

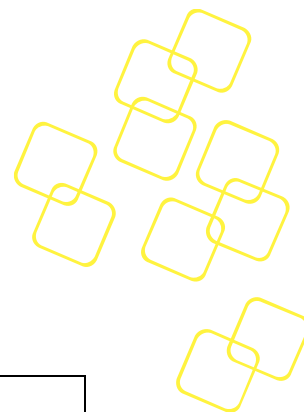
APPLICATION NOTE

REVISION 1.0

DATE 2016/07/14

DEPLOYING OPENSTACK ON THE SKY-8200 CARRIER GRADE SERVER USING MIRANTIS 8.0





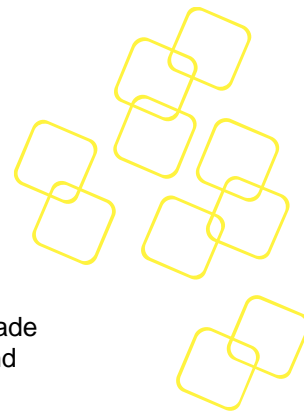
Revision History

Date [mm/dd/yyyy]	Revision	Modifications
07/14/2016	1.0	First release

© Copyright 2016 – Advantech Co., Ltd.

All Rights Reserved

Advantech Co., Ltd. reserves the right to make improvements in the products described in this manual at any time without notice. No part of this manual may be reproduced, copied, translated or transmitted in any form or by any means without the prior written permission of Advantech Co., Ltd. Information provided in this manual is intended to be accurate and reliable. However, Advantech Co., Ltd. assumes no responsibility for its use, nor for any infringements of the rights of third parties, which may result from its use.



Description

This application note describes how to deploy OpenStack on the SKY-8200 Carrier Grade Server using Mirantis 8.0. It is based on the standard process that Mirantis provides and highlights any additional steps required.

We appreciate your input

Please let us know if you consider any aspect of this application note needs improving or correcting. We appreciate your valuable input in helping make our products and documentation better.

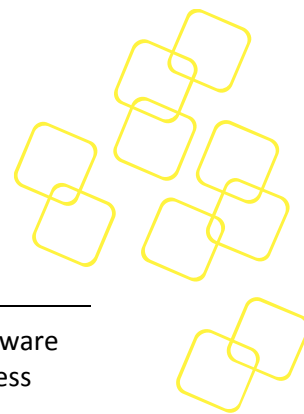
Please send all such comments in writing to: ncg@advantech.com

Acknowledgements

Xeon, QuickAssist and Intel are trademarked by Intel Corp.

Mirantis and FUEL are registered trademarks of Mirantis, Inc.

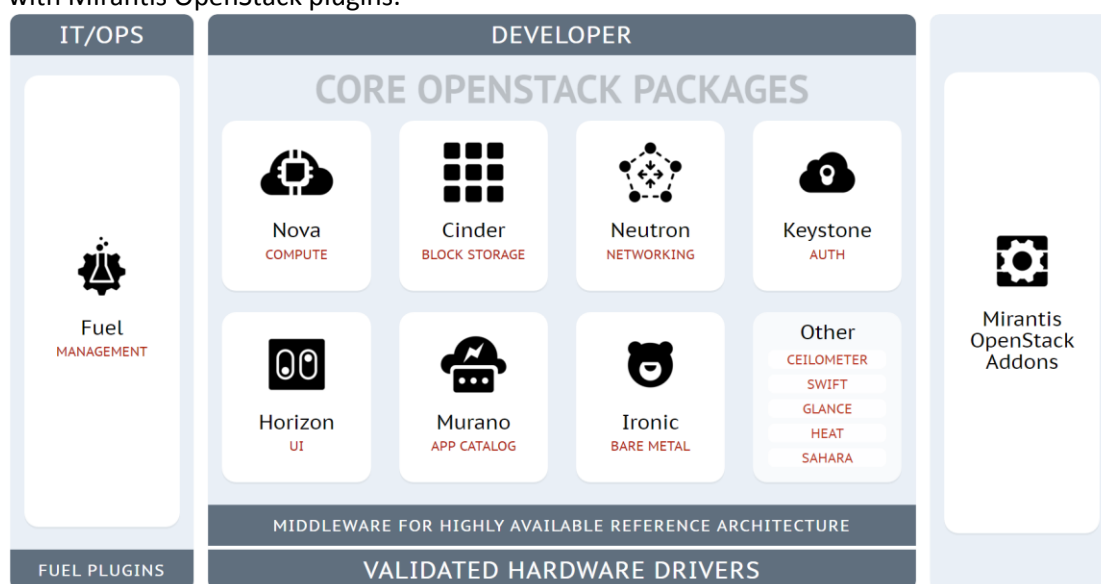
All other product names or trademarks are properties of their respective owners.



1. MIRANTIS UNLOCKS OPENSTACK

An open cloud is vital in helping enterprises and service providers compete in the software economy. With Mirantis OpenStack, you can build web scale clouds that unlock business agility.

Mirantis OpenStack consists of Core OpenStack services weaved into a proven HA reference architecture that can be managed with OpenStack native operations tooling. It can be extended with Fuel deployment plugins and validated drivers. And it can also be extended with Mirantis OpenStack plugins.



While the OpenStack community provides continuous integration (CI) testing, Mirantis provides a significantly higher level of testing. The packages are built from individual project repositories and are tested together using Mirantis reference architectures to ensure that these various software pieces work bug-free as a whole.

The scope of Mirantis testing also explains why customers are increasingly moving away from a do-it-yourself (DIY) approach and moving to Mirantis OpenStack. It is very hard for a single customer to test subsequently fix bugs at the same level of intensity.

Integration with Fuel for deployment and relevant onsite cluster health checks is also a critical piece of the overall hardening story.

Fuel is the leading purpose-built open source deployment and management tool for OpenStack. Developed as an OpenStack community effort, it provides an intuitive, GUI-driven experience for automated deployment and management of OpenStack, related community projects and plugins.

<https://www.mirantis.com/products/mirantis-openstack-software/openstack-deployment-fuel/>



3. FUEL MASTER NODE INSTALLATION

The installation of the Mirantis 8.0 Fuel Master Node is based on the Mirantis Fuel install guide which can be found at the link below.

<https://docs.mirantis.com/openstack/fuel/fuel-8.0/fuel-install-guide.html#install-install-fuel-master-node>

3.1 Deploying Fuel Master Node over PXE

The Fuel Master Node is deployed over PXE. Details on how to do this can be found below. In this setup, the RAS server was used as a one-off PXE server.

https://docs.fuel-infra.org/fuel-dev/develop/pxe_deployment.html

3.2 Enable serial console for Fuel Master Node installation

As is typical for Carrier Grade networks, graphical display output may not be present on the server or may not be accessible over the network. Therefore, a serial console connection has been used in this example (local RS-232 output or via serial-over-LAN (SoL)). Please remember to add

```
--- console=ttyS0,115200

DEFAULT menu.c32
prompt 0
MENU TITLE My Distro Installer

TIMEOUT 600

LABEL localboot
MENU LABEL ^Local Boot
MENU DEFAULT
LOCALBOOT 0

LABEL fuel
MENU LABEL Install ^FUEL
KERNEL /fuel/isolinux/vmlinuz
INITRD /fuel/isolinux/initrd.img
APPEND biosdevname=0 ks=nfs:10.20.0.1:/var/lib/tftpboot/fuel/ks.cfg <<CUT FOR ILLUSTRATION>> showmenu=nc--- console=ttyS0,115200

LABEL reboot
MENU LABEL ^Reboot
KERNEL reboot.c32

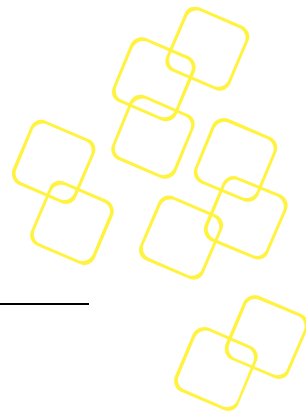
LABEL poweroff
MENU LABEL ^Poweroff
KERNEL poweroff.com
```

at the end of the APPEND line in the `/var/lib/tftpboot/pxelinux.cfg/default` file, where `ttyS0` is the serial console to be used and `115200` is the baud rate.

This option will assure serial console output during the deployment of the master node, which may be very helpful in case there are any issues.

Alternatively, as suggested in the referenced document from Mirantis, you can add other directives to ensure a fully silent installation if no display is available.

After the Fuel Master Node has been deployed, please do not forget to switch off the PXE server to avoid interference with the OpenStack deployment process.



4. PREPARING THE FUEL MASTER NODE

4.1 Updating the Bootstrap image

Login to the Fuel Master Node over SSH as described in:

<https://docs.mirantis.com/openstack/fuel/fuel-8.0/fuel-install-guide.html#install-login-fuel-master-node>

Mirantis 8.0 OpenStack is deployed using the default kernel that comes with Ubuntu 14.04 which may be too old and therefore cause some kernel driver issues on the SKY-8200 Carrier Grade server.

We strongly recommend you create a new Bootstrap image using the latest Ubuntu Its-trusty kernel following the instructions provided by Mirantis in the links below.

<https://docs.mirantis.com/openstack/fuel/fuel-8.0/fuel-install-guide.html#bootstrap-install-kernel>

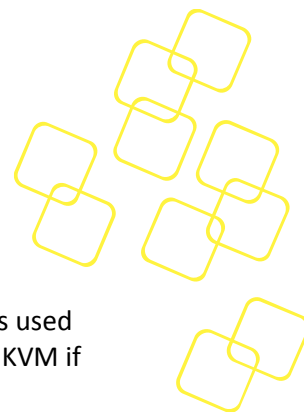
Note: If you desire to enable serial console for debug purpose in the bootstrap images please add the following to your build command.

```
--extend-kopts console=ttyS0,115200
```

If you deploy with a distribution other than Ubuntu and are using your own bootstrap images, please ensure that you have a kernel that supports all the components inside the SKY-8200 Carrier Grade server. The latest Its-trusty kernel used by Ubuntu is 4.2.0-27, this is tested and fully supported by the latest available hardware.

More information about the different hardware components can be found on the Advantech website at the link below.

http://www.advantech.com/products/e50f7a84-65ea-497b-9358-0d2f618b9b6d/sky-8200/mod_a37c9242-a618-4e3c-bba7-eb81e99a425f



4.2 Identify the Systems

Serial-over-LAN (SoL) as provided by the BMC on the SKY-8200 Carrier Grade servers is used in this example for console output. Alternatively, you could use local serial console or KVM if accessible.

To identify the systems:

1. Set the server to boot into BIOS by using the IPMI *chassis bootdev bios* command
2. Power on the server by using the IPMI *power on* command
3. Enable console by using IPMI *sol activate* command
4. Modify boot order in BIOS to always PXE boot
5. Save the BIOS configuration and reboot the machine
6. If done correctly, you will see the PXE boot initiating from the Fuel Master Node on the console.

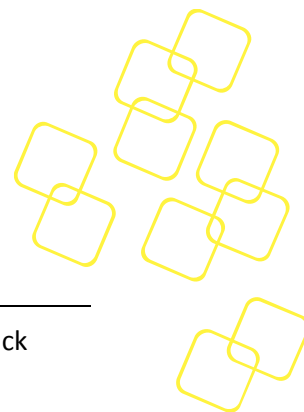
Note:

If you didn't enable console redirection in your bootstrap image, console output will stop after the initrd has been loaded. However, the image will still be booting without console output.

7. Login to the Fuel GUI and, under the "Equipment" tab, the SKY-8200 Carrier Grade should appear after a short time. It will be named "untitled". It is strongly recommend to rename the machine to easily identify which machine is which.
8. After all four machines have booted and have become visible in the "Equipment" tab, you can proceed.

The screenshot shows the Fuel GUI interface. At the top, there are navigation icons (hamburger menu, grid, add, refresh, filter, search). Below is a 'Sort By' dropdown set to 'Status'. The main content area is titled 'Discovered (4)' and includes a 'Select All' checkbox. It contains a table with four rows, each representing a discovered system.

System ID	Status	Hardware Specs	Actions
SKY-8200-1	DISCOVERED	CPU: 2 (56) HDD: 0.9 TB RAM: 32.0 GB	⚙️
SKY-8200-2	DISCOVERED	CPU: 2 (48) HDD: 0.9 TB RAM: 32.0 GB	⚙️
SKY-8200-3	DISCOVERED	CPU: 2 (48) HDD: 0.9 TB RAM: 32.0 GB	⚙️
SKY-8200-4	DISCOVERED	CPU: 2 (48) HDD: 0.9 TB RAM: 32.0 GB	⚙️



5. CREATING A NEW OPENSTACK ENVIRONMENT

After all four systems have become available in the Fuel Master Node, a new OpenStack Environment can be created.

Following the user guide provided by Mirantis the deployment wizard is used as described in <https://docs.mirantis.com/openstack/fuel/fuel-8.0/fuel-user-guide.html#create-a-new-openstack-environment>

The default settings for creating a new OpenStack environment have been used in this example. However, you could add additional items as described in the documentation provided by Mirantis.

5.1 Configuring the OpenStack environment

There are a lot of options when configuring the OpenStack environment. The linked document below provides a detailed explanation regarding how to perform an advanced configuration.

<https://docs.mirantis.com/openstack/fuel/fuel-8.0/fuel-user-guide.html#configure-your-environment>

5.2 Adding nodes to the environment

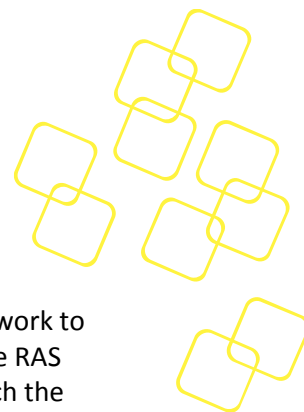
Following the above link you can add nodes to the environment, defining their role(s) as you desire.

This example of provisioning with Mirantis 8.0 describes a simple deployment with three controllers and only one compute node.

You could scale out afterwards with more Compute nodes and/or add additional functions to OpenStack such as Ironic bare metal control for example.

The screenshot displays the OpenStack environment configuration interface. At the top, there are navigation tabs: 'Configure Disks', 'Configure Interfaces', and a green '+ Add Nodes' button. Below the tabs, there is a 'Sort By' dropdown set to 'Roles' and a 'Select All' checkbox. The main content area shows two groups of nodes:

- Controller, Storage - Cinder (3)**: This group contains three nodes, each with a checkbox, a name (SKY-8200-2, SKY-8200-3, SKY-8200-4), a role (CONTROLLER - CINDER), a status (PENDING ADDITION), and hardware specifications (CPU: 2 (48) HDD: 0.9 TB RAM: 32.0 GB). Each node has a gear icon for configuration.
- Compute, Storage - Cinder (1)**: This group contains one node with a checkbox, a name (SKY-8200-1), a role (COMPUTE - CINDER), a status (PENDING ADDITION), and hardware specifications (CPU: 2 (56) HDD: 0.9 TB RAM: 32.0 GB). It also has a gear icon for configuration.



5.3 Setting up networks

Keep the default Mirantis network configuration and add a VLAN ID on the Public network to create a route to the Public network, allowing each node to access the internet via the RAS server. This requires you define the Gateway IP address of the Public network to match the IP address of the RAS server.

Network Settings (Neutron with VLAN segmentation) Add New Node Network Group

Node Network Groups **default**

This node network group uses a shared admin network and cannot be deleted

Settings

Public

The Public network allows inbound connections to VMs (Controllers and Tenant VMs) from external networks (e.g., the Internet) as well as outbound connections from VMs to the external networks.

Neutron L2

Neutron L3

Other

Network Verification

Connectivity Check

CIDR ☐ Use the whole CIDR

IP Range

Start End

Gateway

Use VLAN tagging ☒

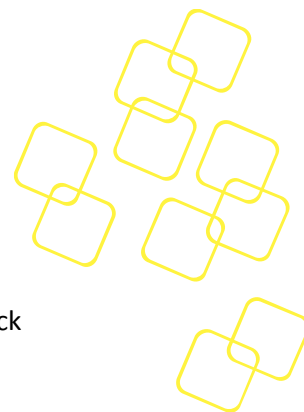
5.4 Configure network interfaces

The final configuration step is to configure the network interfaces so that they reflect the Network Architecture displayed in 2.2. Below you can see how we separated the different networks over the available interfaces.

Configure interfaces on 4 nodes

Bond Network Interfaces Unbond Network Interfaces

<input type="checkbox"/>	Name: enp5s0 Speed: 1.0 Gbps	Admin (PXE)	Management VLAN ID: 101	Offloading Modes: Default	MTU <input type="text" value="Default"/>
<input type="checkbox"/>	Name: enp6s0 Speed: 1.0 Gbps	Public VLAN ID: 103		Offloading Modes: Default	MTU <input type="text" value="Default"/>
<input type="checkbox"/>	Name: ens2f0 Speed: 10.0 Gbps	Storage VLAN ID: 102		Offloading Modes: Default	MTU <input type="text" value="Default"/>
<input type="checkbox"/>	Name: ens2f1 Speed: 10.0 Gbps	Private VLAN ID: 1000-1030		Offloading Modes: Default	MTU <input type="text" value="Default"/>



5.5 Network configuration check

Prior to the deployment of OpenStack it is recommended to run the Connectivity Check which is under the “Networks” tab.

Network Settings (Neutron with VLAN segmentation) Add New Node Network Group

Node Network Groups **Connectivity Check**

default

Settings

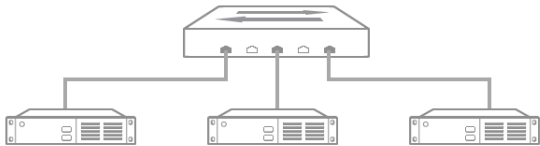
Neutron L2

Neutron L3

Other

Network Verification

Connectivity Check



Network verification checks the following:

1. L2 connectivity checks between nodes in the environment.
2. DHCP discover check on all nodes.
3. Repository connectivity check from the Fuel Master node.
4. Repository connectivity check from the Fuel Slave nodes through the public & admin (PXE) networks.

Verify Networks

Verification succeeded. Your network is configured correctly.

5.6 Live Deployments of OpenStack

It is highly recommended for live deployments to enable and configure TLS. As this example only covers a lab use case, TLS was not setup / configured.

List of changes: TLS is not enabled. It is highly recommended to enable and configure TLS.

Added 4 nodes +

Deploy Changes

Summary

Name	Example ✎
Status	New
OpenStack Release	Liberty on Ubuntu 14.04
Compute	QEMU
Network	Neutron with VLAN segmentation
Storage Backends	Cinder LVM over iSCSI for volumes

Delete Environment Reset Environment + Add Nodes

Capacity

CPU (Cores)	200	HDD	3.6 TB	RAM	128.0 GB
-------------	-----	-----	--------	-----	----------

Node Statistics

Total Nodes	4	Pending Addition	4
Controller	3		
Compute	1		
Storage - Cinder	4		

Mirantis 8.0 provides a Health Check option which will test all aspects of the OpenStack installation.