# Test Plan for Midonet MEM 3.0.0 Fuel Plugin

[Test Plan for Midonet MEM 3.0.0 Fuel Plugin](#h.gjdgxs)

[Revision history](#h.30j0zll)

[Midonet MEM Plugin](#h.2et92p0)

[Developer’s specification](#h.tyjcwt)

[Limitations](#h.3dy6vkm)

[Test strategy](#h.1t3h5sf)

[Acceptance criteria](#h.4d34og8)

[Test environment, infrastructure and tools](#h.2s8eyo1)

[Product compatibility matrix](#h.17dp8vu)

[Type of testing](#h.3rdcrjn)

[Deploy 6 Nodes Midonet MEM Gre](#h.fsfv744s6cv8)

[Deploy 6 Nodes Midonet MEM VxLAN](#h.sk4de99toqze)

[System testing](#h.nfhdkgif41tf)

[Install plugin and deploy environment](#h.35nkun2)

[Modifying env with enabled plugin (removing/adding controller nodes)](#h.1ksv4uv)

[Modifying env with enabled plugin (removing/adding compute node)](#h.44sinio)

[Uninstall of plugin with deployed environment](#h.2jxsxqh)

[Uninstall of plugin](#h.z337ya)

[Appendix](#h.4i7ojhp)

# Revision history

|  |  |  |  |
| --- | --- | --- | --- |
| **Version** | **Revision date** | **Editor** | **Comment** |
| 0.1 | 23.01.2015 | Irina Povolotskaya  (ipovolotskaya@mirantis.com) | Created the template structure. |
| 0.2 | 21.01.2016 | Carmela Rubinos  (carmela@midokura.com) | Filled template for MidoNet MEM Fuel Plugin for Fuel 7.0 |

# Midonet MEM Plugin

Midonet MEM is the Enterprise version for the MidoNet network virtualization software for Infrastructure-as-a-Service (IaaS) clouds. This module provides the puppet manifests to install all the components to deploy MidoNet MEM in a production environment as well as the Midonet Manager (GUI) for enhanced manageability. You will need Midonet MEM credentials to install MidoNet MEM version. Midonet MEM represents an alternative to Neutron’s ml2 Open vSwitch plugin.

## **Developer’s specification**

MEM plugin’s repo [1] contains the developer’s specification. Midonet MEM Fuel Plugin reviews are available in [2].

## **Limitations**

Midonet MEM Fuel Plugin version 3.0.0 has been developed for Fuel 7.0 to enable Enterprise Midonet on OpenStack deployments of Ubuntu 14.04.

Midonet MEM Fuel Plugin specific requirements include the creation of special node roles (NSDB and Gateway) which are not part of vanilla Fuel 6.1 and Fuel 7.0, so it needs to be manually added with the Fuel CLI. See [1] and [2] for more documentation.

# Test strategy

Midonet MEM Fuel Plugin replaces Neutron’s ml2 Open vSwitch networking plugin in an OpenStack deployment. Therefore, the tests that should pass to consider the plugin installation was successful are the ones related to networking/neutron. Regarding the Midonet Manager (GUI) sanity checks will be performed to ensure it is correctly installed.

In addition to the OSTF tests, we plan to run the following list of networking-related tests of Tempest using the Mirantis OpenStack (MOS) Tempest Runner [3].

API tests:

1. tempest.api.network.test\_networks
2. tempest.api.network.test\_networks\_negative
3. tempest.api.network.test\_ports
4. tempest.api.network.test\_routers.RoutersTest.test\_add\_multiple\_router\_interfaces
5. tempest.api.network.test\_routers.RoutersTest.test\_add\_remove\_router\_interface\_with\_port\_id
6. tempest.api.network.test\_routers.RoutersTest.test\_add\_remove\_router\_interface\_with\_subnet\_id
7. tempest.api.network.test\_routers.RoutersTest.test\_create\_router\_setting\_tenant\_id
8. tempest.api.network.test\_routers.RoutersTest.test\_create\_show\_list\_update\_delete\_router
9. tempest.api.network.test\_routers.RoutersTest.test\_update\_router\_admin\_state
10. tempest.api.network.test\_routers.RoutersTest.test\_update\_router\_reset\_gateway\_without\_snat
11. tempest.api.network.test\_routers.RoutersTest.test\_update\_router\_set\_gateway
12. tempest.api.network.test\_routers.RoutersTest.test\_update\_router\_set\_gateway\_with\_snat\_explicit
13. tempest.api.network.test\_routers.RoutersTest.test\_update\_router\_set\_gateway\_without\_snat
14. tempest.api.network.test\_routers.RoutersTest.test\_update\_router\_unset\_gateway
15. tempest.api.network.test\_routers\_negative
16. tempest.api.network.test\_security\_groups
17. tempest.api.network.test\_security\_groups\_negative
18. tempest.api.network.test\_floating\_ips.FloatingIPTestJSON.test\_create\_floating\_ip\_specifying\_a\_fixed\_ip\_address
19. tempest.api.network.test\_floating\_ips.FloatingIPTestJSON.test\_create\_list\_show\_update\_delete\_floating\_ip

Scenario Tests (when there is external connectivity):

1. tempest.scenario.test\_network\_basic\_ops
2. tempest.scenario.test\_network\_advanced\_server\_ops
3. tempest.scenario.test\_security\_groups\_basic\_ops

## **Acceptance criteria**

The set of Fuel OSTF tests that are related to the networking are:

* Request list of networks
* Check network parameters
* Check network connectivity from instance via floating IP

## **Test environment, infrastructure and tools**

Minimum requirement for testing Midonet MEM Fuel Plugin (without HA capabilities) include one Controller, one Compute, one MidoNet Gateway and one NSDB (that can be collocated in the same node as the Controller role, but it that case the RAM memory should be ≥ 6GB):

* CPU: 64-bit x86, quad core or above
* Memory: ≥ 4 GB RAM
* HDD: ≥ 30GB (available free disk space)
* NIC: 2 x ≥ 1Gbit

For testing HA capabilities, the needs are 2 Controller nodes, 2 Computes, 2 MidoNet HA Gateways and 3 NSDB.

First tests will be done on the minimal deployment. Then the tests will increase the number of nodes to tackle:

* HA increasing the number of controller nodes
* Multiple compute nodes
* HA for the NSDB nodes
* HA for MidoNet Gateway nodes
* External connectivity

## **Product compatibility matrix**

Midonet plugin is developed for Fuel 7.0 therefore it will be tested against the only one supported operating system where to deploy OpenStack: Ubuntu 14.04.

The Midonet MEM supported version v1.9.5.

# Type of testing

## **Deploy 6 Nodes Midonet MEM Gre**

|  |  |
| --- | --- |
| Test Case ID | deploy\_6\_node\_midonet\_gre |
| Description | 2 Controllers+collocated NSDB, 2 Computes and 2 HA MidoNet Gateways (and one NSDB collocated in one of them). |
| Steps | 1. Upload plugin to the master node 2. Install plugin 3. Add NSDB and Gateway roles as described in the documentation. 4. Create environment with GRE and enabled plugin in Fuel UI following the operations guide. 5. Add 3 nodes with Controller, 3 NSDB roles, 2 nodes with Compute and and 2 nodes with HA MidoNet Gateway role. 6. Apply network settings 7. Run network verification 8. Deploy the cluster 9. Check plugin health using cli 10. Check Midonet Manager health 11. Run OSTF 12. Run MOS-Tempest-Runner |
| Expected Result | Plugin is installed successfully, cluster is created,network verification and OSTF are passed (HA tests should not run), all plugin services are enabled, MOS-Tempest-Runner passed, Midonet Manager is installed and works correctly. |

## **Deploy 6 Nodes Midonet MEM VxLAN**

|  |  |
| --- | --- |
| Test Case ID | deploy\_6\_node\_midonet\_vxlan |
| Description | 2 Controllers+collocated NSDB, 2 Computes and 2 HA MidoNet Gateways (and one NSDB collocated in one of them). |
| Steps | 1. Upload plugin to the master node 2. Install plugin 3. Add NSDB and Gateway roles as described in the documentation. 4. Create environment with VxLAN and enabled plugin in Fuel UI following the operations guide. 5. Add 3 nodes with Controller, 3 NSDB roles, 2 nodes with Compute and and 2 nodes with HA MidoNet Gateway role. 6. Apply network settings 7. Run network verification 8. Deploy the cluster 9. Check plugin health using cli 10. Check Midonet Manager health 11. Run OSTF 12. Run MOS-Tempest-Runner |
| Expected Result | Plugin is installed successfully, cluster is created,network verification and OSTF are passed (HA tests should not run), all plugin services are enabled, MOS-Tempest-Runner passed, Midonet Manager is installed and works correctly. |

# System testing

## **Install plugin and deploy environment**

|  |  |
| --- | --- |
| Test Case ID | install\_plugin\_deploy\_env |
| Steps | 1. Upload plugin to the master node 2. Install plugin using Fuel CLO 3. Ensure that plugin is installed successfully using Fuel CLI 4. Create environment with enabled plugin in Fuel UI 5. Add 3 nodes with Controller role and 1 node with Compute and another role 6. Apply network settings 7. Run network verification 8. Deploy the cluster 9. Check plugin health using cli 10. Run OSTF |
| Expected Result | *Plugin is installed successfully, cluster is created,* network verification and OSTF are passed, and all plugin services are enabled and work as expected. |

## **Modifying env with enabled plugin (removing/adding controller nodes)**

|  |  |
| --- | --- |
| Test Case ID | modify\_env\_with\_plugin\_remove\_add\_controller |
| Steps | 1. Upload plugin to the master node 2. Install plugin 3. Ensure that plugin is installed successfully using cli 4. Create environment with enabled plugin in fuel ui 5. Add 3 nodes with Controller role and 1 node with Compute and another role 6. Apply network settings 7. Run network verification 8. Deploy the cluster 9. Check plugin services using cli 10. Run OSTF 11. Remove 1 nodes with Controller role   /\*remove node, where plugin’s services available, to ensure that according to ha mode all plugins resources will be replaced and available on another live node and continue to work as expected\*/   1. Re-deploy cluster 2. Check plugin services using cli 3. Run OSTF 4. Add 1 new node with Controller role 5. Re-deploy cluster 6. Check plugin services using cli 7. Run OSTF |
| Expected Result | *Plugin is installed successfully, cluster is created, network verification and OSTF are passed, and all plugin services are enabled after migration in ha mode and worked as expected after modifying of environment.* |

## **Modifying env with enabled plugin (removing/adding compute node)**

|  |  |
| --- | --- |
| Test Case ID | modify\_env\_with\_plugin\_remove\_add\_compute |
| Steps | 1. Upload plugin to the master node 2. Install plugin 3. Ensure that plugin is installed successfully using cli 4. Create environment with enabled plugin in fuel ui 5. Add 3 nodes with Controller role and 2 nodes with compute roles and 1 another role 6. Apply network settings 7. Run network verification 8. Deploy the cluster 9. Check plugin services using cli 10. Run OSTF 11. Remove 1 compute node 12. Re-deploy cluster 13. Check plugin services using cli 14. Run OSTF 15. Add 1 compute node 16. Re-deploy cluster 17. Check plugin services using cli 18. Run OSTF |
| Expected Result | *Plugin is installed successfully, cluster is created, network verification and OSTF are passed, and all plugin services are enabled and worked as expected after modifying of environment.* |

## **Uninstall of plugin with deployed environment**

|  |  |
| --- | --- |
| Test Case ID | uninstall\_plugin\_with\_deployed\_env |
| Steps | 1. install plugin 2. deploy environment with enabled plugin functionality 3. run ostf 4. try to delete plugin and ensure that present in cli alert: "400 Client Error: Bad Request (Can't delete plugin which is enabled for some environment.)" 5. remove environment 6. remove plugin 7. check that it was successfully removed |
| Expected Result | *Plugin was installed successfully. Alert is present when we trying to delete plugin which is attached to enabled environment. When environment was removed, plugin is removed successfully too.* |

## **Uninstall of plugin**

|  |  |
| --- | --- |
| Test Case ID | uninstall\_plugin |
| Steps | 1. install plugin 2. check that it was installed successfully 3. remove plugin via Fuel CLI 4. check that it was successfully removed |
| Expected Result | *Plugin was installed and then removed successfully* |

# Appendix

Provide any links to external resources or documentation here.

|  |  |
| --- | --- |
| **№** | **Resource title** |
| 1 | <https://github.com/openstack/fuel-plugin-midonet/tree/master> |
| 2 | <https://review.openstack.org/#/q/project:openstack/fuel-plugin-midonet,n,z> |
| 3 | <https://github.com/Mirantis/mos-tempest-runner> |