Fuel NSXv plugin testing documentation

Release 3.0-3.0.0-1

Mirantis Inc.

Jul 21, 2016

CHAPTER

ONE

TESTING DOCUMENTS

Test Plan for NSXv plugin v2.0.0

Introduction

Purpose

Main purpose of this document is intended to describe Quality Assurance activities, required to insure that Fuel plugin for VMware NSXv driver is ready for production. The project will be able to offer VMware NSXv integration functionality with MOS. The scope of this plan defines the following objectives:

- Identify testing activities;
- Outline testing approach, test types, test cycle that will be used;
- List of metrics and deliverable elements;
- List of items for testing and out of testing scope;
- Detect exit criteria in testing purposes;
- Describe test environment.

Scope

Fuel NSXv plugin includes NSX plugin for Neutron which is developed by third party. This test plan covers a full functionality of Fuel NSXv plugin, include basic scenarios related with NSXv Neutron plugin.

Following test types should be provided:

- Smoke/BVT tests
- Integration tests
- System tests
- Destructive tests
- GUI tests

Performance testing will be executed on the scale lab and a custom set of rally scenarios must be run with NSXv environment. Configuration, environment and scenarios for performance/scale testing should be determine separately.

Intended Audience

This document is intended for project team staff (QA and Dev engineers and managers) and all other persons who are interested in testing results.

Limitation

Plugin (or its components) has the following limitations:

- VMware NSXv plugin can be enabled only with Neutron tunnel segmentation.
- Environment with enabled VMware NSXv plugin can't contains compute nodes.
- Only VMware NSX Manager Virtual Appliance 6.1.4 or later is supported.

Product compatibility matrix

Table 1.1: product compatibility matrix

Requirement	Version	Comment
MOS	8.0	
OpenStack release	Liberty with Ubuntu 14.04	
vSphere	5.5 and 6.0	
NSXv	6.2.0 and 6.1.4 (not tested)	

Evaluation Mission and Test Motivation

Project main goal is to build a MOS plugin that integrates a Neutron VMware NSX plugin. This will allow to use Neutron for networking in vmware-related environments. The plugin must be compatible with the version 8.0 of Mirantis OpenStack and should be tested with software/hardware described in *product compatibility matrix*.

See the VMware NSX Plugin specification for more details.

Evaluation mission

- Find important problems with integration of Neutron VMware NSX plugin.
- Verify a specification.
- Provide tests for maintenance update.
- Lab environment deployment.
- Deploy MOS with developed plugin installed.
- Create and run specific tests for plugin/deployment.
- Documentation.

Target Test Items

- Install/uninstall Fuel NSXv plugin
- **Deploy Cluster with Fuel NSXv plugin by Fuel**

    - **Roles of nodes**
        - controller
        - mongo
        - compute-vmware
        - cinder-vmware
    - **Hypervisors:**
        - Qemu+Vcenter
    - **Storage:**
        - Ceph
        - Cinder
        - VMWare vCenter/ESXi datastore for images
    - **Network**
        - Neutron with tunnel segmentation
        - HA + Neutron
    - **Additional components**
        - Ceilometer
        - Health Check
    - Upgrade master node
- **MOS and VMware-NSX plugin**
    - **Computes(Nova)**
        - Launch and manage instances
        - Launch instances in batch
    - **Networks (Neutron)**
        - Create and manage public and private networks.
        - Create and manage routers.
        - Port binding / disabling
        - Security groups
        - Assign vNIC to a VM
        - Connection between instances
    - **Horizon**
        - Create and manage projects
    - **Glance**
        - Create and manage images
- **GUI**
    - Fuel UI
- **CLI**

- Fuel CLI

Test approach

The project test approach consists of Smoke, Integration, System, Regression Failover and Acceptance test levels.

Smoke testing

The goal of smoke testing is to ensure that the most critical features of Fuel VMware NSXv plugin work after new build delivery. Smoke tests will be used by QA to accept software builds from Development team.

Integration and System testing

The goal of integration and system testing is to ensure that new or modified components of Fuel and MOS work effectively with Fuel VMware NSXv plugin without gaps in data flow.

Regression testing

The goal of regression testing is to verify that key features of Fuel VMware NSXv plugin are not affected by any changes performed during preparation to release (includes defects fixing, new features introduction and possible updates).

Failover testing

Failover and recovery testing ensures that the target-of-test can successfully failover and recover from a variety of hardware, software, or network malfunctions with undue loss of data or data integrity.

Acceptance testing

The goal of acceptance testing is to ensure that Fuel VMware NSXv plugin has reached a level of stability that meets requirements and acceptance criteria.

Entry and exit criteria

Criteria for test process starting

Before test process can be started it is needed to make some preparation actions - to execute important preconditions. The following steps must be executed successfully for starting test phase:

- all project requirements are reviewed and confirmed;
- implementation of testing features has finished (a new build is ready for testing);
- implementation code is stored in GIT;
- test environment is prepared with correct configuration, installed all needed software, hardware;
- test environment contains the last delivered build for testing;
- test plan is ready and confirmed internally;
- implementation of manual tests and autotests (if any) has finished.

Feature exit criteria

Testing of a feature can be finished when:

- All planned tests (prepared before) for the feature are executed; no defects are found during this run;
- All planned tests for the feature are executed; defects found during this run are verified or confirmed to be acceptable (known issues);

- The time for testing of that feature according to the project plan has run out and Project Manager confirms that no changes to the schedule are possible.

Suspension and resumption criteria

Testing of a particular feature is suspended if there is a blocking issue which prevents tests execution. Blocking issue can be one of the following:

- Testing environment for the feature is not ready
- Testing environment is unavailable due to failure
- Feature has a blocking defect, which prevents further usage of this feature and there is no workaround available
- CI tests fail

Deliverables

List of deliverables

Project testing activities are to be resulted in the following reporting documents:

- Test plan
- Test report
- Automated test cases

Acceptance criteria

- All acceptance criteria for user stories are met.
- All test cases are executed. BVT tests are passed
- Critical and high issues are fixed
- All required documents are delivered
- Release notes including a report on the known errors of that release

Test cases

Smoke

Install Fuel VMware NSX-v plugin.

ID

nsxv_install

Description

Check that plugin can be installed.

Complexity

smoke

Steps

1. Connect to fuel node via ssh.
2. Upload plugin.
3. Install plugin.

Expected result

Output:

```
[root@nailgun ~]# fuel plugins --install nsxv-3.0-3.0.0-1.noarch.rpm
Loaded plugins: fastestmirror, priorities
Examining nsxv-3.0-3.0.0-1.noarch.rpm: nsxv-3.0-3.0.0-1.noarch
Marking nsxv-3.0-3.0.0-1.noarch.rpm to be installed
Resolving Dependencies
--> Running transaction check
---> Package nsxv-3.0.noarch 0:3.0.0-1 will be installed
--> Finished Dependency Resolution

Dependencies Resolved


 Package Arch Version Repository Size
Installing:
 nsxv-3.0 noarch 3.0.0-1 /nsxv-3.0-3.0.0-1.noarch 20 M

Transaction Summary
Install  1 Package

Total size: 20 M
Installed size: 20 M
Downloading packages:
Running transaction check
Running transaction test
Transaction test succeeded
Running transaction
  Installing : nsxv-3.0-3.0.0-1.noarch 1/1
  Ssh key file exists, skip generation
  Verifying  : nsxv-3.0-3.0.0-1.noarch 1/1

Installed:
  nsxv-3.0.noarch 0:3.0.0-1

Complete!
Plugin nsxv-3.0-3.0.0-1.noarch.rpm was successfully installed.
```

Ensure that plugin is installed successfully using cli, run command 'fuel plugins'. Check name, version and package version of plugin.

Uninstall Fuel VMware NSXv plugin.

ID

nsxv_uninstall

Description

Check that plugin can be removed.

Complexity

smoke

Steps

1. Connect to fuel node with preinstalled plugin via ssh.
2. Remove plugin.

Expected result

Output:

```
[root@nailgun ~]# fuel plugins --remove nsxv==3.0.0
Loaded plugins: fastestmirror, priorities
Resolving Dependencies
--> Running transaction check
---> Package nsxv-3.0.noarch 0:3.0.0-1 will be erased
--> Finished Dependency Resolution

Dependencies Resolved

 Package  Arch  Version Repository Size
Removing:
 nsxv-3.0 noarch 3.0.0-1 @/nsxv-3.0-3.0.0-1.noarch 20 M

Transaction Summary
Remove  1 Package

Installed size: 20 M
Downloading packages:
Running transaction check
Running transaction test
Transaction test succeeded
Running transaction
  Erasing    : nsxv-3.0-3.0.0-1.noarch 1/1
  Verifying  : nsxv-3.0-3.0.0-1.noarch 1/1

Removed:
  nsxv-3.0.noarch 0:3.0.0-1
```

```
Complete!
Plugin nsxv==3.0.0 was successfully removed.
```

Verify that plugin is removed, run command 'fuel plugins'.

Verify that all elements of NSXv plugin section meets the requirements.

ID

nsxv_gui

Description

Verify that all elements of NSXv plugin section meets the requirements.

Complexity

smoke

Steps

1. Login to the Fuel web UI.
2. Click on the Networks tab.
3. Verify that section of NSXv plugin is present under the Other menu option.
4. Verify that check box 'NSXv plugin' is enabled by default.
5. Verify that all labels of 'NSXv plugin' section have the same font style and colour.
6. Verify that all elements of NSXv plugin section are vertical aligned.

Expected result

All elements of NSXv plugin section are regimented.

Deployment with plugin, controller and vmware datastore backend.

ID

nsxv_smoke

Description

Check deployment with NSXv plugin and one controller.

Complexity

smoke

Steps

1. Log into Fuel with preinstalled plugin.
2. **Create a new environment with following parameters:**
    - Compute: KVM/QEMU with vCenter
    - Networking: Neutron with tunnel segmentation
    - Storage: default
    - Additional services: default
3. **Add nodes with following roles:**
    - Controller
4. Configure interfaces on nodes.
5. Configure network settings.
6. Enable and configure NSXv plugin.
7. **Configure settings:**
    - Enable VMWare vCenter/ESXi datastore for images (Glance).
8. Configure VMware vCenter Settings. Add 1 vSphere cluster and configure Nova Compute instances on controllers.
9. Deploy cluster.
10. Run OSTF.

Expected result

Cluster should be deployed and all OSTF test cases should be passed.

Deploy HA cluster with NSXv plugin.

ID

nsxv_bvt

Description

Check deployment with NSXv plugin, 3 Controllers, 2 CephOSD, CinderVMware and computeVMware roles.

Complexity

smoke

Steps

1. Connect to the Fuel web UI with preinstalled plugin.
2. **Create a new environment with following parameters:**
    - Compute: KVM/QEMU with vCenter
    - Networking: Neutron with tunnel segmentation
    - Storage: Ceph RBD for images (Glance)
    - Additional services: default
3. **Add nodes with following roles:**
    - Controller
    - Controller
    - Controller
    - CephOSD
    - CephOSD
    - CinderVMware
    - ComputeVMware
4. Configure interfaces on nodes.
5. Configure network settings.
6. Enable and configure NSXv plugin.
7. Configure VMware vCenter Settings. Add 2 vSphere clusters and configure Nova Compute instances on controllers and compute-vmware.
8. Verify networks.
9. Deploy cluster.
10. Run OSTF.

Expected result

Cluster should be deployed and all OSTF test cases should be passed.

Check option 'HA for edges' works correct

ID

nsxv_ha_edges

Description

Check that HA on edges functions properly.

Complexity

core

Steps

1. Install NSXv plugin.
2. Enable plugin on tab Networks -> NSXv plugin.
3. Fill the form with corresponding values.
4. Set checkbox 'Enable HA for NSX Edges'.
5. Deploy cluster with one controller.
6. Run OSTF.

Expected result

Cluster should be deployed and all OSTF test cases should be passed.

Check option 'Bypass NSX Manager certificate verification' works correct

ID

nsxv_insecure_false

Description

Check that insecure checkbox functions properly.

Complexity

core

Steps

1. Install NSXv plugin.
2. Enable plugin on tab Networks -> NSXv plugin.
3. Fill the form with corresponding values.
4. Uncheck checkbox 'Bypass NSX Manager certificate verification'.
5. Deploy cluster with one controller.
6. Run OSTF.

Expected result

Cluster should be deployed and all OSTF test cases should be passed.

Verify that nsxv driver configured properly after enabling NSXv plugin

ID

nsxv_config_ok

Description

Need to check that all parameters of nsxv driver config files have been filled up with values entered from GUI. Applicable values that are typically used are described in plugin docs. Root & intermediate certificate are signed, in attachment.

Complexity

advanced

Steps

1. Install NSXv plugin.
2. Enable plugin on tab Networks -> NSXv plugin.
3. Fill the form with corresponding values.
4. Do all things that are necessary to provide interoperability of NSXv plugin and NSX Manager with certificate.
5. Check Additional settings. Fill the form with corresponding values. Save settings by pressing the button.

Expected result

Check that nsx.ini on controller nodes is properly configured.

Verify disabled roles

ID

nsxv_disabled_roles

Description

Need to check that some disabled roles are unavailable in Fuel wizard.

Complexity

smoke

Steps

1. Create new OpenStack environment.
2. Enable options ‘QENU-KVM’ and ‘vCenter’.
3. Select ‘Neutron with NSXv plugin’.
4. **On tab ‘Storage Backends’ check that are not available:**
    - Ceph - Block Storage
    - Ceph - Ephemeral Storage
5. **On tab ‘Additional Services’ check that are not available:**
    - Install Sahara
    - Install Murano
    - Install Ironic
6. Finish creating new environment.
7. On ‘Nodes’ tab press ‘Add Nodes’.
8. **Check that following roles are not available:**
    - Compute
    - Cinder

Expected result

All described roles are unavailable or in disabled state.

Deploy with specified tenant_router_types option

ID

nsxv_specified_router_type

Description

Deploy with tenant_router_types=exclusive in nsx.ini

Complexity

core

Steps

1. Install and configure nsxv plugin.
2. Specify additional parameter tenant_router_types with value 'exclusive'.
3. Deploy cluster.
4. Run OSTF.

Expected result

All OSTF are passed.

Deploy HOT

ID

nsxv_hot

Description

Template creates flavor, net, security group, instance.

Complexity

smoke

Steps

1. Deploy cluster with NSXv.
2. Copy nsxv_stack.yaml to controller on which heat will be run.
3. On controller node run command:

```
. ./openrc
heat stack-create -f nsxv_stack.yaml teststack
```

   Wait for status COMPLETE.

4. Run OSTF.

Expected result

All OSTF are passed.

Integration

Deploy cluster with NSXv plugin and ceilometer.

ID

nsxv_ceilometer

Description

Check deployment with Fuel NSXv plugin and Ceilometer.

Complexity

core

Steps

1. Log into Fuel UI with preinstalled plugin.
2. **Create a new environment with following parameters:**
    - Compute: KVM/QEMU with vCenter
    - Networking: Neutron with tunnel segmentation
    - Storage: default
    - Additional services: Ceilometer
3. **Add nodes with following roles:**
    - Controller + Mongo
    - Controller + Mongo
    - Controller + Mongo
    - ComputeVMware
4. Configure interfaces on nodes.
5. Configure network settings.
6. Enable and configure NSXv plugin.
7. Configure VMware vCenter Settings. Add 2 vSphere clusters and configure Nova Compute instances on controllers and compute-vmware.
8. Verify networks.
9. Deploy cluster.
10. Run OSTF.

Expected result

Cluster should be deployed and all OSTF test cases should be passed (excluding platform tests that require TestVM image).

Scale

Deploy cluster with plugin and deletion one node with controller role.

ID

nsxv_add_delete_controller

Description

Verifies that system functionality is ok when controller has been removed.

Complexity

core

Steps

1. Log into Fuel with preinstalled plugin.
2. **Create a new environment with following parameters:**
    - Compute: KVM/QEMU with vCenter
    - Networking: Neutron with tunnel segmentation
    - Storage: default
3. **Add nodes with following roles:**
    - Controller
    - Controller
    - Controller
    - Controller
    - CinderVMware
    - ComputeVMware
4. Configure interfaces on nodes.
5. Configure network settings.
6. Enable and configure NSXv plugin.
7. Configure VMware vCenter Settings. Add 2 vSphere clusters and configure Nova Compute instances on controllers and compute-vmware.
8. Deploy cluster.

9. Run OSTF.
10. Launch 2 VMs.
11. Remove node with controller role.
12. Redeploy cluster. Check that all instances are in place.
13. Run OSTF.
14. Add controller.
15. Redeploy cluster. Check that all instances are in place.
16. Run OSTF.

Expected result

Cluster should be deployed and all OSTF test cases should be passed.

Deployment with 3 Controlers, ComputeVMware, CinderVMware and check adding/deleting of nodes.

ID

nsxv_add_delete_nodes

Description

Verify that system functionality is ok after redeploy.

Complexity

advanced

Steps

1. Connect to a Fuel web UI with preinstalled plugin.
2. **Create a new environment with following parameters:**
    - Compute: KVM/QEMU with vCenter
    - Networking: Neutron with VLAN segmentation
    - Storage: default
    - Additional services: default
3. **Add nodes with following roles:**
    - Controller
    - Controller
    - Controller

    - ComputeVMware
4. Configure interfaces on nodes.
5. Configure network settings.
6. Enable and configure NSXv plugin.
7. Configure VMware vCenter Settings. Add 2 vSphere clusters and configure Nova Compute instances on controllers and compute-vmware.
8. Deploy cluster.
9. Run OSTF.
10. Add node with CinderVMware role. Redeploy cluster.
11. Run OSTF.
12. Remove node with CinderVMware role. Redeploy cluster.
13. Run OSTF.
14. Remove node with ComputeVMware role. Redeploy cluster.
15. Run OSTF.

Expected result

Changing of cluster configuration was successful. Cluster should be deployed and all OSTF test cases should be passed.

System

Setup for system tests

ID

nsxv_ha_mode

Description

Deploy environment with 3 controlers and 1 compute-vmware nodes. Nova Compute instances are running on controllers and compute-vmware nodes. It is a config for all system tests.

Complexity

core

Steps

1. Log into Fuel web UI with preinstalled plugin.
2. **Create a new environment with following parameters:**
    - Compute: KVM/QEMU with vCenter

- Networking: Neutron with tunnel segmentation
- Storage: default
- Additional services: default

3. **Add nodes with following roles:**
    - Controller
    - Controller
    - Controller
    - ComputeVMware
4. Configure interfaces on nodes.
5. Configure network settings.
6. Enable and configure NSXv plugin.
7. Configure VMware vCenter Settings. Add 2 vSphere clusters and configure Nova Compute instances on controllers and compute-vmware.
8. Verify networks.
9. Deploy cluster.
10. Split availability zone to vcenter1 and vcenter2 with one nova compute cluster in each zone.
11. Run OSTF.
12. Launch instances from "TestVM-VMDK" image which is included in plugin package and is available under Horizon. Use m1.tiny flavor.

Expected result

Cluster should be deployed and all OSTF test cases should be passed.

Check abilities to create and terminate networks on NSX.

ID

nsxv_create_terminate_networks

Description

Verifies that creation of network is translated to vcenter.

Complexity

core

Steps

1. Setup for system tests.
2. Log in to Horizon Dashboard.
3. Add private networks net_01 and net_02.
4. Remove private network net_01.
5. Add private network net_01.

Expected result

Check that networks are present in the vcenter. Check that network net_01 has been removed from the vcenter on appropriate step.

Check abilities to bind port on NSXv to VM, disable and enable this port.

ID

nsxv_ability_to_bind_port

Description

Verifies that system could manipulate with port.

Complexity

core

Steps

1. Log in to Horizon Dashboard.
2. Navigate to Project -> Compute -> Instances
3. Launch instance VM_1 with image TestVM-VMDK and flavor m1.tiny.
4. Launch instance VM_2 with image TestVM-VMDK and flavor m1.tiny.
5. Verify that VMs should communicate between each other. Send icmp ping from VM_1 to VM_2 and vice versa.
6. Disable NSXv_port of VM_1.
7. Verify that VMs should not communicate between each other. Send icmp ping from VM_2 to VM_1 and vice versa.
8. Enable NSXv_port of VM_1.
9. Verify that VMs should communicate between each other. Send icmp ping from VM_1 to VM_2 and vice versa.

Expected result

Pings should get a response.

Check abilities to assign multiple vNIC to a single VM.

ID

nsxv_multi_vnic

Description

Check abilities to assign multiple vNICs to a single VM.

Complexity

core

Steps

1. Setup for system tests.
2. Log in to Horizon Dashboard.
3. Add two private networks (net01 and net02).
4. Add one subnet (net01_subnet01: 192.168.101.0/24, net02_subnet01, 192.168.102.0/24) to each network. NOTE: We have a constraint about network interfaces. One of subnets should have gateway and another should not. So disable gateway on that subnet.
5. Launch instance VM_1 with image TestVM-VMDK and flavor m1.tiny in vcenter1 az.
6. Launch instance VM_2 with image TestVM-VMDK and flavor m1.tiny in vcenter2 az.
7. Check abilities to assign multiple vNIC net01 and net02 to VM_1.
8. Check abilities to assign multiple vNIC net01 and net02 to VM_2.
9. Send icmp ping from VM_1 to VM_2 and vice versa.

Expected result

VM_1 and VM_2 should be attached to multiple vNIC net01 and net02. Pings should get a response.

Check connectivity between VMs attached to different networks with a router between them.

ID

nsxv_connectivity_diff_networks

Description

Verifies that there is a connection between networks connected through the router.

Complexity

core

Steps

1. Setup for system tests.
2. Log in to Horizon Dashboard.
3. Add two private networks (net01, and net02).
4. Add one subnet (net01_subnet01: 192.168.101.0/24, net02_subnet01, 192.168.102.0/24) to each network. Disable gateway for all subnets.
5. Navigate to Project -> Compute -> Instances
6. Launch instances VM_1 and VM_2 in the network 192.168.101.0/24 with image TestVM-VMDK and flavor m1.tiny in vcenter1 az. Attach default private net as a NIC 1.
7. Launch instances VM_3 and VM_4 in the network 192.168.102.0/24 with image TestVM-VMDK and flavor m1.tiny in vcenter2 az. Attach default private net as a NIC 1.
8. Verify that VMs of same networks should communicate between each other. Send icmp ping from VM_1 to VM_2, VM_3 to VM_4 and vice versa.
9. Verify that VMs of different networks should not communicate between each other. Send icmp ping from VM_1 to VM_3, VM_4 to VM_2 and vice versa.
10. Create Router_01, set gateway and add interface to external network.
11. Enable gateway on subnets. Attach private networks to router.
12. Verify that VMs of different networks should communicate between each other. Send icmp ping from VM_1 to VM_3, VM_4 to VM_2 and vice versa.
13. Add new Router_02, set gateway and add interface to external network.
14. Detach net_02 from Router_01 and attach to Router_02
15. Assign floating IPs for all created VMs.
16. Verify that VMs of different networks should communicate between each other. Send icmp ping from VM_1 to VM_3, VM_4 to VM_2 and vice versa.

Expected result

Pings should get a response.

Check connectivity between VMs attached on the same provider network with shared router.

ID

nsxv_connectivity_via_shared_router

Description

Checks that it is possible to connect via shared router type.

Complexity

core

Steps

1. Setup for system tests.
2. Log in to Horizon Dashboard.
3. Create shared router(default type) and use it for routing between instances.
4. Navigate to Project -> Compute -> Instances
5. Launch instance VM_1 in the provider network with image TestVM-VMDK and flavor m1.tiny in the vcenter1 az.
6. Launch instance VM_2 in the provider network with image TestVM-VMDK and flavor m1.tiny in the vcenter2 az.
7. Verify that VMs of same provider network should communicate between each other. Send icmp ping from VM_1 to VM_2 and vice versa.

Expected result

Pings should get a response.

Check connectivity between VMs attached on the same provider network with distributed router.

ID

nsxv_connectivity_via_distributed_router

Description

Verifies that there is possibility to connect via distributed router type.

Complexity

core

Steps

1. Setup for system tests.
2. Log in to Horizon Dashboard.
3. Create distributed router and use it for routing between instances. Only available via CLI:

```
neutron router-create rdistributed --distributed True
```

4. Disconnect default networks private and floating from default router and connect to distributed router.
5. Navigate to Project -> Compute -> Instances
6. Launch instance VM_1 in the provider network with image TestVM-VMDK and flavor m1.tiny in the vcenter1 az.
7. Launch instance VM_2 in the provider network with image TestVM-VMDK and flavor m1.tiny in the vcenter2 az.
8. Verify that VMs of same provider network should communicate between each other. Send icmp ping from VM_1 to VM_2 and vice versa.

Expected result

Pings should get a response.

Check connectivity between VMs attached on the same provider network with exclusive router.

ID

nsxv_connectivity_via_exclusive_router

Description

Verifies that there is possibility to connect via exclusive router type.

Complexity

core

Steps

1. Setup for system tests.
2. Log in to Horizon Dashboard.

3. Create exclusive router and use it for routing between instances. Only available via CLI:

```
neutron router-create rexclusive --router_type exclusive
```

4. Disconnect default networks private and floating from default router and connect to distributed router.
5. Navigate to Project -> Compute -> Instances
6. Launch instance VM_1 in the provider network with image TestVM-VMDK and flavor m1.tiny in the vcenter1 az.
7. Launch instance VM_2 in the provider network with image TestVM-VMDK and flavor m1.tiny in the vcenter2 az.
8. Verify that VMs of same provider network should communicate between each other. Send icmp ping from VM _1 to VM_2 and vice versa.

Expected result

Pings should get a response.

Check isolation between VMs in different tenants.

ID

nsxv_different_tenants

Description

Verifies isolation in different tenants.

Complexity

core

Steps

1. Setup for system tests.
2. Log in to Horizon Dashboard.
3. Create non-admin tenant test_tenant.
4. Navigate to Identity -> Projects.
5. Click on Create Project.
6. Type name test_tenant.
7. On tab Project Members add admin with admin and member. Activate test_tenant project by selecting at the top panel.
8. Navigate to Project -> Network -> Networks
9. Create network with 2 subnet. Create Router, set gateway and add interface.

10. Navigate to Project -> Compute -> Instances
11. Launch instance VM_1
12. Activate default tenant.
13. Navigate to Project -> Network -> Networks
14. Create network with subnet. Create Router, set gateway and add interface.
15. Navigate to Project -> Compute -> Instances
16. Launch instance VM_2.
17. Verify that VMs on different tenants should not communicate between each other. Send icmp ping from VM_1 of admin tenant to VM_2 of test_tenant and vice versa.

Expected result

Pings should not get a response.

Check connectivity between VMs with same ip in different tenants.

ID

nsxv_same_ip_different_tenants

Description

Verifies connectivity with same IP in different tenants. IMPORTANT: Use exclusive router. For proper work routers should be placed on different edges.

Complexity

advanced

Steps

1. Setup for system tests.
2. Log in to Horizon Dashboard.
3. Create 2 non-admin tenants ‘test_1’ and ‘test_2’.
4. Navigate to Identity -> Projects.
5. Click on Create Project.
6. Type name ‘test_1’ of tenant.
7. Click on Create Project.
8. Type name ‘test_2’ of tenant.
9. On tab Project Members add admin with admin and member.
10. In tenant ‘test_1’ create net1 and subnet1 with CIDR 10.0.0.0/24

11. In tenant ‘test_1’ create security group ‘SG_1’ and add rule that allows ingress icmp traffic
12. In tenant ‘test_2’ create net2 and subnet2 with CIDR 10.0.0.0/24
13. In tenant ‘test_2’ create security group ‘SG_2’
14. In tenant ‘test_1’ add VM_1 of vcenter1 in net1 with ip 10.0.0.4 and ‘SG_1’ as security group.
15. In tenant ‘test_1’ add VM_2 of vcenter2 in net1 with ip 10.0.0.5 and ‘SG_1’ as security group.
16. In tenant ‘test_2’ create net1 and subnet1 with CIDR 10.0.0.0/24
17. In tenant ‘test_2’ create security group ‘SG_1’ and add rule that allows ingress icmp traffic
18. In tenant ‘test_2’ add VM_3 of vcenter1 in net1 with ip 10.0.0.4 and ‘SG_1’ as security group.
19. In tenant ‘test_2’ add VM_4 of vcenter2 in net1 with ip 10.0.0.5 and ‘SG_1’ as security group.
20. Assign floating IPs for all created VMs.
21. Verify that VMs with same ip on different tenants should communicate between each other. Send icmp ping from VM_1 to VM_3, VM_2 to Vm_4 and vice versa.

Expected result

Pings should get a response.

Check connectivity Vms to public network.

ID

nsxv_public_network_availability

Description

Verifies that public network is available.

Complexity

core

Steps

1. Setup for system tests.
2. Log in to Horizon Dashboard.
3. Create net01: net01_subnet, 192.168.112.0/24 and attach it to the router04
4. Launch instance VM_1 of vcenter1 AZ with image TestVM-VMDK and flavor m1.tiny in the net_04.
5. Launch instance VM_1 of vcenter2 AZ with image TestVM-VMDK and flavor m1.tiny in the net_01.
6. Send ping from instances VM_1 and VM_2 to 8.8.8.8 or other outside ip.

Expected result

Pings should get a response.

Check connectivity VMs to public network with floating ip.

ID

nsxv_floating_ip_to_public

Description

Verifies that public network is available via floating ip.

Complexity

core

Steps

1. Setup for system tests.
2. Log in to Horizon Dashboard
3. Create net01: net01_subnet, 192.168.112.0/24 and attach it to the router04
4. Launch instance VM_1 of vcenter1 AZ with image TestVM-VMDK and flavor m1.tiny in the net_04. Associate floating ip.
5. Launch instance VM_1 of vcenter2 AZ with image TestVM-VMDK and flavor m1.tiny in the net_01. Associate floating ip.
6. Send ping from instances VM_1 and VM_2 to 8.8.8.8 or other outside ip.

Expected result

Pings should get a response

Check abilities to create and delete security group.

ID

nsxv_create_and_delete_secgroups

Description

Verifies that creation and removing security group works fine.

Complexity

advanced

Steps

1. Setup for system tests.
2. Log in to Horizon Dashboard.
3. Launch instance VM_1 in the tenant network net_02 with image TestVM-VMDK and flavor m1.tiny in the vcenter1 az.
4. Launch instance VM_2 in the tenant network net_02 with image TestVM-VMDK and flavor m1.tiny in the vcenter2 az.
5. Create security groups SG_1 to allow ICMP traffic.
6. Add Ingress rule for ICMP protocol to SG_1
7. Attach SG_1 to VMs
8. Check ping between VM_1 and VM_2 and vice verse
9. Create security groups SG_2 to allow TCP traffic 22 port. Add Ingress rule for TCP protocol to SG_2
10. Attach SG_2 to VMs.
11. ssh from VM_1 to VM_2 and vice verse.
12. Delete custom rules from SG_1 and SG_2.
13. Check ping and ssh aren't available from VM_1 to VM_2 and vice verse.
14. Add Ingress rule for ICMP protocol to SG_1.
15. Add Ingress rule for SSH protocol to SG_2.
16. Check ping between VM_1 and VM_2 and vice verse.
17. Check ssh from VM_1 to VM_2 and vice verse.
18. Attach VMs to default security group.
19. Delete security groups.
20. Check ping between VM_1 and VM_2 and vice verse.
21. Check SSH from VM_1 to VM_2 and vice verse.

Expected result

We should have the ability to send ICMP and TCP traffic between VMs in different tenants.

Verify that only the associated MAC and IP addresses can communicate on the logical port.

ID

nsxv_associated_addresses_communication_on_port

Description

Verify that only the associated MAC and IP addresses can communicate on the logical port.

Complexity

core

Steps

1. Setup for system tests.
2. Log in to Horizon Dashboard.
3. Launch 2 instances in each AZ.
4. Verify that traffic can be successfully sent from and received on the MAC and IP address associated with the logical port.
5. **Configure a new IP address from the subnet not like original one on the instance associated with the logical port.**
    - ifconfig eth0 down
    - ifconfig eth0 192.168.99.14 netmask 255.255.255.0
    - ifconfig eth0 up
6. Confirm that the instance cannot communicate with that IP address.
7. **Revert IP address. Configure a new MAC address on the instance associated with the logical port.**
    - ifconfig eth0 down
    - ifconfig eth0 hw ether 00:80:48:BA:d1:30
    - ifconfig eth0 up
8. Confirm that the instance cannot communicate with that MAC address and the original IP address.

Expected result

Instance should not communicate with new ip and mac addresses but it should communicate with old IP.

Check creation instance in the one group simultaneously.

ID

nsxv_create_and_delete_vms

Description

Verifies that system could create and delete several instances simultaneously.

Complexity

core

Steps

1. Setup for system tests.
2. Navigate to Project -> Compute -> Instances
3. Launch 5 instance VM_1 simultaneously with image TestVM-VMDK and flavor m1.tiny in vcenter1 az in default net_04.
4. All instance should be created without any error.
5. Launch 5 instance VM_2 simultaneously with image TestVM-VMDK and flavor m1.tiny in vcenter2 az in default net_04.
6. All instance should be created without any error.
7. Check connection between VMs (ping, ssh)
8. Delete all VMs from horizon simultaneously.

Expected result

All instance should be created and deleted without any error.

Check that environment support assigning public network to all nodes

ID

nsxv_public_network_to_all_nodes

Description

Verifies that checkbox “Assign public network to all nodes” works as designed.

Assuming default installation has been done with unchecked option “Assign public network to all nodes”.

Complexity

core

Steps

1. Setup for system tests.
2. Connect through ssh to Controller node. Run ‘ifconfig’.
3. Connect through ssh to compute-vmware node. Run ‘ifconfig’.
4. Redeploy environment with checked option Public network assignment -> Assign public network to all nodes.

5. Connect through ssh to Controller node. Run 'ifconfig'.
6. Connect through ssh to compute-vmware node. Run 'ifconfig'.

Expected result

Verify that before cluster redeployment with checked option only controllers have an IP from public network IP range, other nodes don't. Verify that after cluster redeployment all nodes have an IP from public IP range.

Verify LBaaS functionality

ID

nsxv_lbaas

Description

Setup LBaaS before test. Plugin requires attaching of an exclusive router to the subnet prior to provisioning of a load balancer. You can not use 22 port as port for VIP if you enable ssh access on edge.

Complexity

advanced

Steps

1. Setup for system tests.
2. - Create private network.
   - Create exclusive router (neutron router-create rexclusive –router_type exclusive).
   - Attach router to the external and private networks.
3. Create a security group that allows SSH (on port other than 22, e.g, 6022) and HTTP traffic.
4. - Create three instances based on TestVM-VMDK image.
   - Use created private network and security group.
5. Configure Load Balancer or several for different protocols. Here is example for TCP. * From Networks -> Load Balancers press button Add Pool. Example of settings: Provider vmwareedge Subnet subnet 10.130.0.0/24 Protocol TCP Load Balancing Method ROUND_ROBIN * Add members. Members: 10.130.0.3:22 10.130.0.4:22 10.130.0.5:22 * Add Monitor: Health Monitors PING delay:2 retries:2 timeout:2
6. Add VIP. Example of settings: Subnet subnet 10.130.0.0/24 Address 10.130.0.6 Floating IP 172.16.211.103 Protocol Port 6022 Protocol TCP Pool Name_from_step4 Session Persistence Type: ROUND_ROBIN Connection Limit -1
7. If LB with TCP was configured. Try to connect on Floating IP 172.16.211.103 using any TCP protocol. Use tool Mausezahn (in Ubuntu mz) or other.
8. If LB with HTTP was configured. Create a file index.html on instance. Like:

```
<!DOCTYPE html>
<html>
<body>
  Hi
</body>
</html>
```

Make on instances: while true; do { echo -e 'HTTP/1.1 200 OK\r\n'; cat index.html; } | sudo nc -l -p 80; done
Generate HTTP traffic on VIP floating IP.

Script to send http GET requests in parallel:

```
#!/bin/bash

LIMIT=100
for ((a=1; a <= LIMIT ; a++)) ;do
  curl http://172.16.211.127/ &
done
```

9. • Change Load Balancing Method to SOURCE_IP
   • Generate traffic.
10. • Delete one instance from Members.
    • Generate traffic.
11. • Add this member again.
    • Generate traffic.

Expected result

All steps passed without errors.

Deploy cluster with enabled SpoofGuard

ID

nsxv_spoofguard

Description

Nsxv spoofguard component is used to implement port-security feature. If a virtual machine has been compromised, the IP address can be spoofed and malicious transmissions can bypass firewall policies. http://pubs.vmware.com/NSX-62/topic/com.vmware.ICbase/PDF/nsx_62_admin.pdf p.137

Complexity

core

Steps

1. Deploy cluster with enabled SpoofGuard.
2. Run OSTF.
3. Setup spoofguard:

- In the vSphere Web Client, navigate to Networking & Security -> SpoofGuard.
- Click the Add icon.
- Type a name for the policy.
- Select Enabled or Disabled to indicate whether the policy is enabled.
- For Operation Mode, select Automatically Trust IP Assignments on Their First Use
- Click Allow local address as valid address in this namespace to allow local IP addresses in your setup. When you power on a virtual machine and it is unable to connect to the DHCP server, a local IP address is assigned to it. This local IP address is considered valid only if the SpoofGuard mode is set to Allow local address as valid address in this namespace. Otherwise, the local IP address is ignored.
- Click Next.
- To specify the scope for the policy, click Add and select the networks, distributed port groups, or logical switches that this policy should apply to. A port group or logical switch can belong to only one SpoofGuard policy.
- Click OK and then click Finish.

4. Run OSTF

Expected result

All OSTF test cases should be passed besides exceptions that are described in Limitation section of Test plan.

Deploy cluster with KVM virtualization

ID

nsxv_kvm_deploy

Description

Verify that nodes with compute-vmware role could be deployed in KVM.

Complexity

core

Steps

1. Create cluster based on KVM.
2. Add controller and compute-vmware nodes.
3. Deploy environment.

Expected result

Environment has been deployed successfully.

Check that metadata server is not available with disabled nsxv_metadata_initializer

ID

nsxv_metadata_mgt_disabled

Description

Test case verifies option nsxv_metadata_initializer in disabled state.

Complexity

core

Steps

1. Configure cluster.
2. Install fuel and nsxv plugin.
3. Configure nsxv plugin. Uncheck 'Init metadata infrastructure'.
4. Launch instance and run command from it:

```
wget -O - 169.254.169.254
```

Expected result

Options about metadata are in hide state. Request should return:

```
Connecting to 169.254.169.254 (169.254.169.254:80)
wget: download timed out!
```

Check availability metadata server in public network

ID

nsxv_metadata_listen_public

Description

Test case verifies option nsxv_metadata_listen in public state.

Complexity

core

Steps

1. Configure cluster.
2. Install fuel and nsxv plugin.
3. Configure nsxv plugin. In field 'Which network will be used to access the nova-metadata' choose 'Public network'. Manually specify the IP address, network mask and default route for the proxy metadata router.
4. Launch instance and run command from it:

```
wget -O - 169.254.169.254
```

Expected result

'Init metadata infrastructure' is checked by default. Request should return:

```
Connecting to 169.254.169.254 (169.254.169.254:80)
1.0
2007-01-19
2007-03-01
2007-08-29
2007-10-10
2007-12-15
2008-02-01
2008-09-01
2009-04-04
```

Check availability metadata server in management network

ID

nsxv_metadata_listen_management

Description

Test case verifies option nsxv_metadata_listen in management state.

Complexity

core

Steps

1. Configure cluster.
2. Install fuel and nsxv plugin. Configure nodes interfaces. Connect third interface (enp0s5) to Management.
3. Configure nsxv plugin. In field 'Which network will be used to access the nova-metadata' choose 'Management network'.
4. Launch instance and run command from it:

```
wget -O - 169.254.169.254
```

Expected result

'Init metadata infrastructure' is checked by default. Options about metadata are in hide state. Request should return:

```
Connecting to 169.254.169.254 (169.254.169.254:80)
1.0
2007-01-19
2007-03-01
2007-08-29
2007-10-10
2007-12-15
2008-02-01
2008-09-01
2009-04-04
```

Verify that nsxv driver configured properly with enabled SSL and custom certificates for access nova-metadata

ID

nsxv_config_ok_metadata_custom_certificate

Description

Need to check that nsxv plugin can access nova-metadata with certificate.

Complexity

advanced

Steps

1. Install NSXv plugin.
2. Enable plugin on tab Networks -> NSXv plugin.
3. Fill the form with corresponding values.
4. Unset checkbox “Metadata insecure”.
5. Generate certificate and upload it into field ‘Certificate for metadata proxy’.
6. Upload key into field ‘Private key’(‘Private key for metadata certificate’). Set passphrase.
7. Add to field ‘Metadata allowed ports’ value ‘443,8775’.
8. Deploy cluster.
9. Run OSTF.
10. Launch instance and run command from it:

    wget –no-check-certificate -O - https://169.254.169.254

Expected result

Check that nsx.ini on controller nodes is properly configured. Connection to nova-metadata is established. Request should return:

```
Connecting to 169.254.169.254 (169.254.169.254:80)
1.0
2007-01-19
2007-03-01
2007-08-29
2007-10-10
2007-12-15
2008-02-01
2008-09-01
2009-04-04
```

Verify that nsxv driver configured properly with enabled SSL and self-signed certificates for access nova-metadata

ID

nsxv_config_ok_metadata_self_signed_certificate

Description

Need to check that nsxv plugin can access nova-metadata with certificate.

Complexity

advanced

Steps

1. Install NSXv plugin.
2. Enable plugin on tab Settings -> NSXv plugin.
3. Fill the form with corresponding values.
4. Unset checkbox “Metadata insecure”.
5. Deploy cluster.
6. Run OSTF.
7. Launch instance and run command from it:

```
wget --no-check-certificate -O - https://169.254.169.254
```

Expected result

Check that nsx.ini on controller nodes is properly configured. Connection to nova-metadata is established. Request should return:

```
Connecting to 169.254.169.254 (169.254.169.254:80)
1.0
2007-01-19
2007-03-01
2007-08-29
2007-10-10
2007-12-15
2008-02-01
2008-09-01
2009-04-04
```

Verify that instances could be launched on enabled compute host

ID

nsxv_disable_hosts

Description

Check instance creation on enabled cluster.

Complexity

core

Steps

1. Setup cluster with 3 controllers and cinder-vmware + compute-vmware role.

2. Assign instances in each az.
3. Disable one of compute host with vCenter cluster (Admin -> Hypervisors).
4. Create several instances in vcenter az.
5. Check that instances were created on enabled compute host (vcenter cluster).
6. Disable second compute host with vCenter cluster and enable first one.
7. Create several instances in vcenter az.
8. Check that instances were created on enabled compute host (vcenter cluster).

Expected result

All instances work fine.

Check that settings about new cluster are placed in neutron config

ID

nsxv_smoke_add_compute

Description

Adding compute-vmware role and redeploy cluster with NSXv Plugin has effect in neutron configs.

Complexity

core

Steps

1. Upload the plugin to master node.
2. Create cluster and configure NSXv for that cluster.
3. Provision three controller node.
4. Deploy cluster with plugin.
5. Get configured clusters morefid from neutron config.
6. Add node with compute-vmware role.
7. Redeploy cluster with new node.
8. Get new configured clusters morefid from neutron config.
9. Check new cluster added in neutron config.

Expected result

Clusters are reconfigured after compute-vmware has been added.

Fuel create mirror and update core repos on cluster with NSXv plugin

ID

nsxv_update_core_repos

Description

Fuel create mirror and update core repos in cluster with NSXv plugin

Complexity

core

Steps

1. Setup for system tests
2. **Log into controller node via Fuel CLI and get PID of services which were launched by plugin and store them:** *ps ax | grep neutron-server*
3. **Launch the following command on the Fuel Master node:** *fuel-mirror create -P ubuntu -G mos ubuntu*
4. **Run the command below on the Fuel Master node:** *fuel-mirror apply -P ubuntu -G mos ubuntu –env <env_id> –replace*
5. **Run the command below on the Fuel Master node:** *fuel –env <env_id> node –node-id <node_ids_separeted_by_coma> –tasks setup_repositories* And wait until task is done.
6. Log into controller node and check plugins services are alive and their PID are not changed.
7. Check all nodes remain in ready status.
8. Rerun OSTF.

Expected result

Cluster (nodes) should remain in ready state. OSTF test should be passed on rerun

Failover

Verify that it is not possible to uninstall of Fuel NSXv plugin with deployed environment.

ID

nsxv_uninstall_negative

Description

It is not possible to remove plugin while at least one environment exists.

Complexity

smoke

Steps

1. Install NSXv plugin on master node.
2. Create a new environment with enabled plugin.
3. Try to delete plugin via cli from master node:

```
fuel plugins --remove nsxv==2.0.0
```

Expected result

Alert: "400 Client Error: Bad Request (Can't delete plugin which is enabled for some environment.)" should be displayed.

Shutdown primary controller and check plugin functionality.

ID

nsxv_shutdown_controller

Description

Check plugin functionality after shutdown primary controller.

Complexity

core

Steps

1. Log in to Fuel with preinstalled plugin and deployed enviroment with 3 controllers.
2. Log in to Horizon.
3. Create VM and check connectivity to outside world from VM.
4. Shutdown primary controller.
5. Ensure that VIPs are moved to other controller.
6. Ensure connectivity to outside world from created VM.
7. Create a new network and attach it to router.
8. Create a VM with new network and check network connectivity.

Expected result

Networking is working correct after failure of primary controller.

Check cluster functionality after reboot vcenter.

ID

nsxv_reboot_vcenter

Description

Verifies that system functionality is ok when vcenter has been rebooted.

Complexity

core

Steps

1. Log in to Fuel with preinstalled plugin and deployed enviroment.
2. Log in to Horizon.
3. Launch instance VM_1 with image TestVM-VMDK and flavor m1.tiny.
4. Launch instance VM_2 with image TestVM-VMDK and flavor m1.tiny.
5. Check connection between VMs, send ping from VM_1 to VM_2 and vice verse.
6. Reboot vcenter:

```
vmrun -T ws-shared -h https://localhost:443/sdk -u vmware -p pass
reset "[standard] vcenter/vcenter.vmx"
```

7. Check that controller lost connection with vCenter.
8. Wait for vCenter.
9. Ensure that all instances from vCenter displayed in dashboard.
10. Ensure connectivity between vcenter1's and vcenter2's VM.
11. Run OSTF.

Expected result

Cluster should be deployed and all OSTF test cases should be passed. ping should get response.