Test Plan for MidoNet Fuel Plugin v3.0.1

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
0.3	29.02.2016	Samir Ibradžić (samir@midokura.com)	Add changes related to MidoNet Fuel Plugin v3.0.1, general fixes
0.3a	29.06.2016	Samir Ibradžić (samir@midokura.com)	Describe extra limitations and add fuel_create_mirror_update_repos

Introduction

MidoNet Fuel Plugin

MidoNet MEM is the Enterprise version for the MidoNet network virtualization software for Infrastructure-as-a-Service (IaaS) clouds. This Fuel plugin module provides the puppet manifests to install all the components to deploy MidoNet MEM in a production environment. You will need MidoNet MEM debian package repository credentials to install MidoNet MEM version. MidoNet MEM represents an alternative to Neutron's default OpenvSwitch plugin.

Developer's specification

MidoNet MEM plugin source code repository [1] contains the developer's specification. MidoNet MEM Fuel Plugin reviews are available in [2].

Limitations

MidoNet MEM Fuel Plugin version 3.0.1 has been developed for Fuel 7.0 to enable Enterprise MidoNet on OpenStack deployments on top of Ubuntu 14.04 platform.

MidoNet MEM Fuel Plugin specific requirements include the creation of special node roles (Network State Database - NSDB and Gateway) which are not part of standard Fuel 7.0 release, so these roles needs to be added by either MidoNet Fuel Plugin rpm package post-installation scripts or manually by user using CLI. See [1] and [2] for more details.

In its current version MidoNet MEM Fuel Plugin has some limitations regarding node count scalability for NSDB (Network State Database) and OpenStack Controller role nodes. Once number of nodes with such roles have been determined on initial deployment, it can not be changed. Compute role nodes are not affected by this limitation, current plugin version supports Compute scalability.

Test strategy

MidoNet MEM Fuel Plugin replaces Neutron default OpenvSwitch networking plugin in a typical OpenStack deployment based on Neutron. Therefore, the goal of the tests is to to ensure that plugin installation was successful as well to ensure that OpenStack Neutron networking supported by MidoNet is fully functional. As Fuel OSTF health check provide a solid base for testing most of MidoNet functionality, we make sure that all OSTF tests are passing correctly.

Acceptance criteria

The set of Fuel OSTF tests whose successful execution fully depends on MidoNet are:

1. Sanity tests:
    a. Request list of networks
2. Functional tests:
    a. Check network connectivity from instance via floating IP
    b. Check network parameters
    c. Launch instance with file injection

Although the list is somewhat short, executing these functional tests successfully will ensure that a larger set of MidoNet functionality is operating correctly. Just to mention few: MidoNet NSDB (any test listed above), MidoNet REST API (any test listed above), MidoNet Neutron plugin (any test listed above), MidoNet networking agents (tests 2.a and 2.b; on controllers, computes as well as gateways), MidoNet BGP gateway including Floating IPs (tests 2.a and 2.c).

Test environment, infrastructure and tools

Minimum requirement for testing MidoNet MEM Fuel Plugin in non-HA setup include one OpenStack Controller, one Compute, one MidoNet Gateway and one NSDB node. The base hardware specification of those nodes are:

- CPU: 64-bit x86, quad core or above
- Memory: ≥ 8 GB RAM
- HDD: ≥ 30GB
- NIC: 2 x ≥ 1Gbit

For testing HA capabilities, the minimum needs are 2 Controller nodes, 2 Computes, MidoNet HA Gateway and 3 NSDB. Same base node hardware specification applies. All tests nodes can be both physical hardware as well as the virtual machines (confirmed working on libvirt KVM hypervisor), of same base hardware specification.

Initial tests will be done by simply installing and removing Fuel MidoNet plugin. Next test step is to verify minimal but fully functional deployments with both GRE and VxLAN tunneling, followed by compute scalability test. Finally, full HA scenario test will be done resulting with the following nodes to be deployed and tested:

- 3 NSDB nodes (native HA cluster)
- 3 OpenStack controllers (corosync HA)
- 2 Compute nodes
- 1 Telemetry MongoDB node
- 1 Storage-Cinder node
- 1 MidoNet BGP gateway

An external connectivity tests that verify MidoNet BGP gateway have a special requirement. As of v3.0.1 release of MidoNet Fuel Plugin, the only Neutron network gateway mode supported by the plugin is BGP gateway. That means that the test environment has to have at least one of the external BGP peers available as the endpoint for passing all Floating IP (FIP) traffic between the deployed OpenStack cloud instances and the external network(s). These BGP peers are usually available for production or data-centre ISP environments, so for the sake of supporting BGP tests under lab or proof-of-concept conditions we are providing instructions [3] on how to set up a “fake” BGP peer that supports the OSTF test scenarios that require FIP. To make the testing easier, we assign BGP gateway IP and AS addresses so that they match the values from BGP peer setup example [3]:

- BGP IP subnet: **10.88.88.0/30**
- VyOS BGP peer IP address: **10.88.88.1**
- VyOS BGP peer AS number: **65535**
- MidoNet BGP gateway IP address: **10.88.88.2**
- MidoNet BGP gateway AS number: **12345**

- Floating IP subnet: **200.200.200.0/24**

These parameters can be set in the Neutron MidoNet plugin section via Fuel WEB interface.

Product compatibility matrix

MidoNet plugin is developed for Fuel 7.0 therefore it will be tested against the only supported operating system, Ubuntu 14.04. The supported MidoNet MEM version is v1.9.x (any latest 1.9 bug-fix release). For the test BGP peer OS VyOS 1.1.7 or later can be used.

Basic System and Deployment test scenarios

MidoNet Fuel Plugin installation cycle

Test Case ID	**inst_uninst_plugin**
Steps	1. install plugin using Fuel cli 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. Running time: ~ 2 m

Install plugin and deploy environment

Test Case ID	**inst_plugin_deploy_env**
Steps	1. install plugin using Fuel cli 2. check if plugin is installed successfully 3. create environment with enabled plugin in Fuel WEB UI 4. add 1x NSDB, 1x Controller and 1x Compute role node 5. run network verification 6. deploy the cluster 7. check plugin health using cli 8. run all OSTF “Sanity tests”
Expected Result	Plugin is installed successfully, All OSTF “Sanity tests” passed, all plugin services are enabled and work as expected. Running time: ~ 1h 30m

Uninstall of plugin with deployed environment

Test Case ID	**uninst_plugin_deployed_env**
Steps	1. do all steps as in **inst_plugin_deploy_env** 2. try to delete midonet-fuel-plugin using Fuel cli 3. make sure that midonet-fuel-plugin was not removed 4. in Fuel UI “Delete Environment” set up in 1st step 5. try to delete midonet-fuel-plugin using Fuel cli 6. check that plugin was successfully removed
Expected Result	Plugin was installed successfully. Alert is present when we trying to delete plugin when attached to enabled environment: `400 Client Error: Bad Request (Can't delete plugin which is enabled for some environment.)` When environment was removed, plugin is removed successfully too.

Fuel create mirror and update core repos

Test Case ID	**fuel_create_mirror_update_repos**

Steps	1. do all steps as in **inst_plugin_deploy_env**
	2. Take note of node IDs and update core repos using Fuel cli:
	`fuel-createmirror -M`
	`fuel --env <env_ID> node --node-id <node_IDs> --tasks upload_core_repos`
	3. verify services of all nodes and ensure service PIDs are unchanged
	4. verify if status of all nodes is ‘ready’ using Fuel cli: `fuel nodes`
	5. run all OSTF tests “Sanity tests”

Expected Result	Plugin is installed successfully at the Fuel Master node as indicated in CLI. Cluster is created and network check is passed. Plugin is enabled and configured in the Fuel Web UI. OSTF "Sanity tests" are passing. Environment is deployed successfully. Plugin's services shouldn't be restarted after above tasks were executed. If they are restarted as some exception, this information should be added to plugin's User Guide. Cluster (nodes) should remain in ready state. Re-run of OSTF "Sanity tests" are passing. Running time: ~1h 30m

Functional Test scenarios

MidoNet MEM 4-node GRE tunnel

Test Case ID	**simple_4_node_midonet_gre**

Description	1x NSDB, 1x Controller, 1x Compute and 1x MidoNet GW
Steps	1. make sure plugin is installed 2. create environment with GRE and enabled plugin in Fuel UI following the operations guide 3. add 1x NSDB node, 1x Controller nodes, 1x Compute node and 1x MidoNet Gateway node 4. deploy the cluster 5. make sure external BGP test peer [3] is set-up and available 6. run all OSTF “Sanity tests” and all “Functional tests” other than two tests beginning with “Create volume ...”
Expected Result	Plugin is installed successfully, cluster is created, network verification and OSTF tests passed (no HA, volume or Heat tests), all plugin services are enabled, stack works correctly. Running time: ~2h

MidoNet MEM 4-node VxLAN tunnel

Test Case ID	**simple_4_node_midonet_vxlan**
Description	1x NSDB, 1x Controller, 1x Compute and 1x MidoNet GW
Steps	1. make sure plugin is installed 2. create environment with VxLAN and enabled plugin in Fuel UI following the operations guide 3. add 1x NSDB node, 1x Controller nodes, 1x Compute node and 1x MidoNet Gateway node 4. deploy the cluster 5. make sure external BGP test peer [3] is set-up and available 6. run all OSTF “Sanity tests” and all “Functional tests” other than two tests beginning with “Create volume ...”
Expected Result	Plugin is installed successfully, cluster is created, network verification and OSTF tests passed (no HA, volume or Heat tests), all plugin services are enabled, stack works correctly. Running time: ~2h

Compute scalability

Test Case ID	**modify_compute_count**
Description	1x NSDB, 1x Controller, 1x initial Compute & 1x MidoNet GW
Steps	1. do all steps as in **simple_4_node_midonet_vxlan**, or simply continue from there 2. add 1 compute node 3. “Deploy Changes” in Fuel UI 4. run all OSTF “Sanity tests” and all “Functional tests” other than two tests beginning with “Create volume ...” 5. delete 1 Compute node in Fuel UI 6. “Deploy Changes” in Fuel UI again

	7. run all OSTF “Sanity tests” and all “Functional tests” other than two tests beginning with “Create volume ...”
Expected Result	Additional Compute role node successfully added to the environment, all OSTF “Sanity tests” passes, all “Functional tests” passes after node addition. Same set of tests passes after Compute node deletion, except “Sanity tests: Check that required services are running”. All plugin services are enabled, stack works correctly after Compute additions and removal. Running time: ~2h 30m

MidoNet Full HA

Test Case ID	**midonet_full_ha**
Description	3x NSDB, 3x Controller, 2x Compute, 1x Telemetry-MongoDB, 1x Storage-Cinder and 1x MidoNet GW roles
Steps	1. make sure plugin is installed 2. create environment with VxLAN and enabled plugin in Fuel UI following the operations guide 3. enable Ceilometer in the Fuel UI; Settings, Additional Components, Install Ceilometer, check, Save Settings 4. add 3x NSD, 3x Controller, 2x Compute, 1x Telemetry, 1x Storage - Cinder and 1x MidoNet Gateway node 5. deploy the cluster 6. make sure external BGP test peer [3] is set-up and available 7. run all available OSTF test except “Configuration tests”
Expected Result	HA cluster is successfully created, all OSTF tests except “Configuration tests” passed, all plugin services are enabled, stack works correctly. Running time: ~4h !

Appendix

Provide any links to external resources or documentation here.

№	Resources
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/openstack/fuel-plugin-midonet/blob/master/doc/content/bgp-peer.rst