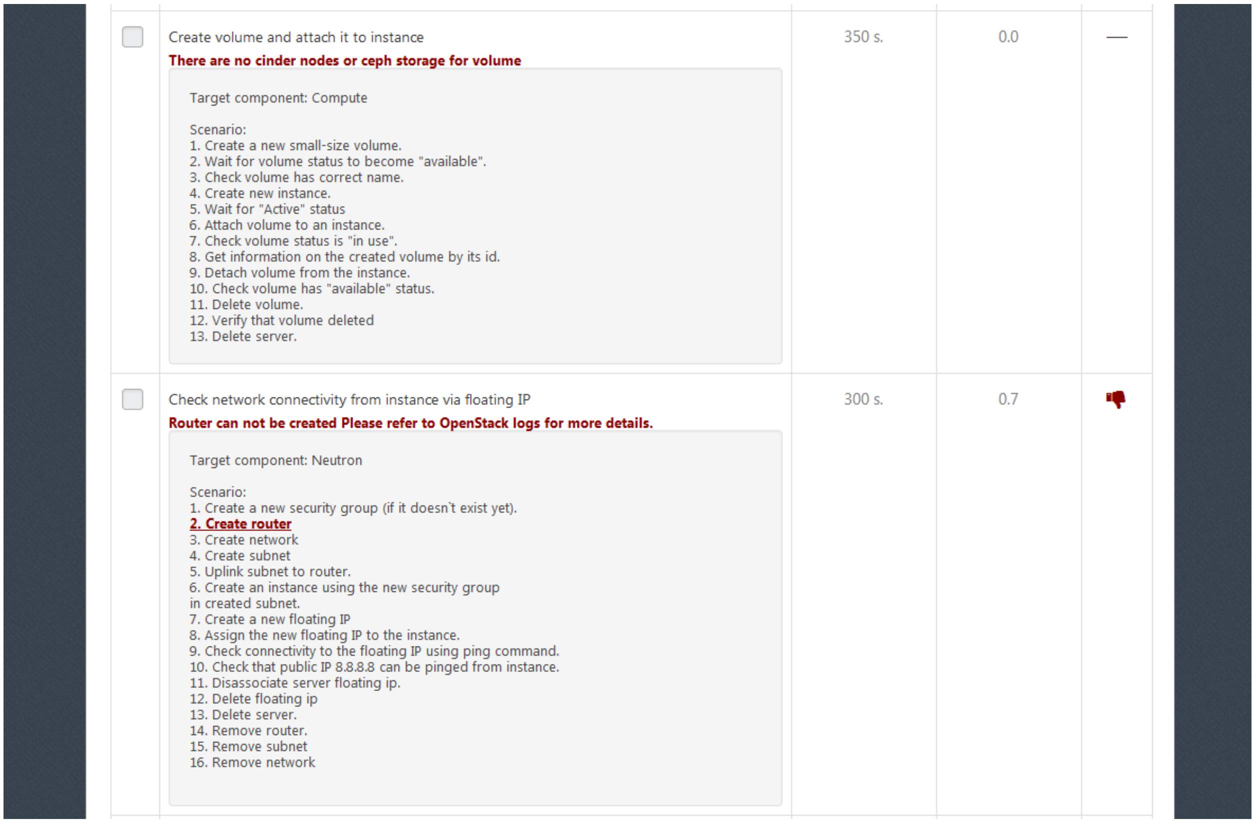
Success

Deployment of environment 'Openstack' is done. Access the OpenStack dashboard (Horizon) at http://10.24.41.3/ or via internal network at http://10.20.0.4/

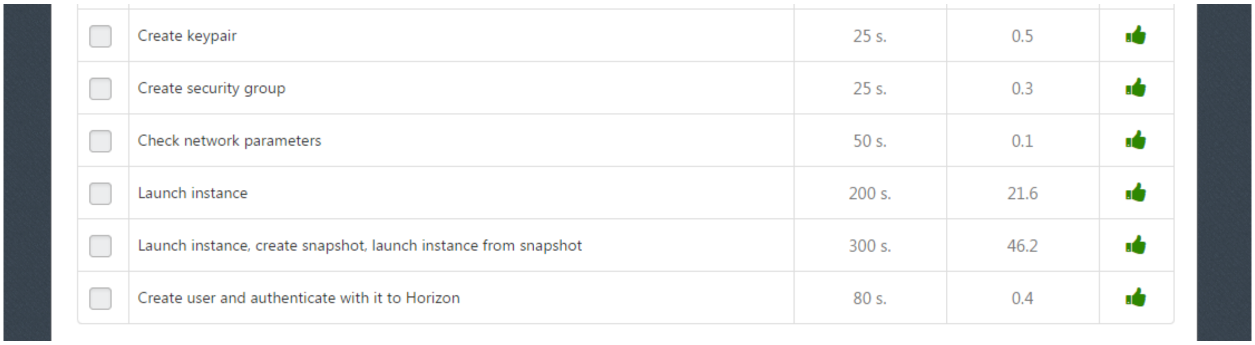
OpenStack Health Check

□	**Sanity tests. Duration 30 sec - 2 min**	Expected Duration	Actual Duration	Status
□	Request flavor list	20 s.	0.2	👍
□	Request image list	20 s.	0.2	👍
□	Request instance list	20 s.	0.2	👍
□	Request absolute limits list	20 s.	0.0	👍
□	Request snapshot list	20 s.	0.2	👍
□	Request volume list	20 s.	0.0	👍
□	Check that required services are running	100 s.	1.9	👍
□	Check internet connectivity from a compute	100 s.	0.2	👍
□	Check DNS resolution on compute node	120 s.	0.4	👍
□	Request list of networks	20 s.	0.1	👍

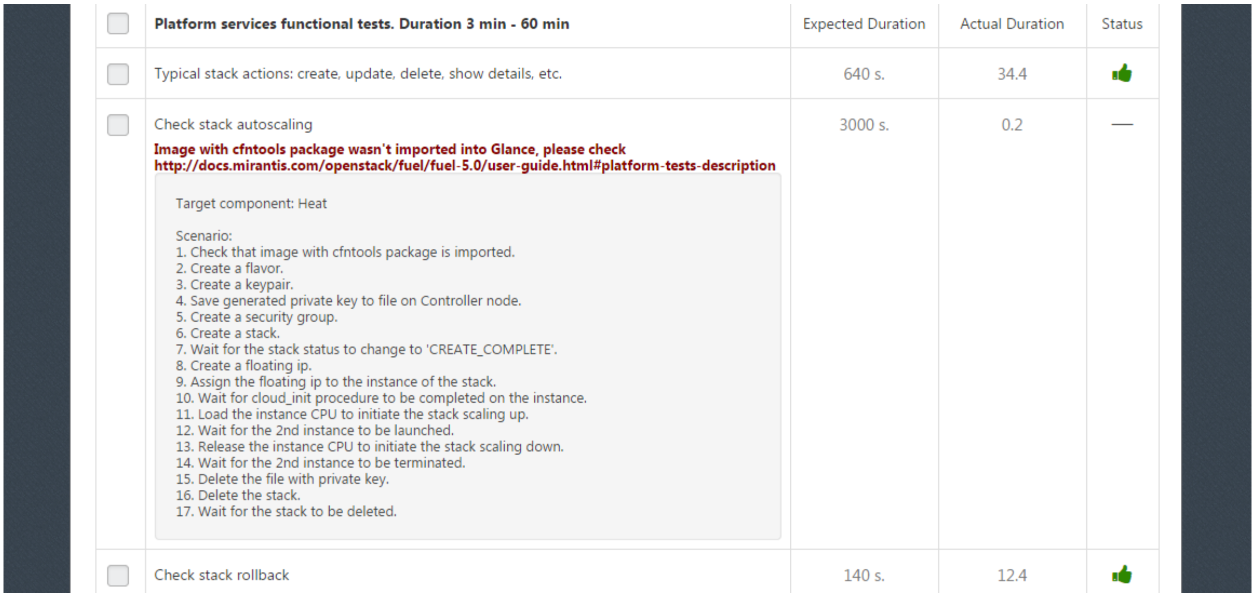
□	**Functional tests. Duration 3 min - 14 min**	Expected Duration	Actual Duration	Status
□	Create instance flavor	30 s.	0.2	👍
□	Create volume and boot instance from it **There are no cinder nodes or ceph storage for volume** Target component: Compute Scenario: 1. Create a new small-size volume from image. 2. Wait for volume status to become "available". 3. Launch instance from created volume. 4. Wait for "Active" status. 5. Delete instance. 6. Delete volume. 7. Verify that volume deleted	350 s.	0.0	—



□	Create volume and attach it to instance **There are no cinder nodes or ceph storage for volume** Target component: Compute Scenario: 1. Create a new small-size volume. 2. Wait for volume status to become "available". 3. Check volume has correct name. 4. Create new instance. 5. Wait for "Active" status 6. Attach volume to an instance. 7. Check volume status is "in use". 8. Get information on the created volume by its id. 9. Detach volume from the instance. 10. Check volume has "available" status. 11. Delete volume. 12. Verify that volume deleted 13. Delete server.	350 s.	0.0	—
□	Check network connectivity from instance via floating IP **Router can not be created Please refer to OpenStack logs for more details.** Target component: Neutron Scenario: 1. Create a new security group (if it doesn't exist yet). **2. Create router** 3. Create network 4. Create subnet 5. Uplink subnet to router. 6. Create an instance using the new security group in created subnet. 7. Create a new floating IP 8. Assign the new floating IP to the instance. 9. Check connectivity to the floating IP using ping command. 10. Check that public IP 8.8.8.8 can be pinged from instance. 11. Disassociate server floating ip. 12. Delete floating ip 13. Delete server. 14. Remove router. 15. Remove subnet 16. Remove network	300 s.	0.7	👎



□	Create keypair	25 s.	0.5	👍
□	Create security group	25 s.	0.3	👍
□	Check network parameters	50 s.	0.1	👍
□	Launch instance	200 s.	21.6	👍
□	Launch instance, create snapshot, launch instance from snapshot	300 s.	46.2	👍
□	Create user and authenticate with it to Horizon	80 s.	0.4	👍



□	**Platform services functional tests. Duration 3 min - 60 min**	Expected Duration	Actual Duration	Status
□	Typical stack actions: create, update, delete, show details, etc.	640 s.	34.4	👍
□	Check stack autoscaling **Image with cfntools package wasn't imported into Glance, please check http://docs.mirantis.com/openstack/fuel/fuel-5.0/user-guide.html#platform-tests-description** Target component: Heat Scenario: 1. Check that image with cfntools package is imported. 2. Create a flavor. 3. Create a keypair. 4. Save generated private key to file on Controller node. 5. Create a security group. 6. Create a stack. 7. Wait for the stack status to change to 'CREATE_COMPLETE'. 8. Create a floating ip. 9. Assign the floating ip to the instance of the stack. 10. Wait for cloud_init procedure to be completed on the instance. 11. Load the instance CPU to initiate the stack scaling up. 12. Wait for the 2nd instance to be launched. 13. Release the instance CPU to initiate the stack scaling down. 14. Wait for the 2nd instance to be terminated. 15. Delete the file with private key. 16. Delete the stack. 17. Wait for the stack to be deleted.	3000 s.	0.2	—
□	Check stack rollback	140 s.	12.4	👍

Note :

Neutron network creation process might hang some times with netconf client timeout expiration error. Please refer the defect in the below location for more details https://bugs.launchpad.net/neutron/+bug/1395976

Support Details

Customers with valid Mirantis support contracts can contact Mirantis for any Open Stack related issues and Brocade for any VDX/NOS and Plug-in related issues.

Below are the valid Brocade support contacts-

Brocade Contact: https://www.brocade.com/service-support/index.html
Brocade Direct Support SLA: http://www.brocade.com/services-support/support-plans/direct-support/index.page