



# Red Hat OpenStack Platform

## 9

# Dell EqualLogic Back End Guide

---

A Guide to Using Dell EqualLogic Storage in a Red Hat OpenStack Platform Overcloud

OpenStack Team



# Red Hat OpenStack Platform 9 Dell EqualLogic Back End Guide

---

## A Guide to Using Dell EqualLogic Storage in a Red Hat OpenStack Platform Overcloud

OpenStack Team  
rhos-docs@redhat.com

## Legal Notice

Copyright © 2017 Red Hat, Inc.

The text of and illustrations in this document are licensed by Red Hat under a Creative Commons Attribution–Share Alike 3.0 Unported license ("CC-BY-SA"). An explanation of CC-BY-SA is available at

<http://creativecommons.org/licenses/by-sa/3.0/>

. In accordance with CC-BY-SA, if you distribute this document or an adaptation of it, you must provide the URL for the original version.

Red Hat, as the licensor of this document, waives the right to enforce, and agrees not to assert, Section 4d of CC-BY-SA to the fullest extent permitted by applicable law.

Red Hat, Red Hat Enterprise Linux, the Shadowman logo, JBoss, OpenShift, Fedora, the Infinity logo, and RHCE are trademarks of Red Hat, Inc., registered in the United States and other countries.

Linux ® is the registered trademark of Linus Torvalds in the United States and other countries.

Java ® is a registered trademark of Oracle and/or its affiliates.

XFS ® is a trademark of Silicon Graphics International Corp. or its subsidiaries in the United States and/or other countries.

MySQL ® is a registered trademark of MySQL AB in the United States, the European Union and other countries.

Node.js ® is an official trademark of Joyent. Red Hat Software Collections is not formally related to or endorsed by the official Joyent Node.js open source or commercial project.

The OpenStack ® Word Mark and OpenStack logo are either registered trademarks/service marks or trademarks/service marks of the OpenStack Foundation, in the United States and other countries and are used with the OpenStack Foundation's permission. We are not affiliated with, endorsed or sponsored by the OpenStack Foundation, or the OpenStack community.

All other trademarks are the property of their respective owners.

## Abstract

This document describes how to deploy a single Dell EqualLogic device as a back end to the Red Hat OpenStack Platform 9 Overcloud.

---

## Table of Contents

<b>1. INTRODUCTION</b>	<b>2</b>
<b>2. PROCESS DESCRIPTION</b>	<b>2</b>
<b>3. DEFINE A SINGLE BACK END</b>	<b>3</b>
<b>4. DEPLOY THE CONFIGURED BACK END</b>	<b>5</b>
<b>5. TEST THE CONFIGURED BACK END</b>	<b>6</b>
<b>6. ADDRESS VOLUME SIZE DISCREPANCIES WITH DELL EQUALLOGIC BACK ENDS</b>	<b>6</b>
6.1. Example	7
6.2. Workaround	8

## 1. INTRODUCTION

This document describes how to configure OpenStack to use one or more Dell EqualLogic back ends. It also includes instructions on addressing volume size discrepancies between Dell EqualLogic devices and the OpenStack Block Storage service.

The following sections assume that:

- ✦ You intend to use only Dell EqualLogic devices and drivers for Block Storage back ends
- ✦ The OpenStack Overcloud has already been deployed through Director, with a properly-functional Block Storage service
- ✦ The Dell storage device has already been deployed and configured as a storage repository
- ✦ A Dell EqualLogic Group is already deployed and accessible through SSH
- ✦ You have the necessary credentials for connecting to the Group manager of the available Dell EqualLogic Group (namely, CHAP and Group manager credentials)
- ✦ You have the username and password of an account with elevated privileges. You can use the same account that was created to deploy the Overcloud; in [Creating a Director Installation User](#), we create and use the **stack** user for this purpose.

When RHEL OpenStack Platform is deployed through the Director, all major Overcloud settings (in particular, the Block Storage service back end) must be defined and orchestrated through the Director as well. This ensures that the settings will persist through any further Overcloud updates. For more information about deploying OpenStack through the Director, see [Director Installation and Usage](#).

The purpose of this document is to explain how to orchestrate your desired Dell EqualLogic back end configuration to the Overcloud's Block Storage service. This document will not discuss the different deployment configurations possible with the back end. Rather, to learn more about the different available deployment configurations, see your device's product documentation.

Once you are familiar with the resulting back end configuration you want to deploy (and its corresponding settings), refer to this document for instructions on how to orchestrate it through the Director.



### Note

At present, the Director only has the integrated components to deploy a **single** instance of a Dell EqualLogic back end. As such, this document only describes the deployment of a single back end.

Deploying multiple instances of a Dell EqualLogic back end requires a *custom back end configuration*. See the [Custom Block Storage Back End Deployment Guide](#) for instructions.

## 2. PROCESS DESCRIPTION

RHEL OpenStack Platform includes all the drivers required for all Dell devices supported by the Block Storage service. In addition, the Director also has the puppet manifests, environment files, and Orchestration templates necessary for integrating the device as a back end to the Overcloud.

Configuring a [single Dell device as a back end](#) involves editing the default **environment file** and including it in the Overcloud deployment. This file is available locally on the Undercloud, and can be edited to suit your environment.

After editing this file, invoke it through the Director. Doing so ensures that it will persist through future Overcloud updates. The following sections describe this process in greater detail. In addition, the default environment file already contains enough information to call the necessary puppet manifests and Orchestration (Heat) templates that will configure the rest of the required Block Storage settings.

### 3. DEFINE A SINGLE BACK END



#### Important

This section describes the deployment of a single back end. Deploying multiple instances of a Dell EqualLogic back end requires a *custom back end configuration*. See the [Custom Block Storage Back End Deployment Guide](#) for instructions.

With a Director deployment, the easiest way to define a **single** Dell EqualLogic back end is through the integrated environment file. This file is located in the following path of the Undercloud node:

**`/usr/share/openstack-tripleo-heat-templates/environments/cinder-eqlx-config.yaml`**

Copy this file to a local path where you can edit and invoke it later. For example, to copy it to `~/templates/`:

```
$ cp /usr/share/openstack-tripleo-heat-templates/environments/cinder-eqlx-config.yaml ~/templates/
```

Afterwards, open the copy (`~/templates/cinder-eqlx-config.yaml`) and edit it as you see fit. The following snippet displays the default contents of this file:

```
# A Heat environment file which can be used to enable a
# a Cinder eqlx backen, configured via puppet
resource_registry:
  OS::TripleO::ControllerExtraConfigPre:
    ../puppet/extraconfig/pre_deploy/controller/cinder-eqlx.yaml # 1

parameter_defaults: # 2
  CinderEnableEqlxBackend: true # 3
  CinderEqlxBackendName: 'tripleo_eqlx'
  CinderEqlxSanIp: ''
  CinderEqlxSanLogin: ''
  CinderEqlxSanPassword: ''
  CinderEqlxSanThinProvision: true
  CinderEqlxGroupname: 'group-0'
  CinderEqlxPool: 'default'
  CinderEqlxChapLogin: ''
  CinderEqlxChapPassword: ''
  CinderEqlxUseChap: false
```

The **OS::TripleO::ControllerExtraConfigPre:** parameter in the **resource\_registry** section refers to a Heat template named **cinder-eqlx.yaml**. This is the template that the Director should use to load the necessary resources for configuring the back end. By default, the parameter specifies the path to **cinder-eqlx.yaml** relatively. As such, update this parameter with the absolute path to the file:

```
resource_registry:
  OS::TripleO::ControllerExtraConfigPre: /usr/share/openstack-tripleo-heat-templates/puppet/extraconfig/pre_deploy/controller/cinder-eqlx.yaml
```

2

The **parameter\_defaults** section contains your back end definition. Specifically, it contains the parameters that the Director should pass to the resources defined in **cinder-eqlx.yaml**.

3

The **CinderEnableEqlxBackend: true** line instructs the Director to use the puppet manifests necessary for the default configuration of a Dell EqualLogic back end. This includes defining the volume driver that the Block Storage service should use (specifically, **cinder.volume.drivers.eqlx.DellEQLSanISCSIDriver**).

To define your Dell EqualLogic back end, edit the settings in the **parameter\_defaults** section as you see fit. The following table explains each parameter, and also lists its corresponding **/etc/cinder/cinder.conf** setting.

**Table 1. Dell EqualLogic settings**

Parameter	/etc/cinder/cinder.conf setting	Description
CinderEqlxBackendName	volume_backend_name	An arbitrary name to identify the volume back end.
CinderEqlxSanIp	san_ip	The IP address used to reach the Dell EqualLogic Group through SSH.
CinderEqlxSanLogin	san_login	The user name to login to the Group manager via SSH at the <b>CinderEqlxSanIp</b> . The default user name is <b>grpadmin</b> .
CinderEqlxSanPassword	san_password	The corresponding password of <b>CinderEqlxSanLogin</b> . The default password is <b>password</b> .



Parameter	/etc/cinder/cinder.conf setting	Description
CinderEqLxSanThinProvision	san_thin_provision	Sets whether thin provisioning for SAN volumes is enabled ( <b>true</b> ), as is required for this setup.
CinderEqLxGroupname	eqlx_group_name	The group to be used for a pool where the Block Storage service will create volumes and snapshots. The default group is <b>group-0</b> .
CinderEqLxPool	eqlx_pool	The pool where the Block Storage service will create volumes and snapshots. This option cannot be used for multiple pools utilized by the Block Storage service on a single Dell EqualLogic Group. The default pool is <b>default</b> .
CinderEqLxChapLogin	eqlx_chap_login	The CHAP login account for each volume in a pool. The default account name is <b>chapadmin</b> .
CinderEqLxChapPassword	eqlx_chap_password	The corresponding password of <b>CinderEqLxChapLogin</b> . The default password is randomly generated in hexadecimal, so you must set this password manually.
CinderEqLxUseChap	eqlx_use_chap	Sets whether CHAP authentication is disabled ( <b>false</b> by default) or enabled ( <b>true</b> ).

## 4. DEPLOY THE CONFIGURED BACK END

The Director installation uses a non-root user to execute commands, which includes orchestrating the deployment of the Block Storage back end. In [Creating a Director Installation User](#), we create a user named **stack** for this purpose. This user is configured with elevated privileges.

To deploy the lone back end configured in [Section 3, "Define a Single Back End"](#), first log in as the **stack** user to the Undercloud. Then, deploy the back end (defined in the edited `~/templates/cinder-eqlx-config.yaml`) by running the following:

```
$ openstack overcloud deploy --templates -e ~/templates/cinder-eqlx-config.yaml
```



### Important

If you passed any extra environment files when you created the Overcloud, pass them again here using the `-e` option to avoid making undesired changes to the Overcloud.

For more information, see [Scaling the Overcloud](#) and [Updating the Overcloud Packages](#).

Once the Director completes the orchestration, test the back end. See [Section 5, “Test the Configured Back End”](#) for instructions.

## 5. TEST THE CONFIGURED BACK END

After deploying the back end, test whether you can successfully create volumes on it. Doing so will require loading the necessary environment variables first. These variables are defined in `/home/stack/overcloudrc` by default.

To load these variables, run the following command as the `stack` user:

```
$ source /home/stack/overcloudrc
```



### Note

For more information, see [Accessing the Basic Overcloud](#).

You should now be logged in to the Controller node. From there, you can create a *volume type*, which can be used to specify the back end you want to use (in this case, the newly-defined back end in [Section 3, “Define a Single Back End”](#)). This is required in an OpenStack deployment where you have other back ends enabled (preferably, also through Director).

To create a volume type named `delleql`, run:

```
$ cinder type-create delleql
```

Next, map this volume type to the back end defined in [Section 3, “Define a Single Back End”](#). Given the back end name `tripleo_eq1x` (as defined through the `CinderEq1xBackendName` parameter, in [Section 3, “Define a Single Back End”](#)), run:

```
$ cinder type-key delleql set volume_backend_name=tripleo_eq1x
```

You should now be able to create a 2GB volume on the newly defined back end by invoking its volume type. To do so, run:

```
$ cinder create --volume-type delleql 2
```

## 6. ADDRESS VOLUME SIZE DISCREPANCIES WITH DELL EQUALLOGIC BACK ENDS

When reporting volume sizes, Dell EqualLogic (EQL) back ends also account for additional storage to be used for internal volume metadata. This size will be slightly larger than the volume size reported by the Block Storage services. However, the volume size reported by an EQL back end is the same one used by the Image service.

As a result, when creating an image-backed volume on an EQL back end, check the size of the image first. If the image was originally volume-backed, then EQL (and, by extension, the Image service) will be reporting a volume size slightly larger than what is reported by the Block Storage service.

If the image size reported by EQL is slightly larger, then you need to take the size discrepancy into consideration when creating volumes backed by this image.

## 6.1. Example

To illustrate, when you create a 1GB volume:

```
# cinder create --display-name vol1 1
```

Property	Value
attachments	[]
availability_zone	nova
bootable	false
created_at	2014-12-19T03:57:47.730359
display_description	None
display_name	vol1
encrypted	False
id	6bdace69-bd41-42fc-a63a-f834fb65a2e4
metadata	{}
size	1
snapshot_id	None
source_volid	None
status	creating
volume_type	None

The Block Storage service will report a volume size of 1GB, but on the EQL array the size (VolReserve) will be slightly bigger:

```
eq1> volume select volume-6bdace69-bd41-42fc-a63a-f834fb65a2e4
```

```
eq1 (volume-6bdace69-bd41-42fc-a63a-f834fb65a2e4)> show
```

```

_____ Volume Information _____...
Name: volume-6bdace69-bd41-42fc-a63a-f834fb65a2e4
Size: 1GB
VolReserve: 1.01GB
...

```

When you create a new image from this volume, **cinder** will report a correct volume size of 1GB:

```
# cinder upload-to-image --disk-format raw --container-format bare vol1
image_vol1
```

```
+-----+-----+
```

Property	Value
container_format	bare
disk_format	raw
display_description	None
id	6bdace69-bd41-42fc-a63a-f834fb65a2e4
image_id	c65f7eae-e2c1-44ba-8af1-e33695897559
image_name	image_vol1
size	1
status	uploading
updated_at	2014-12-19T03:57:48.000000
volume_type	None

However, the Image service will report a slightly larger size:

```
# glance image-list
```

```
...+-----+-----+-----+-----+
...| Name          | Disk Format | Container Format | Size          | Status |
...+-----+-----+-----+-----+
...| image_vol1    | raw        | bare            | 1085276160   | active |
...+-----+-----+-----+-----+
```

The **glance** tool reports an image size of approximately 1.01GB. As a result, creating a new 1GB volume backed by this image will fail:

```
# cinder create --display-name vol2 --image-id c65f7eae-e2c1-44ba-8af1-
e33695897559 1
```

```
ERROR: Invalid input received: Size of specified image 2 is larger than
volume size 1
```

## 6.2. Workaround

As mentioned earlier, you need to consider the discrepancy between the volume sizes reported by the Image and the Block Storage services when specifying the size of image-backed volumes. This means that when specifying the size of the image-backed volume, use the **next whole number after the image size reported by glance**.

Using the previous example, **glance** reported an image size of 1.01GB. This means that when you create a volume, you need to specify a volume size of 2GB instead of 1GB:

```
# cinder create --display-name vol2 --image-id c65f7eae-e2c1-44ba-8af1-
e33695897559 2
```

Property	Value
attachments	[]
availability_zone	nova
bootable	false
created_at	2014-12-19T04:54:07.036260
display_description	None
display_name	vol2

encrypted	False
id	fcf49715-094d-4bba-9f05-8b7fa6deffce
image_id	c65f7eae-e2c1-44ba-8af1-e33695897559
metadata	{}
size	2
snapshot_id	None
source_volid	None
status	creating
volume_type	None

+-----+