THE OPENSTACK KICKSTART GUIDE FOR THE DSS 9000
Transforming management for scale out infrastructure

ABSTRACT / SCOPE
This document intends to demonstrate how Dell EMC® has greatly simplified the management of OpenStack cloud platforms by integrating Intel® Rack Scale Design (RSD) and the OpenStack Fuel management utility. To that end, the Dell EMC Extreme Scale Infrastructure (ESI) group provides integration software that combines the Intel Pod Manager (PODM) component of RSD, and OpenStack Fuel to provide an easier and seamless “kickstart” for workload deployment. An example reference implementation and step-by-step description are included to illustrate how an administrator uses PODM and Fuel to quickly and easily initialize, provision and configure workloads in an OpenStack cluster environment, on the DSS 9000 infrastructure solution.

January, 2017
# TABLE OF CONTENTS

## EXECUTIVE SUMMARY ................................................................................................................. 4
  Audience ........................................................................................................................................ 4

## 1 OVERVIEW OF DSS 9000 MANAGEMENT COMPONENTS .................................................. 5
  1.1 DSS 9000 hardware .................................................................................................................. 5
  1.2 Top of Rack (ToR) network switch ......................................................................................... 5
  1.3 Utility node ............................................................................................................................. 5
  1.4 Intel Rack Scale Design .......................................................................................................... 5
  1.5 Redfish API ............................................................................................................................ 6
  1.6 Intel POD Manager .................................................................................................................. 6
  1.7 OpenStack Fuel ....................................................................................................................... 7
  1.8 Dell EMC kickstart integration components ......................................................................... 7

## 2 CONFIGURATION AND PREPARATION OF THE REFERENCE IMPLEMENTATION ...... 7
  2.1 The DSS 9000 reference implementation ............................................................................ 8
  2.2 Management Controller (MC) configuration ....................................................................... 8
  2.3 Network configuration ........................................................................................................... 8
  2.4 Utility node configuration ..................................................................................................... 9
  2.5 Fuel Master installation and configuration ......................................................................... 9
  2.6 KickStart integration software installation and configuration ........................................... 9

## 3 OPENSTACK CLOUD DEPLOYMENT: ALLOCATE, PROVISION, DEPLOY ............ 11
  3.1 Allocate resources for a workload ....................................................................................... 11
  3.2 Assemble the nodes .............................................................................................................. 11
  3.3 Update BIOS/RAID and firmware ..................................................................................... 12
  3.2 Deploy the nodes into the OpenStack Cluster ................................................................... 14

## 4 STEP BY STEP APPROACH TO DYNAMIC WORKLOAD ASSIGNMENT ............ 14
  Prerequisite checklist .................................................................................................................. 14
  Step-by-step tasks ....................................................................................................................... 14

## FOR MORE INFORMATION ........................................................................................................ 15
EXECUTIVE SUMMARY

The increasingly complex nature of today’s innovative software solutions require more and more IT resources, which in turn accentuate the need for more efficient IT solutions. The resulting virtualization of all aspects of IT has resulted in truly software defined data centers (SDDC), where all the resources needed for a particular workload can be (virtually) mustered from anywhere in the physical datacenter - across machines, operating systems, vendor brands – transparently to the application. The applications, users, and business owners now operate in a world that is simpler, faster and more efficient for them. Thanks to these innovations they can readily scale up to thousands of nodes.

But now, transformation is needed in the management of large scale IT environments - not just hyperscale data centers, but growing scale customers as well. To justify IT outlays these companies need to “get to live” or “get to money” more rapidly than has been available in the past. They need a simpler, faster way to deploy and manage massively-scaled IT environments.

Dell EMC and Intel have collaborated using Intel® Rack Scale Design and OpenStack to allow for more simplified and automated management of scale-out resources on the latest rack scale solution from Dell EMC – the DSS 9000. This paper details the various software components involved, the new orchestration capabilities they bring to rack scale data centers, and how to implement the kickstart management process.

Audience

This paper is intended for storage architects, engineers, and IT administrators who want to understand how to use these new tools to manage scale out infrastructure more easily and to understand the interaction of the various components.
1 Overview of DSS 9000 management components

To understand how the kickstart process works, it is helpful to have an understanding of each of the hardware and software components involved in cloud management on the DSS 9000. Those are:

- DSS 9000 hardware - rack, compute, storage with single point of management
- Top of Rack switch - networking component
- Utility node
- Intel Rack Scale Design (RSD) - architecture and APIs that enable disaggregation
- Redfish API - Open standard management APIs – the basis of much integration
- Intel POD Manager - management software that is part of RSD
- OpenStack Fuel – an open source management utility
- Dell EMC integration components – software that transparently integrates management components

The following sections describe these components in more detail.

1.1 DSS 9000 hardware

The DSS 9000 is a rack level solution comprised of pools of compute, storage and networking resources which are managed through a single point of rack management. Each DSS 9000 allows combinations of compute and storage sleds, along with shared power, cooling and networking to provide maximum configuration flexibility.

Each DSS 9000 also has a hardware controller (rack manager) that serves as the single point of management for the resources in the rack. This rack manager conforms to the Distributed Management Task Force’s (DMTF) Redfish API and the Redfish extensions for Intel POD Manager, enabling easier integration of the management components. (http://www.dmtf.org)

1.2 Top of Rack (ToR) network switch

A DSS 9000 rack solution is agnostic to the Top of Rack switch a customer may choose. The reference implementation this document uses to illustrate the kickstart process is the Dell EMC S4048-ON. It is a x48 SFP+ port network switch running Cumulus Linux OS 2.5.X, with its ports divided into data and management ports. It is possible to add another management switch to segregate the management traffic from the data traffic, if required.

1.3 Utility node

The utility node is an additional server, connected to the DSS 9000 through the ToR, that is used as a management station and runs the DSS 9000 management software. This reference implementation uses a PowerEdge R430 server.

1.4 Intel Rack Scale Design

RSD is a software-defined architecture that allows IT administrators to consider compute, storage, and networking as disaggregated resources that can be assembled dynamically, as needed, to meet various demands in a data center/cloud.

Disaggregation, in addition to allowing hardware refresh at different rates for each of the storage, compute, and networking resources, supports more efficient resource utilization based on right-sized assignments. Imagine a cloud that grows and shrinks to meet optimum utilization by virtue of being comprised of racks of resources which allows dynamic assignment and release, where one might assemble a system of resources where some nodes have copious storage and others provide pure compute horsepower.

To bring such a vision to light, the industry collaborated to define and standardize a RESTful set of APIs – DMTF’s Scalable Platform Management Forum (SPMF) specification is called Redfish (described in more
As part of RSD, Intel also extended the Redfish APIs to allow support of managing node power control, discovering hardware capabilities, and collecting advanced telemetry information.

Dell EMC takes advantage of these open standards to provide transparent integration of the various management components using the kickstart integration software.

### 1.5 Redfish API

The Redfish API is an infrastructure management schema and API standard defined by the Distributed Management Task Force (DMTF). Redfish has the following characteristics:

- RESTful Interface over HTTP in JSON format based on Odata v4
- Useable by client applications and browser-based GUIs
- A Secure interfaces over HTTPS to replace IPMI-over-LAN
- Multi-Node capable replacement for IPMI
- Example Python code to retrieve serial number:
  ```python
  o rawData = urllib.urlopen('http://192.168.1.117/redfish/v1/Systems/1')
  o jsonData = json.loads(rawData)
  o Print(jsonData["SerialNumber"]
  o Output: 1A87C4442K
  ```

The Redfish API is an essential element of the overall integration of management software in the DSS 9000 solution, allowing the various components to interact seamlessly.

### 1.6 Intel POD Manager

PODM is a software component of RSD that supports the assembling and releasing of nodes. POD Manager acts as a resource manager, for disaggregated pools of hardware resources, by communicating using standard Redfish APIs to hardware-aware software within the DSS 9000.

![Figure 1: HW resources from 3 racks are virtually pooled and allocated to 3 workloads](image)

Dell EMC Confidential
1.7 OpenStack Fuel

Fuel is an open source deployment and management tool for OpenStack developed via an OpenStack community effort. It provides an intuitive, Web UI-driven experience for administrators of an OpenStack environment. (See Figure 2 below.)

![OpenStack Fuel Web UI](image)

*Figure 2: OpenStack Fuel Web UI (cropped)*

The simplicity of the Fuel UI streamlines and accelerates the otherwise time-consuming, often complex, and error-prone process of deploying, testing and maintaining various configuration flavors of OpenStack at scale.

The key features of Fuel are:

- Hardware discovery
- Hardware configuration in UI (networks & disk partitioning)
- Ability to spin up and manage multiple OpenStack clusters
- Support for non-HA and HA OpenStack deployment configurations
- Pre-deployment checks and network validation
- Post-deployment checks and running a set of tests for validating deployed OpenStack
- View logs in real-time through UI
- Support for CentOS and Ubuntu, and it can be extended to support other distributions
- Support for multiple OpenStack distributions

Unlike other platform-specific deployment or management utilities, Fuel is an upstream OpenStack project that focuses on automating the deployment and testing of OpenStack and a range of third-party options, so it’s not compromised by hard bundling or vendor lock-in components.

1.8 Dell EMC kickstart integration components

Dell EMC has created integration software that also implements the Redfish/RSD APIs and works with Fuel and RSD, allowing you to easily and quickly construct nodes to deploy an OpenStack Cloud on the DSS 9000. With these components working together, transparently, you can grow and shrink the cloud to meet utilization needs as they arise. How to install, configure and use these components is described in later sections.

2 Configuration and preparation of the reference implementation

The following sections describe the different preparations you need to take to ensure a smooth OpenStack implementation on the DSS 9000.
2.1 The DSS 9000 reference implementation

The DSS 9000 rack scale solution can contain a variable number of compute and storage “blocks” and implements shared power, cooling, and networking infrastructure. Below are the hardware specifications for configurations used on this DSS 9000 reference implementation:

<table>
<thead>
<tr>
<th></th>
<th>DSS 9500 (full-width)</th>
<th>DSS 9520 (half-width)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Processors</td>
<td>2x Intel Xeon E5-2600v4</td>
<td>2x Intel Xeon E5-2600v4</td>
</tr>
<tr>
<td>TDP Max</td>
<td>135W</td>
<td>135W</td>
</tr>
<tr>
<td>Chipset</td>
<td>Intel C610</td>
<td>Intel C610</td>
</tr>
<tr>
<td># DIMMs</td>
<td>Up to 16</td>
<td>Up to 16</td>
</tr>
<tr>
<td>DIMM Type</td>
<td>DDR4</td>
<td>DDR4</td>
</tr>
<tr>
<td>Hot-swap HDDs/SSDs</td>
<td>Up to 12x 3.5” SASSATA</td>
<td>Up to 2x 2.5” SATA (on-board controller)</td>
</tr>
<tr>
<td>Non-HS HDDs/SSDs</td>
<td>Up to 4x 2.5” SAS/SATA</td>
<td>Up to 4x 2.5” or 8x 2.5” SAS/SATA, Up to 4x 2.5” NVMe SSD</td>
</tr>
<tr>
<td>PCIe Gen3 slots</td>
<td>x16 half-height, half-length x8 mezzanine</td>
<td>x16 half-height, half-length x8 mezzanine</td>
</tr>
<tr>
<td>LOM</td>
<td>Dual 10GbE SFP+</td>
<td>Dual 10GbE SFP+</td>
</tr>
<tr>
<td>Management</td>
<td>Dedicated RJ45, BMC, iDRAC, IPMI 2.0/DCMI 1.5, Redfish</td>
<td>Dedicated RJ45, BMC, iDRAC, IPMI 2.0/DCMI 1.5, Redfish</td>
</tr>
</tbody>
</table>

2.2 Management Controller (MC) configuration

The Management Controller (MC) on the DSS 9000 must have firmware 3.19 or newer and the management interface must be set to DHCP.
The hostname on the MC must include the string “psme”, for example: psme-mc-hostname. PODM uses DHCP and the hostname to discover Pooled System Management Engines (PSME).
The MC can be configured via serial or Ethernet by logging into the MC and running the MC CLI. Please refer to the MC CLI user guide for configuration related issues.
It is also important to enable RSD features in the MC. Usually this is done by setting a properties in the MC configuration file found at: /opt/dell/mc/conf/

Below are the properties that are required to be set in order to enable RSD features:

R2Apis=true
R2ApiDataSource=Dynamic
R2DynApiCacheUpdate=AutoAndPoll
R2DynApiCachPollInterval=3600

2.3 Network configuration

In the reference implementation for this paper, each node in the DSS 9000 rack scale solution has two onboard 10Gb LOMs which are connected to the S4040-ON switch.
Since only one switch is used for this reference implementation, the switch is divided into data and management ports. Ports 1-44 are used for the data network to connect to each server and ports 45-48 are used for management network.
The management ports are used for the following:

- Rack manager management port
- Switch management port
- Utility node management station

2.4 Utility node configuration

The utility node is used as the management station and for this reference implementation, a PowerEdge R430 is used. The utility node is required to run Ubuntu 14.04 or 16.04 and needs to have Virtual Box 5.x.x installed in order to host two VMs; one VM to host the Fuel Master, the other to host the POD Manager packages (explained below).

For networking, the utility node is required to have two network interfaces at a minimum; one NIC connected to the PODM management network, the second NIC connected to the data network of the switch.

2.5 Fuel Master installation and configuration

Fuel Master 8.0 or 9.0 needs to be installed as a VM from the Fuel ISO. Although Fuel Master can also be setup on a stand-alone server, for the purpose of this implementation we have chosen to run Fuel as a VM. The Fuel ISO can be downloaded from http://www.mirantis.com. It is outside the scope of this document to detail the Fuel installation. Please refer to http://docs.openstack.org/developer/fuel-docs/userdocs/fuel-install-guide.html for instructions on how to install Fuel.

The Fuel Master VM needs two bridged NIC interfaces (eth0/eth1), where eth0 is bridged to the Fuel PXE network and eth1 is bridged PODM communication.

You will also need to install the Ansible IT automation engine on the Fuel Master to enable the configuration of BIOS/Firmware. For information on how to install Ansible, please refer to: http://docs.ansible.com/ansible/intro_installation.html.

2.6 KickStart integration software installation and configuration

The Dell EMC kickstart integration components consist of several tools that are installed on Fuel Master and the utility node that can be used to get your systems up and running quickly, and simplify deployment of resources.

The integration components can be obtained from Dell EMC on request and are delivered as a single file named kickstart.tar.gz which has separate folders containing the following components:

- Intel PODM virtual box VM
  As part of the kickstart, Dell EMC has created an OVA file that enables you to easily import Intel PODM into VirtualBox. The VM you create during this step requires a single bridged interface to the PODM network. By default, the PODM interface is setup to a static IP 10.3.0.1. PODM also runs a DHCP server that will lease IPs to interfaces connected to the management network.

  Once PODM VM is started, you can login to the VM from the utility node by using SSH. The default username is “user” and default password is “password”

  If you are not able to login to the PODM VM, insure that you have a route from the utility node to the PODM VM over the 10.3.0.0/24 network.

- Kickstart management tools to be installed on utility node
  - The podm_tool.py tool allows you to interface with PODM to do things like power control or set PXE on multiple nodes. This tool also allows you to assemble nodes from a pool of resources.
The `switch_tool.py` tool is used to configure and manage VLANs on an RSD enabled switch. For example, the S4048-ON running the network PSME under Cumulus can be used as an RSD-enabled switch.

- **Kickstart PODM-Fuel component to run on Fuel master node**
  This is a single file called `podm_fuel.py` that needs to run on the Fuel Master. It is a service that runs in a loop that constantly polls PODM to discover newly assembled nodes and ensure they can be powered on and boot into Fuel PXE network. Start this service by running: `python podm_fuel.py <interval in seconds>`.

  Note: This service requires that you have a network interface on the Fuel Master that can reach PODM network.

- **A customized bootstrap image for Fuel to support the DSS 9000**
  This is a Linux shell script that creates and activates a new bootstrap image from within the Fuel master. This script pulls Dell EMC system configuration tools into the newly created image. These tools include `racadm`, `syscfg`, `perccli`, and `arcconf`, which are used to update BIOS/Firmware and configure RAID controllers.

  Note: In order for the bootstrap image to be built successfully, a network interface is required so these tools can be pulled from an [outside repository](https://dell.com) at dell.com.

- **Firmware/BIOS update tools.**

  These tools run on the Fuel master node to apply firmware and BIOS configuration to multiple nodes. They make use of the Ansible playbook, which is why Ansible must be installed on the Fuel master. As part of the Ansible playbook there is a script “update_firmware.sh”. This script initiates a firmware and BIOS update on all discovered nodes. There is also a configuration parameter file called “data.txt” that describes the location/version/payload information for the firmware update process. To start the firmware/BIOS update process, you first edit data.txt and then run the script `update_firmware.sh`.

  The figure below illustrates how the utility node should be set up and how the different components interact.
You are now ready to begin allocating resources on the DSS 9000.

3 OpenStack cloud deployment: allocate, provision, deploy

Once you have done the set-up described in the installation and configuration section, allocating resources on your OpenStack cloud comes down to just a few simple (yet powerful) commands.

3.1 Allocate resources for a workload

Allocate a workload from PODM using the command line on the utility node, for example, to allocate a node made up of 3 servers each with specifically 2 CPUs and 6400 MB of memory, type at the python prompt:

```
python podm_tool.py node allocate "Sample-Workload" 3 CPU=2 MEM=64000
```

3.2 Assemble the nodes

Once the 3 nodes are allocated, assemble them to make them ready to be used, using this command:

```
python podm_tool.py node assemble 1-5
```

At this point, the PODM-Fuel agent will detect the newly allocated workload and automatically power on the nodes, if they are off. It will also automatically set the nodes to “boot to PXE” so all the new nodes can boot to Fuel's PXE network.

Node locale information

Once the nodes are discovered by Fuel, their locale information will be displayed in the Fuel Web-UI.
Figure 4: Node locale information returned in the Fuel UI

Figure 4 shows where the 3 allocated nodes are located in Rack 1 (the specific block and shelf) and the exact characteristics of each node.

3.3 Update BIOS/RAID and firmware

Once your nodes are allocated in a cluster, you can navigate to the kickstart integration firmware/BIOS update utility. From there, you can easily update BIOS/Firmware/RAID on all nodes by running a simple script “update_firmware.sh” that will apply new settings to all the newly discovered nodes from within Fuel. The settings can be specified in a text file that will describe location of the firmware/BIOS/RAID files and BIOS/Firmware settings to be applied to each node.

The following is an example excerpt from a text file for the configuration of a node:
Once configured, the nodes are ready to deploy.
3.2 Deploy the nodes into the OpenStack Cluster

Using the Fuel Web-UI, a new OpenStack environment can be created and the newly discovered nodes can added to the cluster using standard Fuel methods.

4 Step by Step approach to dynamic workload assignment

Prerequisite checklist

- The utility node must run Ubuntu 14.04 or 16.04
- The utility node needs to have Virtual Box 5.x.x installed
- Master 8.0 or 9.0 needs to be installed as a VM from the Fuel ISO. The fuel ISO can be downloaded from the http://www.mirantis.com. Refer to: http://docs.openstack.org/developer/fuel-docs/userdocs/fuel-install-guide.html for instructions on how to install Fuel.
- The Fuel Master VM needs two bridged interfaces where one for the Fuel PXE network, the other for PODM communication.
- The Management Controller (MC) on the DSS 9000 must have firmware 3.15 or newer. The management interface must be set to DHCP.
- The hostname on the MC must include the string “psme”. For example: psme-mc-hostname.

Step-by-step tasks

1) Power on the DSS 9000.
2) The S4048-ON switch should be setup for PODM and Fuel network. Please refer to the Cumulus OS user guide on how to create multiple bridges.
3) On a Utility Node (R430), you should have two VMs running plus the Kickstart integration components for PODM.
   - PODM VM. This is a virtual box VM running Ubuntu 14.04 with all Kickstart components installed
   - Fuel Master VM. This is a virtual box VM that is installed via the ISO installer for Fuel
4) The utility node will have two networks (NIC1/NIC2). One connected to the PODM management network and another network is for the Fuel network
5) Login to the Utility node using your credential.
6) Using the Kickstart tools, allocate a workload from PODM. For example, you may have a need for 5 servers with specific CPU and memory requirements:

   python podm_tool.py node allocate “Example Workload” 5 CPU=16 MEM=64000

7) Once the allocation is complete, assemble the nodes to make the new nodes ready.

   python podm_tool.py node assemble 1-5

8) The PODM fuel agent will detect the newly allocated workload. The agent will automatically power-on the nodes, if they are off. The agent will automatically set the nodes to boot to PXE so all the new nodes can boot to Fuel’s PXE network.
9) Once the nodes are discovered by Fuel, they will each be identified by LOCALE information. For example, each node will be identified as RackXX, BlockXX, SledXX”. Where XX is the Rack, Block, and Sled location in the rack.
10) From within Fuel, you can navigate to the Kickstart firmware/BIOS update utility. You can simply update BIOS/Firmware/RAID on all nodes by running a simple script “update_firmware.sh” that will apply new
settings to all the newly discovered nodes from within Fuel. The settings can be specified in a text file describing location of the firmware/BIOS files and BIOS/Firmware settings to be applied to each node. For example, setting the virtualization flag in BIOS.

11) Now that the nodes are ready within Fuel, a new OpenStack environment can be created using the Fuel Web-UI and the newly discovered nodes can added to the cluster via standard Fuel methods.

Below is a flow chart illustration of the dynamic workload assignment process …

For more information

For more information about the DSS 9000 or this OpenStack kickstart process, contact your Dell EMC sales representative or send email to: ESI@dell.com.