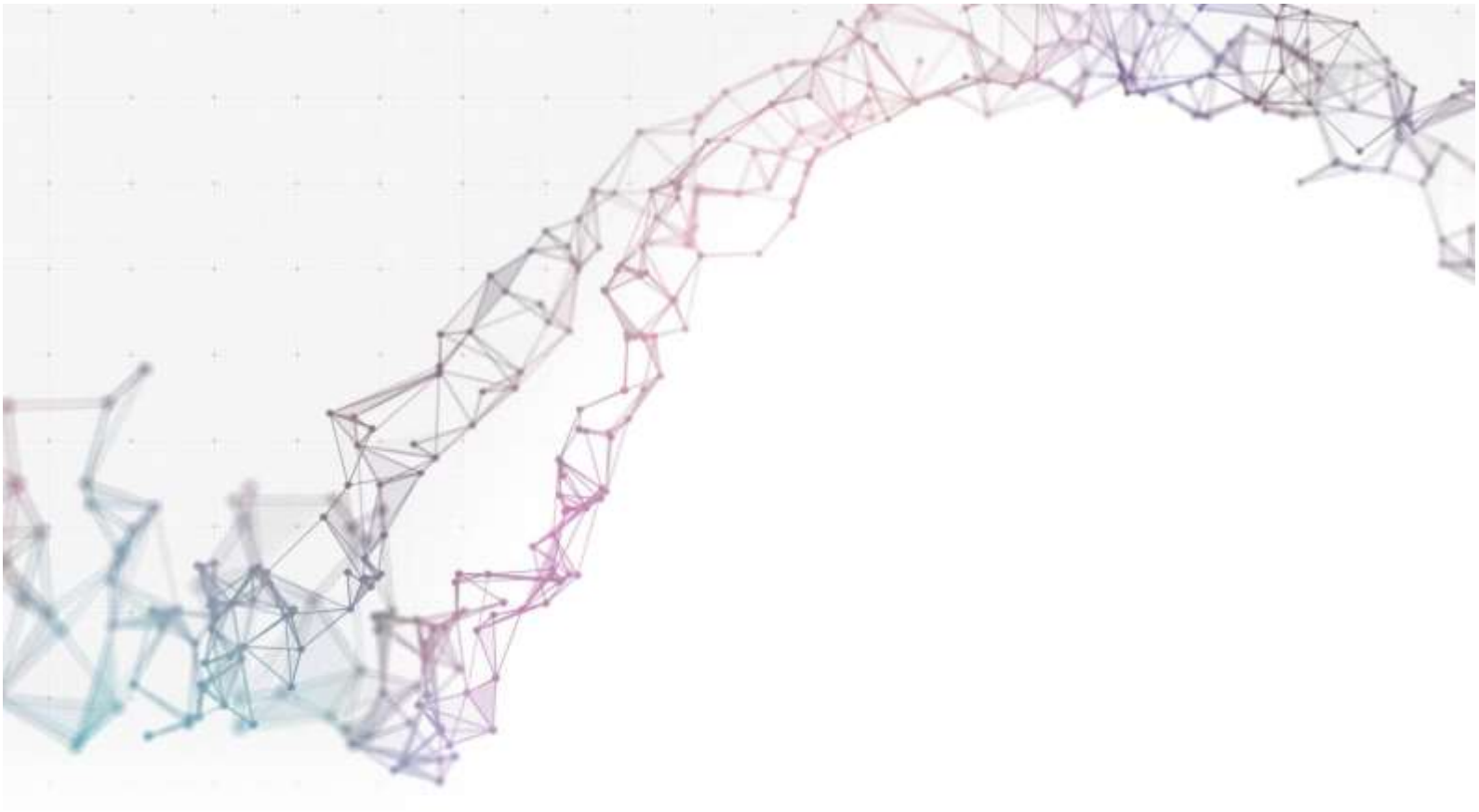

MIRANTIS

FUEL for OpenStack®

TEST PLAN

for

kafka-1.0-1.0.0-1.noarch.rpm

Mirantis OpenStack 9.0

Revision history

Version	Revision date	Editor	Comment
1.0	06.10.2016	Artem Minasyan (aminasyan@mirantis.com)	Initial revision.

Developer's specification

https://github.com/openstack/fuel-plugin-kafka

Test strategy

The test plan describes system and functional tests. These tests will be automated but tests of user interfaces will have to be done manually.

Acceptance criteria

Environment should be deployed.

Test environment and infrastructure

The Kafka plugin is installed on the Fuel master node. For controller nodes, it is recommended to deploy on hosts with at least 2 CPUs and 8G of RAM.

Product compatibility matrix

Issue	Version
Mirantis OpenStack	9.0, 9.1
Plugin Kafka	1.0.0

Functional testing

Check messages

Test Case ID	check_messages
Description	Verify that sending messages works correctly.
Prerequisites	Environment deployed with the plugin (deploy_plugin).
Steps	1. Create a topic : *bin/kafka-topics.sh --create --zookeeper localhost:2181 --replication-factor 1 --partitions 1 --topic test* 2. Send messages: *bin/kafka-console-producer.sh --broker-list localhost:9092 --topic test* 3. Start consumer: *bin/kafka-console-consumer.sh --zookeeper localhost:2181 --topic test --from-beginning*
Expected Result	*Check that quantity of sent messages equal to quantity of received messages.*

System testing

Install the plugin

Test Case ID	install_plugin
Description	Verify that the plugin can be installed
Steps	1. Copy the plugin to the Fuel master node using scp. 2. Connect to the Fuel master node using ssh. 3. Install the plugin using the fuel CLI. 4. Connect to the Fuel web UI. 5. Create a new environment using the Fuel UI Wizard. 6. Click on the Settings tab.
Expected Result	*The plugin presents in the Fuel UI.*

Deploy an environment with the plugin

Test Case ID	deploy_plugin
Description	Verify that the plugin can be deployed.
Prerequisites	Plugin is installed on the Fuel master node.
Steps	1. Connect to the Fuel web UI. 2. Create a new environment with the Fuel UI wizard with the default settings. 3. Click on the Settings tab of the Fuel web UI. 4. Select the plugin checkbox. 5. Click 'Deploy changes'. 6. After the end of deployment run OSTF.
Expected Result	*The environment is deployed successfully. OSTF tests pass successfully.*

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

Test Case ID	modify_env_with_plugin_remove_add_controller
Description	Verify that the env with plugin can scale (remove/add controller)
Prerequisites	Environment deployed with the plugin (deploy_plugin).
Steps	1. Copy the plugin to the Fuel Master node (please refer to the User Guide for more details). 2. Install the plugin. 3. Ensure that the plugin is installed successfully using CLI with running `fuel plugins --list` command in the Fuel CLI. 4. Create an environment with enabled plugin in the Fuel Web UI. 5. Add 3 nodes with Controller role and 1 node with Compute and another role. 6. Finalize environment configuration (e.g. networking, nodes interfaces). 7. Enable the plugin and configure it following the instructions from the Plugin Guide. 8. Run network verification check 9. Deploy the cluster. 10. Run OSTF 11. Remove 1 node with Controller role (*i.e. remove the primary Controller node which should have the lowest ID, where plugin's services are running to ensure that all plugins resources are migrated to another Controller node*). 12. Re-deploy the cluster. 13. Run OSTF 14. Add 1 new node with Controller role. 15. Re-deploy the cluster. 16. Run OSTF.
Expected Result	*Plugin is installed successfully at the Fuel Master node and the corresponding output appears in the CLI.* *Cluster is created and network verification check is passed.* *Plugin is enabled and configured in the Fuel Web UI.* *OSTF tests (Health Checks) are passed.* *Environment is deployed successfully.* *When adding/removing Controller node (where plugin-related services are run):*

	a. *all plugins resources are migrated to another Controller node* b. *the environment is redeployed successfully when adding/removing Controller node.*

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

Test Case ID	modify_env_with_plugin_remove_add_compute
Description	Verify that the env with plugin can scale (remove/add compute)
Prerequisites	Environment deployed with the plugin (deploy_plugin).
Steps	1. Add 1 node with the compute role 2. Re-deploy the cluster 3. Check the plugin services using cli 4. Run OSTF 5. Remove 1 node with the compute role 6. Re-deploy cluster 7. Check the plugin services using cli 8. Run OSTF
Expected Result	*OSTF tests pass successfully, and all the plugin services are running and worked as expected after each modification of the environment.*

Uninstall the plugin with deployed environment

Test Case ID	uninstall_plugin_with_deployed_env
Description	Verify that the plugin can delete with installed env.
Prerequisites	Environment deployed with the plugin (deploy_plugin).
Steps	1. 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.)" 2. Remove the environment.

	3. Remove the plugin. 4. Check that it was successfully removed
Expected Result	*Alert is present when we try to delete plugin which are attached to enabled environment. When the environment is removed, plugin is removed successfully too.*

Uninstall plugin

Test Case ID	uninstall_plugin
Description	Verify that the plugin successfully uninstalled.
Prerequisites	The plugin installed on the Fuel node (install_plugin).
Steps	1. Remove the plugin. 2. Check that they was successfully removed.
Expected Result	*Plugin is removed successfully*

Fuel create mirror and update (setup) of core repositories

Test Case ID	create_mirror
Description	Create a mirror on the existing cluster
Prerequisites	Environment deployed with the plugin (deploy_plugin).

Steps	1. Copy plugin to the Fuel master node. 2. Install the plugin 3. Ensure that the plugin is installed successfully using CLI with running fuel plugins --list command in the Fuel CLI. 4. Create an environment with enabled plugin in the Fuel Web UI. 5. Add 3 nodes with Controller role and 1 node with compute and another role. 6. Finalize environment configuration (e.g. networking, nodes interfaces). 7. Enable the plugin and configure it following the instructions from the Plugin Guide. 8. Run network verification check. 9. Deploy the cluster 10. Run OSTF 11. Go in cli through controller / compute / storage /etc nodes and get pid of services which were launched by plugin and store them. 12. Launch the following command on the Fuel Master node: fuel-createmirror -M 13. Launch the following command on the Fuel Master node: a) For MOS < 8.0: fuel --env <ENV_ID> node --node-id <NODE_ID1> <NODE_ID2> <NODE_ID_N> --tasks upload_core_repos b) For MOS 8.0: fuel --env <ENV_ID> node --node-id <NODE_ID1> <NODE_ID2> <NODE_ID_N> --tasks setup_repositories 14. Go to controller/plugin/storage node and check if plugin's services are alive and aren't changed their pid. 15. Check with fuel nodes command that all nodes are remain in ready status. 16. Run OSTF
Expected result	*Plugin is installed successfully at the Fuel Master node and the corresponding output appears in the CLI.Cluster is created and network verification check is passed.Plugin is enabled and configured in the Fuel Web UI.OSTF tests (Health Checks) are passed. Environment is deployed successfully.When adding/removing Compute node (where plugin-related services are run):all plugins resources are migrated to another Compute node the environment is re-deployed successfully when adding/removing Compute node.Plugin's services shouldn't be restarted after corresponding*

Appendix

№	Resource title
1	Blueprint: https://blueprints.launchpad.net/lma-toolchain/+spec/kafka-plugin-for-lma
2	Design spec link: https://drive.google.com/open?id=1TzXRUG8ZtpyTa_ckSpZcIvRDcu74dad8QjMBs1Zt-qs
3	Test report: https://docs.google.com/document/d/1Z5itnwfvf8TCg0vdTcbq7uwI-iUBbHHRrVNOJQzErZs/edit#