INSTALLATION RUNBOOK FOR Netronome Agilio OvS

Product Name:	Agilio OvS
Driver Version:	2.2-r4603
MOS Version:	8.0
OpenStack Version:	Liberty

Product Type: **Network Offload Driver**

1. Introduction
    1.1 Target Audience
2. Product Overview
3. Joint Reference Architecture
4. Physical and Logical Network Topology
5. Installation and Configuration
    5.1 Environment Preparation
    5.2 MOS Installation
        Setup the Fuel orchestration server
        Customize the Fuel orchestration server
        Installation steps on the Fuel master node
        Update the firmware (if necessary) on the Agilio compute nodes
        Create a new cloud
        Provision an OpenStack Cloud
        Activate Agilio acceleration on the compute nodes
            Passthrough acceleration
            VirtIO Relay acceleration
        Endpoint configuration
        Spawn instances
    5.3 Health Check Results
    5.4 Limitations
Appendix A: Health Check Results

Document History

Version	Revision Date	Description
0.1	04-08-2016	Initial Version
0.2	21-09-2016	Expanded Joint Reference Architecture Expanded Installation and Configuration Added Health Check Results

1. Introduction

This document is a detailed Deployment Guide for Netronome Agilio OvS, which integrates Mirantis OpenStack 8.0 Fuel with Agilio Network Adaptors for use as Compute hosts. It provides a reference architecture for deploying the solution and detailed installation instructions. The document also notes limitations of the driver and integration, describes tests performed during validation, and provides the results of these tests.

This document does not provide details on the Agilio OvS solution itself. For more information, please consult the Getting Started Guide, User's Guide, Programmer's Reference Manual and Release Notes in the Agilio OvS Software Release 2.2 section at https://support.netronome.com.

1.1 Target Audience

The target audience of this document is systems integration specialists aiming to install and use Agilio network adaptors in an OpenStack deployment. Familiarity with the Agilio OvS product, SR-IOV, VF passthrough, hugepages, vhost-user and the VirtIO Relay is highly recommended. Familiarity is assumed with Mirantis Fuel 8.0 and OpenStack Liberty.

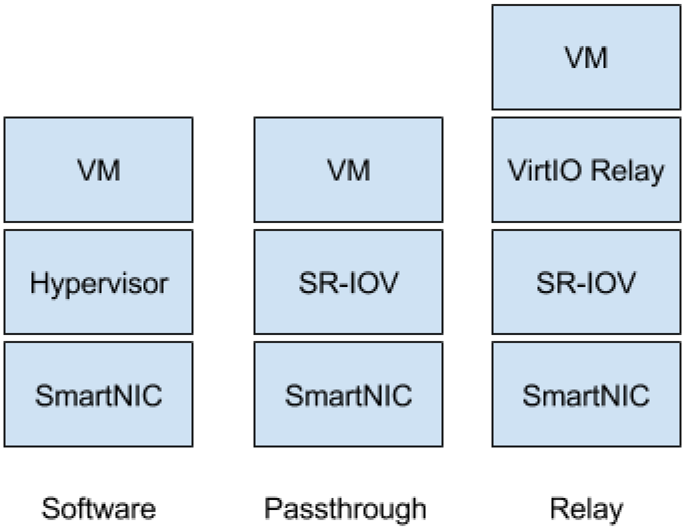
2. Product Overview

The Product Brief for Agilio OvS is located at:
https://www.netronome.com/media/redactor_files/PB_Agilio_OvS_SW.pdf

Mirantis OpenStack and Fuel provide a seamless deployment solution for an OpenStack Cloud utilizing Agilio OvS.

3. Joint Reference Architecture

The Agilio OvS Software Architecture White Paper is located at:
https://www.netronome.com/media/redactor_files/WP_Agilio_SW.pdf



Agilio OvS integrates with OpenStack on the compute node by providing three different plugging methods to connect the instances to the SmartNIC.

- Software: This is the classic mechanism where the the OvS softswitch processing is done on the hypervisor.
- Passthrough: This method provides a VM with an SR-IOV device using the IOMMU passthrough mechanism. Packet processing is done on the SmartNIC and the VM has low latency access to a PCI device.
- Relay: This method provides a compromise between the other two methods. Packet processing is done on the SmartNIC. A VirtIO Relay client runs on the hypervisor that connects to the VM. The VM provides a VirtIO device instead of a native PCI device. Live migration is supported with this method.

This solution adds support for Agilio OvS to the OpenStack OvS Mechanism driver, OpenStack Nova and OpenStack Neutron:

- OpenStack Nova and Neutron were modified to support two extra VIF plugging methods: Agilio Passthrough and Agilio Relay.
- The OvS Mechanism driver was extended to support three separate agents:
    - Standard OvS
    - Agilio Passthrough
    - Agilio Relay
- Depending on which agent is running on a compute node, Nova will employ the corresponding plugging code.

In addition, the Fuel Agent and mcollective were modified to recognise the Agilio OvS representative netdevs.

4. Physical and Logical Network Topology

This solution supports the physical and logical topology of the Open vSwitch ML2 plugin configured without security groups. Consult the Agilio OvS documentation to determine which types of overlay networks support acceleration. This guide will assume that Neutron with tunneling segmentation has been chosen as the network configuration.

5. Installation and Configuration

5.1 Environment Preparation

At least one Compute node with an Agilio adaptor is required.

5.2 MOS Installation

Set up the Fuel orchestration server

- Download Mirantis Fuel 8.0 from https://software.mirantis.com/releases/
- Download the Agilio OvS 2.2 Fuel tarball (agilio-ovs-2.2*-fuel.tar.gz) from the Agilio OvS Software Release 2.2 section at https://support.netronome.com.

Install Fuel on a VM or on an orchestration machine. Note: The standard Mirantis install steps are followed. The Fuel orchestration machine must be able to serve PXE boot images to clients attached to its network interface.

Verify that a normal OpenStack cloud can be installed by following the install guide: https://docs.mirantis.com/openstack/fuel/fuel-8.0/quickstart-guide.html

Customize the Fuel orchestration server

Extract the agilio-ovs-2.2*-fuel.tar.gz tarball and copy the contents in the extracted directory agilio-ovs-2.2*-fuel/ to the /root/ directory of the Fuel orchestration server.

Contents of agilio-ovs-2.2*-fuel.tar.gz:

- Files to be deployed on the compute nodes with Agilio OvS:

  enable_agilio_passthrough.sh
  enable_agilio_relay.sh

- Installation scripts for the Fuel master node:

  00_backup_settings.sh: backs up configuration files to be modified
  01_install_agilio_master_node.sh: installs the deb repo on the master node and updates the mcollective docker image
  02_generate_bootstrap.sh: script to generate a bootstrap image with Agilio OvS drivers loaded

- Utility scripts:

  `forward.sh`: Sample firewall configuration to allow the Fuel master node to act as a gateway
  `update_yaml/`: Utility scripts that updates the bootstrap yaml config

- Resource packages:

  `agilio_master_node.tar`: contains the resource files required for the Fuel master node setup
  `fuel_puppet_config.tar.bz2`: example puppet configuration files for the Fuel master node
  `cirros-x86_64-nfp_netvf.img`: minimal cirros image with nfp_netvf netdev

Installation steps on the Fuel master node

1. Run `00_backup_settings.sh` to make a backup copy of the configuration files that will be modified during this install. This will make a backup of a number of files in `/etc`, leaving a copy in a file with the `.orig` suffix. If the backup file exists, it will not be overwritten.

2. Run `01_install_agilio_master_node.sh` in order to extract the resources, update the `mcollective` docker image and enable the agilio repo on the master node.

3. The Fuel puppet scripts require modification to disable security groups. `fuel_puppet_config.tar.bz2` contains the updated files: they can be reviewed and used to replace the puppet scripts on the Fuel orchestration server.

4. After these scripts have been updated, they can be applied by running:
   ```
   fuel rel --sync-deployment-tasks --dir /etc/puppet/
   ```

5. Run `02_generate_bootstrap.sh` to generate and apply a Fuel bootstrap image containing the Agilio OvS drivers. Note that this operation will take a long time and requires access to upstream repositories. This process requires a large amount of temporary workspace, allow for 2GB of free space.

Update the firmware (if necessary) on the Agilio compute nodes

After the bootstrap image has been updated, and the Fuel slave nodes have been booted using this image, they should be discovered by the Fuel master node.

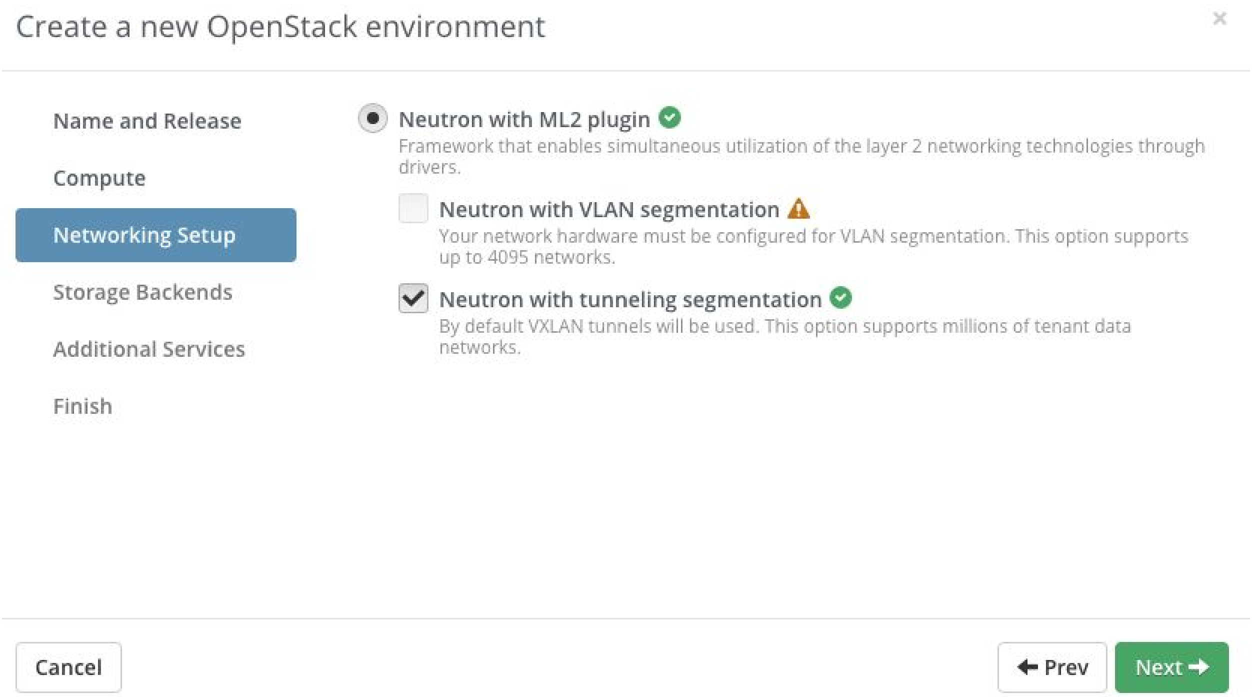
The command “fuel nodes” can be used to determine the management IP addresses of the compute nodes. Use this IP address to access the root shell of the compute nodes.

If required, execute nfp-update-flash.sh on the nodes.
NOTE: Do not interrupt this process.

Create a new cloud

When creating a new OpenStack Environment, make sure to choose the following configuration options:

In ‘Create a new OpenStack environment -> Networking Setup’:
Choose ‘Neutron with tunneling segmentation’



Create a new OpenStack environment

● Neutron with ML2 plugin ✓
Framework that enables simultaneous utilization of the layer 2 networking technologies through drivers.

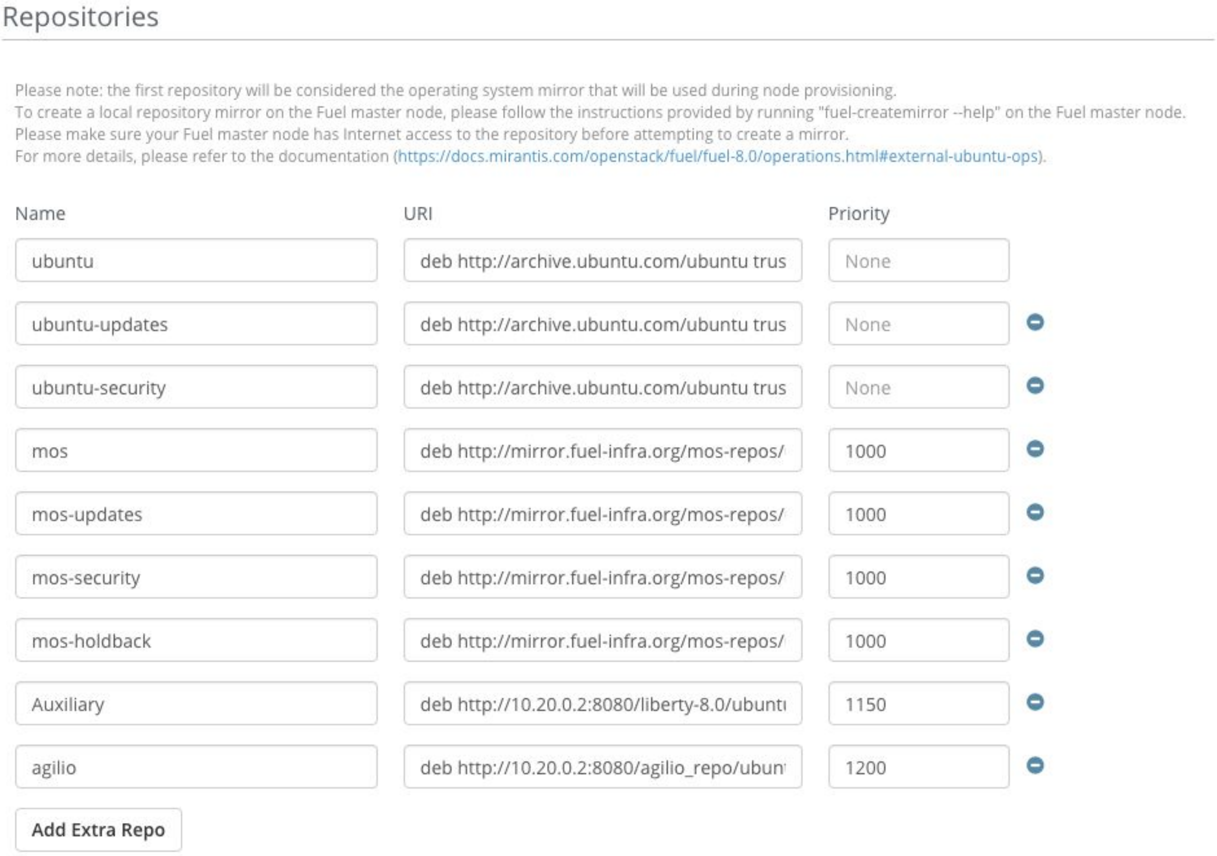
□ Neutron with VLAN segmentation ⚠
Your network hardware must be configured for VLAN segmentation. This option supports up to 4095 networks.

☑ Neutron with tunneling segmentation ✓
By default VXLAN tunnels will be used. This option supports millions of tenant data networks.

In ‘OpenStack Environment Dashboard -> Settings -> General’:
Add an additional repo named ‘agilio’ with priority 1200:

```
deb http://10.20.0.2:8080/agilio_repo/ubuntu trusty main
```

(Replace the IP address with the corresponding address used during the Fuel master node install.)



Repositories

Please note: the first repository will be considered the operating system mirror that will be used during node provisioning.
To create a local repository mirror on the Fuel master node, please follow the instructions provided by running "fuel-createmirror --help" on the Fuel master node.
Please make sure your Fuel master node has Internet access to the repository before attempting to create a mirror.
For more details, please refer to the documentation (https://docs.mirantis.com/openstack/fuel/fuel-8.0/operations.html#external-ubuntu-ops).

Name	URI	Priority
ubuntu	deb http://archive.ubuntu.com/ubuntu trus	None
ubuntu-updates	deb http://archive.ubuntu.com/ubuntu trus	None
ubuntu-security	deb http://archive.ubuntu.com/ubuntu trus	None
mos	deb http://mirror.fuel-infra.org/mos-repos/	1000
mos-updates	deb http://mirror.fuel-infra.org/mos-repos/	1000
mos-security	deb http://mirror.fuel-infra.org/mos-repos/	1000
mos-holdback	deb http://mirror.fuel-infra.org/mos-repos/	1000
Auxiliary	deb http://10.20.0.2:8080/liberty-8.0/ubunt	1150
agilio	deb http://10.20.0.2:8080/agilio_repo/ubun	1200

Add the following to the default kernel parameters:

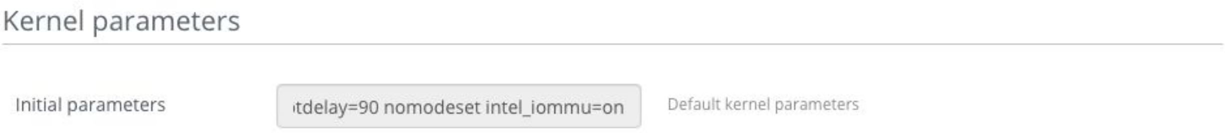
- For Passthrough Acceleration

  ```
  intel_iommu=on
  ```

- For VirtIO Relay Acceleration

  ```
  hugepagesz=2M hugepages=8192 default_hugepagesz=2M
  ```

  (Modify the hugepage parameters to suit your install)

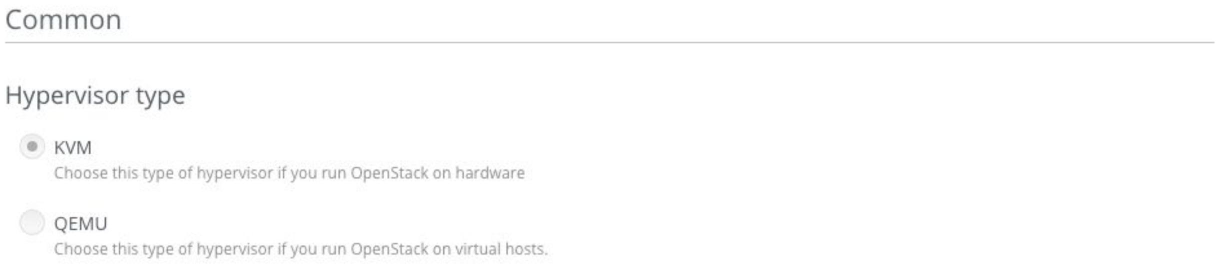


Kernel parameters

Initial parameters: tdelay=90 nomodeset intel_iommu=on

Default kernel parameters

In ‘OpenStack Environment Dashboard -> Settings -> Compute’:
Enable KVM as the Hypervisor type



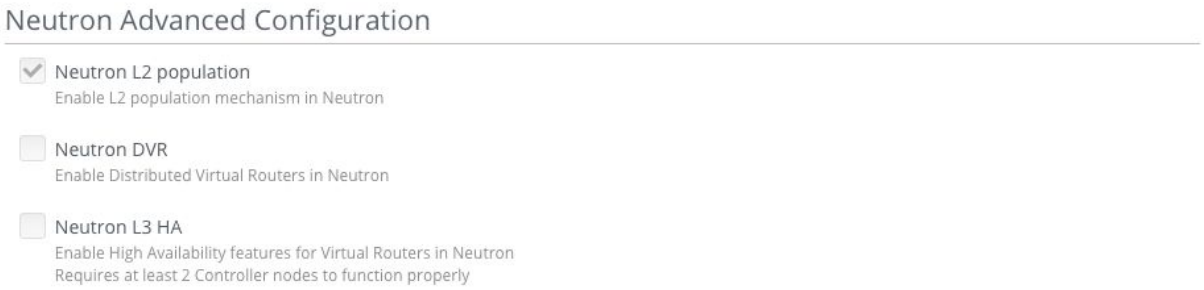
Common

Hypervisor type

● KVM
Choose this type of hypervisor if you run OpenStack on hardware

○ QEMU
Choose this type of hypervisor if you run OpenStack on virtual hosts.

In ‘OpenStack Environment Dashboard -> Network Settings -> Other’:
Enable Neutron L2 population under ‘Neutron Advanced Configuration’



Neutron Advanced Configuration

☑ Neutron L2 population
Enable L2 population mechanism in Neutron

□ Neutron DVR
Enable Distributed Virtual Routers in Neutron

□ Neutron L3 HA
Enable High Availability features for Virtual Routers in Neutron
Requires at least 2 Controller nodes to function properly

Note: before provisioning the servers, make sure to run the network connectivity check. If that fails, it's unlikely a provisioning run will be successful.

Provision an OpenStack Cloud

Make sure that the correct BIOS settings have been enabled on the compute nodes with Agilio hardware. In particular, SR-IOV and Virtualization options need to be activated.

Currently, PXE boot deployment using MOS8.0 has not been tested on Agilio interfaces; a secondary management network is recommended for this.

Activate Agilio acceleration on the compute nodes

The command “`fuel nodes`” can be used to determine the management IP addresses of the compute nodes. Use this IP address to access the root shell of the compute nodes.

A compute node may be set up in one of two configurations: VFIO passthrough or VirtIO Relay.

Passthrough acceleration

Copy `enable_agilio_passthrough.sh` to the node and execute it.

VirtIO Relay acceleration

Copy `enable_agilio_relay.sh` to the node and execute it.

Follow the instructions to set up huge page support in:
https://www.mirantis.com/blog/mirantis-openstack-7-0-nfvi-deployment-guide-huge-pages/

- On the compute node:

Confirm the required hugepage setup on the kernel command-line
set `KVM_HUGEPAGES=1` in `/etc/default/qemu-kvm`
run `update-grub2` and `update-initramfs -u`

- On the control node:

Follow the instructions to set up a flavor that supports hugepages in the link above.

Endpoint configuration

Depending on the configuration, it might be useful to change the configuration of the tunnel endpoints to move them off the Linux bridge set up by Fuel.

This can be done by modifying the interfaces configured at:

```
    /etc/network/interfaces.d/ifcfg-*
```

Note that adding or removing interfaces might change the status of the Fuel slave node to “discover”. OpenStack operation of the cloud should not be affected.

Spawn instances

A minimal cirros image with the nfp_netvf driver can be uploaded to test network connectivity of new instances. Copy the cirros-x86_64-nfp_netvf.img image to the control node, and enter the following commands:

```
 source openrc
glance image-create --name cirros-nfp --disk-format qcow2 \
    --container-format bare --file cirros-x86_64-nfp_netvf.img \
    --property visibility=public
```

Instances on compute nodes with passthrough acceleration will now spawn with Agilio VFs passed through instead of virtio netdevs. VM images require the nfp_netvf driver available in nfp-bsp-release-2015.11-dkms_*_all.deb in the /root/agilio_debs directory on the Fuel orchestration server.

Instances on compute nodes with virtio relay acceleration will spawn using vhost-user. Internally the VM's will have virtio NICs, requiring no additional drivers.

5.3 Health Check Results

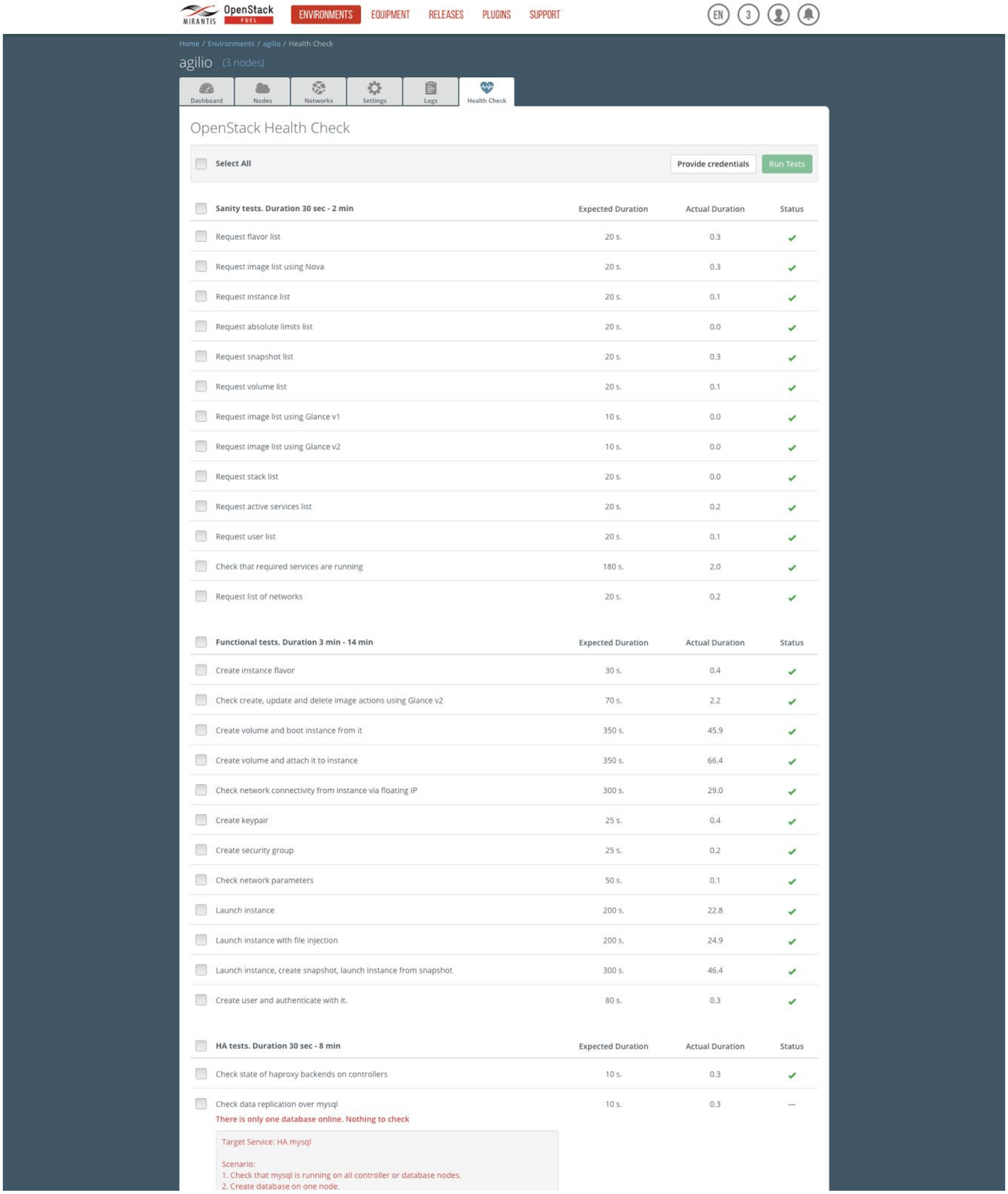
When running a health check on a cloud configured with passthrough acceleration, snapshotting and suspending VM's is expected to fail because the VM's have direct hardware access. With VirtIO Relay acceleration, these operations will complete successfully. Consult Appendix A for an example run.

OpenStack Release	Liberty on Ubuntu 14.04
Compute	KVM
Network	Neutron with tunneling segmentation

5.4 Limitations

Tuning is required, in particular CPU and memory pinning, in order to provide optimal performance. Consult the Agilio OvS user's guide for more information in this regard. Passthrough acceleration prohibits Live Migration.

Appendix A: Health Check Results



agilio (3 nodes)

OpenStack Health Check

□ Select All

□ Sanity tests. Duration 30 sec - 2 min	Expected Duration	Actual Duration	Status
□ Request flavor list	20 s.	0.3	✓
□ Request image list using Nova	20 s.	0.3	✓
□ Request instance list	20 s.	0.1	✓
□ Request absolute limits list	20 s.	0.0	✓
□ Request snapshot list	20 s.	0.3	✓
□ Request volume list	20 s.	0.1	✓
□ Request image list using Glance v1	10 s.	0.0	✓
□ Request image list using Glance v2	10 s.	0.0	✓
□ Request stack list	20 s.	0.0	✓
□ Request active services list	20 s.	0.2	✓
□ Request user list	20 s.	0.1	✓
□ Check that required services are running	180 s.	2.0	✓
□ Request list of networks	20 s.	0.2	✓

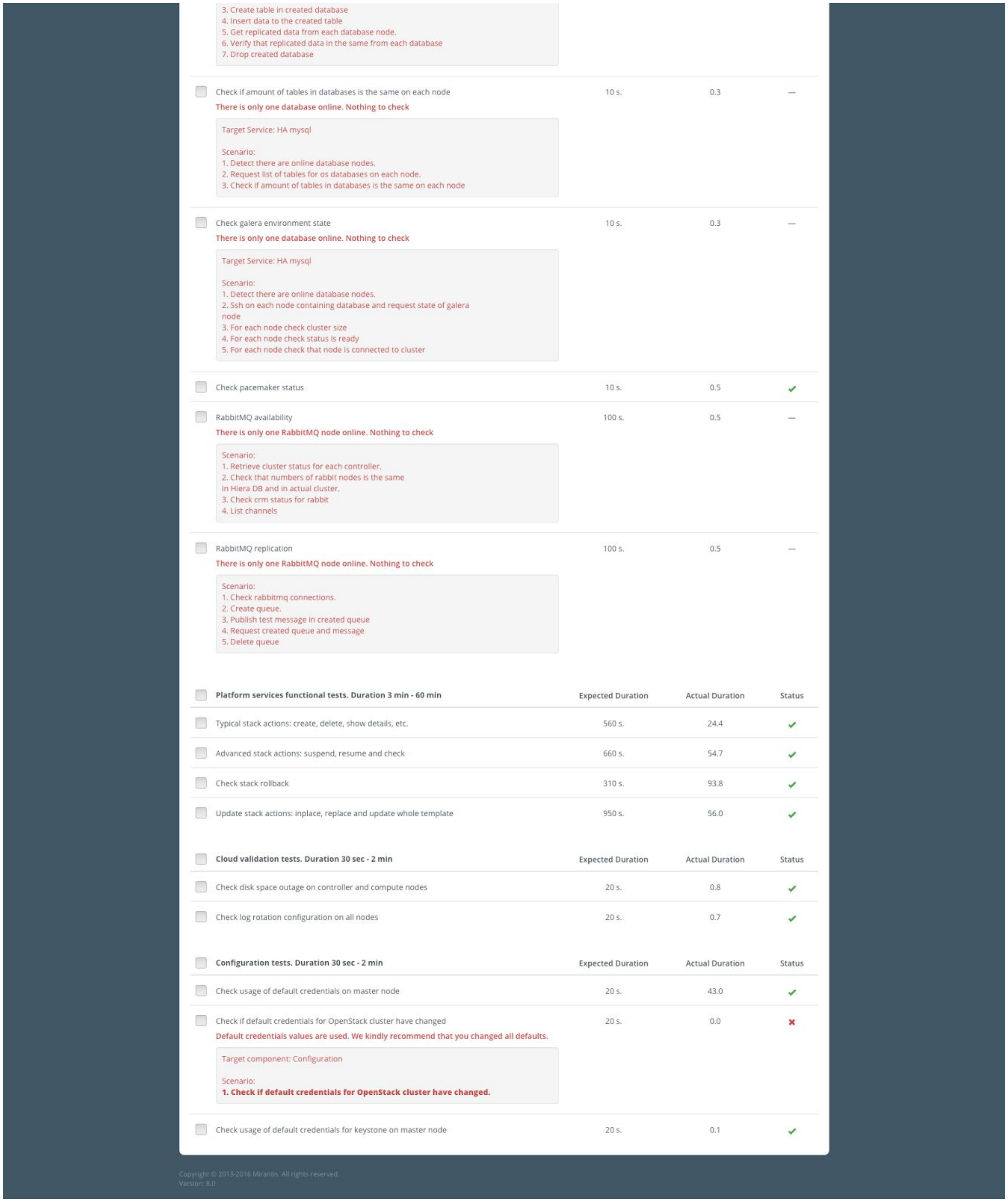
□ Functional tests. Duration 3 min - 14 min	Expected Duration	Actual Duration	Status
□ Create instance flavor	30 s.	0.4	✓
□ Check create, update and delete image actions using Glance v2	70 s.	2.2	✓
□ Create volume and boot instance from it	350 s.	45.9	✓
□ Create volume and attach it to instance	350 s.	66.4	✓
□ Check network connectivity from instance via floating IP	300 s.	29.0	✓
□ Create keypair	25 s.	0.4	✓
□ Create security group	25 s.	0.2	✓
□ Check network parameters	50 s.	0.1	✓
□ Launch instance	200 s.	22.8	✓
□ Launch instance with file injection	200 s.	24.9	✓
□ Launch instance, create snapshot, launch instance from snapshot	300 s.	46.4	✓
□ Create user and authenticate with it.	80 s.	0.3	✓

□ HA tests. Duration 30 sec - 8 min	Expected Duration	Actual Duration	Status
□ Check state of haproxy backends on controllers	10 s.	0.3	✓
□ Check data replication over mysql	10 s.	0.3	—

There is only one database online. Nothing to check

Target Service: HA mysql

Scenario:
1. Check that mysql is running on all controller or database nodes.
2. Create database on one node.



3. Create table in created database
4. Insert data to the created table
5. Get replicated data from each database node.
6. Verify that replicated data in the same from each database
7. Drop created database

□	Check if amount of tables in databases is the same on each node **There is only one database online. Nothing to check** Target Service: HA mysql Scenario: 1. Detect there are online database nodes. 2. Request list of tables for os databases on each node. 3. Check if amount of tables in databases is the same on each node	10 s.	0.3	—
□	Check galera environment state **There is only one database online. Nothing to check** Target Service: HA mysql Scenario: 1. Detect there are online database nodes. 2. Ssh on each node containing database and request state of galera node 3. For each node check cluster size 4. For each node check status is ready 5. For each node check that node is connected to cluster	10 s.	0.3	—
□	Check pacemaker status	10 s.	0.5	✓
□	RabbitMQ availability **There is only one RabbitMQ node online. Nothing to check** Scenario: 1. Retrieve cluster status for each controller. 2. Check that numbers of rabbit nodes is the same in Hiera DB and in actual cluster. 3. Check crm status for rabbit 4. List channels	100 s.	0.5	—
□	RabbitMQ replication **There is only one RabbitMQ node online. Nothing to check** Scenario: 1. Check rabbitmq connections. 2. Create queue. 3. Publish test message in created queue 4. Request created queue and message 5. Delete queue	100 s.	0.5	—

□	**Platform services functional tests. Duration 3 min - 60 min**	Expected Duration	Actual Duration	Status
□	Typical stack actions: create, delete, show details, etc.	560 s.	24.4	✓
□	Advanced stack actions: suspend, resume and check	660 s.	54.7	✓
□	Check stack rollback	310 s.	93.8	✓
□	Update stack actions: inplace, replace and update whole template	950 s.	56.0	✓

□	**Cloud validation tests. Duration 30 sec - 2 min**	Expected Duration	Actual Duration	Status
□	Check disk space outage on controller and compute nodes	20 s.	0.8	✓
□	Check log rotation configuration on all nodes	20 s.	0.7	✓

□	**Configuration tests. Duration 30 sec - 2 min**	Expected Duration	Actual Duration	Status
□	Check usage of default credentials on master node	20 s.	43.0	✓
□	Check if default credentials for OpenStack cluster have changed **Default credentials values are used. We kindly recommend that you changed all defaults.** Target component: Configuration Scenario: **1. Check if default credentials for OpenStack cluster have changed.**	20 s.	0.0	✗
□	Check usage of default credentials for keystone on master node	20 s.	0.1	✓

Version: 8.0