



Red Hat OpenStack Platform 10 Fujitsu ETERNUS Back End Guide

A Guide to Using a Fujitsu ETERNUS Back End in a Red Hat OpenStack Platform 10 Environment

OpenStack Team

A Guide to Using a Fujitsu ETERNUS Back End in a Red Hat OpenStack Platform 10 Environment

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 configure Red Hat OpenStack Platform 10 to use a Fujitsu ETERNUS Disk Storage System as a back end.

Table of Contents

1. INTRODUCTION	2
2. CONFIGURE THE FUJITSU ETERNUS DEVICE	2
3. PREPARE THE HEAT TEMPLATE	3
3.1. Create Driver Definitions for Each Back End	4
3.2. Sample Completed Heat Template	6
4. CREATE THE ENVIRONMENT FILE	7
5. DEPLOY THE CONFIGURED BACK ENDS	10
6. TEST YOUR CONFIGURATION	10

1. INTRODUCTION

This document describes how to configure OpenStack to use a Fujitsu ETERNUS Disk Storage System as a back end for the Block Storage service. In particular, this document covers how to define a fibre channel and iSCSI back end provided by an ETERNUS device on an overcloud deployment. This will involve defining both back ends as a *custom back end* for the Block Storage service.

The following sections assume that:

- ✱ A Red Hat OpenStack Platform overcloud has already been deployed through the director.
- ✱ You intend to use only Fujitsu ETERNUS Disk Storage System devices and drivers for Block Storage back ends.
- ✱ You can use the *director installation user*, which is created as part of the overcloud deployment. See [Creating a Director Installation User](#) (from [Director Installation and Usage](#)) for more information.
- ✱ You have access to an **Admin** account on the ETERNUS device; that is, you can log in using an account with the **Admin** role through the ETERNUS Web GUI or CLI.
- ✱ The Block Storage will be installed on Controller nodes (default).

You can use either Fibre Channel or iSCSI interfaces with a Fujitsu ETERNUS device. Each interface has its own settings and driver: Red Hat supports the use of both interfaces (and their respective drivers) with OpenStack.



Note

For related details on how to define a custom back end, see [Custom Block Storage Back End Deployment Guide](#).

2. CONFIGURE THE FUJITSU ETERNUS DEVICE

Before you can define the Fujitsu ETERNUS device as a Block Storage back end, you need to configure storage pools and ports on the device first. Consult your device documentation for details on each step:

1. Set up a LAN connection between the Controller nodes (where the Block Storage service is hosted) and MNT ports of the ETERNUS device.
2. Set up a SAN connection between the Compute nodes and CA ports of the ETERNUS device.
3. Log in to the ETERNUS device using an account with the **Admin** role.
4. Enable the SMI-S of ETERNUS DX.
5. Register an **Advanced Copy Feature** license and configure the copy table size.
6. Create a storage pool for volumes. This pool will be used later in the **EternusPool** setting in [Section 3.1, “Create Driver Definitions for Each Back End”](#).



Note

If you want to create volume snapshots on a different storage pool, create a storage pool for that as well. This pool will be used in the **EternusSnapPool** setting in [Section 3.1](#), “Create Driver Definitions for Each Back End”.

7. Create a *Snap Data Pool Volume (SDPV)* to enable Snap Data Pool (SDP) for the **create a snapshot** function.
8. Configure *storage ports* to be used by the Block Storage service. Then:
 - a. Set those ports to **CA** mode.
 - b. Enable the **host-affinity** settings of those storage ports. To enable **host-affinity**, run the following from the ETERNUS CLI for each port:

```
CLI> set PROTO-parameters -host-affinity enable -port CM# CA#
PORT
```

Where: * *PROTO* defines which storage protocol is in use, as in **fc** (Fibre Channel) or **iscsi**.
 * *CM# CA#* refer to the controller enclosure where the port is located. * *PORT* is the port number.

3. PREPARE THE HEAT TEMPLATE

When deploying a Red Hat OpenStack Platform overcloud, it is advisable to perform all service configuration through the director. This ensures that your settings persist throughout future updates to the overcloud.

An ETERNUS back end requires the following on the Controller node (where the Block Storage service is hosted):

- ✱ The **pywbem** package should be installed.
- ✱ An XML configuration file for the driver settings of each back end.

Both tasks can be orchestrated by the director through a *heat template*. For more information on the syntax of heat templates used for the director, see [Customizing Overcloud Pre-Configuration](#).

The following template (*eternus-temp.yaml*) contains the basic syntax for the required heat template. It also includes instructions for installing the required **pywbem** package:

eternus-temp.yaml

```
heat_template_version: 2014-10-16

description: >
  Installs the pywbem package on all Controller nodes

parameters:
  server:
    type: string

resources:
```

```

EternusSetup: # 1
type: OS::Heat::SoftwareConfig
properties:
  group: script
  config: | # 2
    #!/bin/bash
    sudo yum install pywbem -y
    # 3

ExtraPreDeployment:
type: OS::Heat::SoftwareDeployment
properties:
  config: {get_resource: EternusSetup}
  server: {get_param: server}
  actions: [CREATE, UPDATE]

```

1

EternusSetup declares the resource that will orchestrate the tasks we need to run on the Controller node.

2

The **config** section contains all the commands we need to run on the Controller node. At the moment, it only contains the command for installing **pywbem**.

3

We will add commands here for creating the XML configuration files for the driver settings of each back end in [Section 3.1, “Create Driver Definitions for Each Back End”](#).

Store this file in `/home/stack/templates/` of the director node. On a typical Red Hat OpenStack Platform director deployment, custom heat templates are stored here.

3.1. Create Driver Definitions for Each Back End

Driver settings for each ETERNUS back end are defined on separate XML files rather than the Block Storage configuration file (`/etc/cinder/cinder.conf`). Each back end must have its own XML file. Each XML file should contain the following settings:

EternusIP

IP address of the SMI-S connection of the ETERNUS device. Specifically, use the IP address of the MNT port of the device.

EternusPort

port number for the SMI-S connection port of the ETERNUS device.

EternusUser

User name to be used for the SMI-S connection (**EternusIP**).

EternusPassword

Corresponding password of **EternusUser** on **EternusIP**.

EternusPool

Name of the storage pool created for volumes (from [Section 2, “Configure the Fujitsu ETERNUS device”](#)). Specifically, use the pool’s RAID Group name or TPP name in the ETERNUS device.

EternusSnapPool

Name of the storage pool created for volume snapshots (from [Section 2, “Configure the Fujitsu ETERNUS device”](#)). Specifically, use the pool’s RAID Group name in the ETERNUS device. If you did not create a different pool for snapshots, use the same value as **EternusPool**.

EternusISCSIIP

(iSCSI only) IP address for iSCSI connections to the ETERNUS device. You can specify multiple IPs by creating an entry for each one.

For example, to define a fibre-channel configuration:

eternus-fc.xml

```
<?xml version='1.0' encoding='UTF-8'?>
<FUJITSU>
<EternusIP>0.0.0.0</EternusIP>
<EternusPort>5988</EternusPort>
<EternusUser>smisuser</EternusUser>
<EternusPassword>smispassword</EternusPassword>
<EternusPool>raid5_0001</EternusPool>
<EternusSnapPool>raid5_0001</EternusSnapPool>
</FUJITSU>
```

For an iSCSI configuration with 4 iSCSI connections:

eternus-iscsi.xml

```
<?xml version='1.0' encoding='UTF-8'?>
<FUJITSU>
<EternusIP>0.0.0.0</EternusIP>
<EternusPort>5988</EternusPort>
<EternusUser>smisuser</EternusUser>
<EternusPassword>smispassword</EternusPassword>
<EternusPool>raid5_0001</EternusPool>
<EternusSnapPool>raid5_0001</EternusSnapPool>
<EternusISCSIIP>1.1.1.1</EternusISCSIIP>
<EternusISCSIIP>1.1.1.2</EternusISCSIIP>
<EternusISCSIIP>1.1.1.3</EternusISCSIIP>
<EternusISCSIIP>1.1.1.4</EternusISCSIIP>
</FUJITSU>
```

To orchestrate the creation of these XML files, use a bash commands. These commands need to be added to the **config** section of the **EternusSetup** resource in `/home/stack/templates/eternus-temp.yaml` (from [Section 3, “Prepare the Heat Template”](#)). For example, to orchestrate the creation of `eternus-fc.xml` and `eternus-iscsi.xml`:

■

```
sudo cat > /etc/cinder/eternus-fc.xml <<EOF
<?xml version=1.0 encoding=UTF-8?>
<FUJITSU>
<EternusIP>0.0.0.0</EternusIP>
<EternusPort>5988</EternusPort>
<EternusUser>smisuser</EternusUser>
<EternusPassword>smispassword</EternusPassword>
<EternusPool>raid5_0001</EternusPool>
<EternusSnapPool>raid5_0001</EternusSnapPool>
</FUJITSU>
EOF
```

```
sudo cat > /etc/cinder/eternus-iscsi.xml <<EOF
<?xml version=1.0 encoding=UTF-8?>
<FUJITSU>
<EternusIP>0.0.0.0</EternusIP>
<EternusPort>5988</EternusPort>
<EternusUser>smisuser</EternusUser>
<EternusPassword>smispassword</EternusPassword>
<EternusPool>raid5_0001</EternusPool>
<EternusSnapPool>raid5_0001</EternusSnapPool>
<EternusISCSIIP>1.1.1.1</EternusISCSIIP>
<EternusISCSIIP>1.1.1.2</EternusISCSIIP>
<EternusISCSIIP>1.1.1.3</EternusISCSIIP>
<EternusISCSIIP>1.1.1.4</EternusISCSIIP>
</FUJITSU>
EOF
```

Use **sudo cat** accordingly to orchestrate the creation of as many XML configuration files as you need.

The XML files also need to be owned by the **cinder** user and group, and only usable by the owner. As such, you also need to orchestrate the ownership and permissions of these files:

```
sudo chown cinder:cinder /etc/cinder/eternus-*.xml
sudo chmod 0600 /etc/cinder/eternus-*.xml
```

See [Section 3.2, “Sample Completed Heat Template”](#) for an example of a completed heat template.

3.2. Sample Completed Heat Template

The following `/home/stack/templates/eternus-temp.yaml` file contains all the necessary components for installing the required **pywbem** package and declaring the example XML configuration files from [Section 3.1, “Create Driver Definitions for Each Back End”](#) (namely, `eternus-fc.xml` and `eternus-iscsi.xml`):

`/home/stack/templates/eternus-temp.yaml`

```
heat_template_version: 2014-10-16

description: >
  Installs the pywbem package on all Controller nodes

parameters:
  server:
    type: string
```

```

resources:
  EternusSetup:
    type: OS::Heat::SoftwareConfig
    properties:
      group: script
      config: |
        #!/bin/bash
        sudo yum install pywbem -y
        sudo cat > /etc/cinder/eternus-fc.xml <<EOF
        <?xml version=1.0 encoding=UTF-8?>
        <FUJITSU>
        <EternusIP>0.0.0.0</EternusIP>
        <EternusPort>5988</EternusPort>
        <EternusUser>smisuser</EternusUser>
        <EternusPassword>smispassword</EternusPassword>
        <EternusPool>raid5_0001</EternusPool>
        <EternusSnapPool>raid5_0001</EternusSnapPool>
        </FUJITSU>
        EOF
        sudo cat > /etc/cinder/eternus-iscsi.xml <<EOF
        <?xml version=1.0 encoding=UTF-8?>
        <FUJITSU>
        <EternusIP>0.0.0.0</EternusIP>
        <EternusPort>5988</EternusPort>
        <EternusUser>smisuser</EternusUser>
        <EternusPassword>smispassword</EternusPassword>
        <EternusPool>raid5_0001</EternusPool>
        <EternusSnapPool>raid5_0001</EternusSnapPool>
        <EternusISCSIIP>1.1.1.1</EternusISCSIIP>
        <EternusISCSIIP>1.1.1.2</EternusISCSIIP>
        <EternusISCSIIP>1.1.1.3</EternusISCSIIP>
        <EternusISCSIIP>1.1.1.4</EternusISCSIIP>
        </FUJITSU>
        EOF
        sudo chown cinder:cinder /etc/cinder/eternus-*.xml
        sudo chmod 0600 /etc/cinder/eternus-*.xml

  ExtraPreDeployment:
    type: OS::Heat::SoftwareDeployment
    properties:
      config: {get_resource: EternusSetup}
      server: {get_param: server}
      actions: [CREATE, UPDATE]

```

4. CREATE THE ENVIRONMENT FILE

The environment file contains the settings for each back end you want to define. It also contains other settings relevant to the deployment of a custom back end. For more information about environment files, see [Environment Files](https://www.redhat.com/en/documentation/red-hat-openstack-platform/10/single/advanced-overcloud-customization/[Advanced Overcloud Customization] guide) (in the [redhat.com/documentation/en/red-hat-openstack-platform/10/single/advanced-overcloud-customization/\[Advanced Overcloud Customization\]](https://www.redhat.com/en/documentation/red-hat-openstack-platform/10/single/advanced-overcloud-customization/[Advanced Overcloud Customization] guide) guide).

In addition, the environment file will also register the heat template you created earlier in [Section 3, “Prepare the Heat Template”](#). In doing so, the installation and echo commands defined in that template will be carried out on the appropriate nodes.

The following sample environment file contains all the necessary sections for defining an ETERNUS device as a Block Storage back end. It also creates the back end definitions for each corresponding XML file orchestrated in [Section 3.1, “Create Driver Definitions for Each Back End”](#) and [Section 3.2, “Sample Completed Heat Template”](#):

eternusbackend-env.yaml

```
parameters: # 1
  CinderEnableIscsiBackend: false
  CinderEnableRbdBackend: false
  CinderEnableNfsBackend: false
  NovaEnableRbdBackend: false
  GlanceBackend: file # 2

resource_registry:
  OS::TripleO::NodeExtraConfig: /home/stack/templates/eternus-temp.yaml # 3

parameter_defaults:
  controllerExtraConfig: # 4
    cinder::config::cinder_config:
      FJFC/volume_driver: # 5
        value: cinder.volume.drivers.fujitsu.eternus_dx_fc.FJDXFCDriver
      FJFC/cinder_eternus_config_file: # 6
        value: /etc/cinder/eternus-fc.xml
      FJFC/volume_backend_name: # 7
        value: FJFC
      FJISCSI/volume_driver: # 8
        value:
cinder.volume.drivers.fujitsu.eternus_dx_iscsi.FJDXISCSIDriver
      FJISCSI/cinder_eternus_config_file:
        value: /etc/cinder/eternus-iscsi.xml
      FJISCSI/volume_backend_name:
        value: FJISCSI
    cinder_user_enabled_backends: [FJFC, FJISCSI] # 9
```

1

The following parameters are set to **false**, and thereby disable other back end types:

- ✧ **CinderEnableIscsiBackend**: other iSCSI back ends.
- ✧ **CinderEnableRbdBackend**: Red Hat Ceph.
- ✧ **CinderEnableNfsBackend**: NFS.
- ✧ **NovaEnableRbdBackend**: ephemeral Red Hat Ceph storage.

2

The **GlanceBackend** parameter sets what the Image service should use to store images. The following values are supported:

remaining values are supported:

- ✧ **file**: store images on **/var/lib/glance/images** on each Controller node.
- ✧ **swift**: use the Object Storage service for image storage.
- ✧ **cinder**: use the Block Storage service for image storage.

3

NodeExtraConfig defines custom settings that will be applied to all nodes **before** the core Puppet configuration. This ensures that by the time the Block Storage service is deployed on the overcloud:

- ✧ The Controller node already has the **pywbem** package installed, and
- ✧ The XML configuration files for each back end are already created.

4

controllerExtraConfig defines custom settings that will be applied to all controller nodes. The **cinder::config::cinder_config** class means the settings should be applied to the Block Storage (**cinder**) service. This, in turn, means that the back end settings will ultimately end in the **/etc/cinder/cinder.conf** file of each node.

5

The **FJFC/** string creates a back end definition named **FJFC**, and the following parameter will be declared under that back end definition. The **volume_driver** parameter sets the specific ETERNUS driver for the back end; in this case, **cinder.volume.drivers.fujitsu.eternus_dx_fc.FJDXFCDriver** sets the fibre channel driver.

6

The **cinder_eternus_config_file** sets the path to the XML configuration file that the driver should use for the back end. The creation of **/etc/cinder/eternus-fc.xml** is orchestrated through the heat template (namely, **/home/stack/templates/eternus-temp.yaml**).

7

The **volume_backend_name** is the name that the Block Storage service should use to enable the back end.

8

The **FJISCSI/** string creates a new back end definition, in the same manner as **FJFC** earlier.

9

The **cinder_user_enabled_backends** class sets and enables custom back ends. As the name implies, this class should only be used for user-enabled back ends; specifically, those defined in the

cinder::config::cinder_config class.

After creating the environment file, you can now deploy your configuration. See [Section 5, “Deploy the Configured Back Ends”](#) for details on how to use the environment file `/home/stack/templates/eternusbackend-env.yaml` for this purpose.

5. DEPLOY THE CONFIGURED BACK ENDS

Once you have created the [custom-env.yaml](#) file in `/home/stack/templates/`, log in as the **stack** user. Then, deploy the back end configuration by running:

```
$ openstack overcloud deploy --templates -e
/home/stack/templates/eternusbackend-env.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 [Modifying the Overcloud Environment](#) (from [Director Installation and Usage](#)).

Once the Director completes the orchestration, test the back end. See [Section 6, “Test Your Configuration”](#) for instructions.

6. TEST YOUR CONFIGURATION

After configuring the Block Storage service to use the new ETERNUS back ends, declare a *volume type* for each one. Volume types allow you to specify which back end to use when creating new volumes. The following commands create two volume types: **FJFC** (for the fibre channel back end) and **FJISCSI** (for the iSCSI back end):

```
# cinder type-create FJFC
# cinder type-create FJISCSI
```

Next, map these volume types to their respective back ends (as defined in [Section 4, “Create the Environment File”](#)):

```
# cinder type-key FJFC set volume_backend_name=FJFC
# cinder type-key FJISCSI volume_backend_name=FJISCSI
```

Verify your configuration by creating a 1GB iSCSI volume named **test_iscsi**:

```
# cinder create --volume_type FJISCSI --display_name test_iscsi 1
```

To test the fibre channel back end:

```
# cinder create --volume_type FJFC --display_name test_fc 1
```