Red Hat OpenStack Platform 16.1

Fujitsu ETERNUS Back End Guide

A guide to using a Fujitsu ETERNUS back end in a Red Hat OpenStack Platform environment

Red Hat OpenStack Platform 16.1 Fujitsu ETERNUS Back End Guide

A guide to using a Fujitsu ETERNUS back end in a Red Hat OpenStack Platform environment

OpenStack Team
rhos-docs@redhat.com

Legal Notice

Abstract

This document describes how to configure Red Hat OpenStack Platform 16.1 to use a Fujitsu ETERNUS Disk Storage System as a back end.

Table of Contents

CHAPTER 1. INTRODUCTION

This document describes how to configure Red Hat OpenStack Platform to use a Fujitsu ETERNUS Disk Storage System as a back end for the Block Storage service. The document covers how to define a Fibre Channel and iSCSI back end provided by an ETERNUS device on an overcloud deployment. This process involves defining both back ends as a **custom back end** for the Block Storage service. By default, Controller nodes contain the Block Storage service.

Prerequisites

- 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*, that you create with the overcloud deployment. For more information about creating the stack user, see Preparing the undercloud in the *Director Installation and Usage* guide.

- You have access to an **Admin** account on the ETERNUS device through the ETERNUS Web GUI or CLI.

Red Hat supports using Fibre Channel or iSCSI interfaces, and the respective drivers and settings, with a Fujitsu ETERNUS device.

> **NOTE**
>
> For more information about defining a custom back end, see the *Custom Block Storage Back End Deployment Guide*.

CHAPTER 2. CONFIGURING THE FUJITSU ETERNUS DEVICE

Configure storage pools and ports on the device before you define the Fujitsu ETERNUS device as a Block Storage back end. Consult your device documentation for details on each step:

Procedure

1. Configure a LAN connection between the Controller nodes that host the Block Storage service and the MNT ports of the ETERNUS device.

2. Configure 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. Set the SSH key

6. Add a user account with **software** role.

7. Register an **Advanced Copy Feature** license and configure the copy table size.

8. Create a storage pool for volumes. You use this pool later in the **EternusPool** setting in Section 3.1, "Creating driver definitions for each Fujitsu ETERNUS back end" .

9. Optional: Create a separate storage pool for volume snapshots. This pool represents the **EternusSnapPool** setting in Section 3.1, "Creating driver definitions for each Fujitsu ETERNUS back end".

10. Optional: Create a **Snap Data Pool Volume (SDPV)** to enable Snap Data Pool (SDP) for the **create a snapshot** function.

    TIP

    Skip this step if you use a Thin Provision Pool for **EternusSnapPool**.

11. Configure **storage ports** to be used by the Block Storage service.

12. Set the **storage ports** ports to **CA** mode.

13. To enable **host-affinity** for the storage ports, enter the following command from the ETERNUS CLI for each port:

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

    - Replace <PROTO> with the storage protocol, such as **fc** or **iscsi**.

    - Replace <CM#> and <CA#> with the name of the controller enclosure where the port is located.

    - Replace <PORT> with the port number.

14. Use the following commands to generate the SSH key on the undercloud, and upload the **eternus.ietf** file to the ETERNUS device.

```
ssh-keygen -t rsa -N "" -f ./eternus
ssh-keygen -e -f ./eternus.pub > ./eternus.ietf
```

NOTE

Save the **eternus** file for later use. For security reasons, do not copy or move it from the undercloud or overcloud.

CHAPTER 3. PREPARING THE FUJITSU ETERNUS HEAT TEMPLATE

To ensure that your settings persist throughout future updates to the Red Hat OpenStack Platform overcloud, perform all service configuration during deployment through director.

Include the following configuration on the Controller node of the ETERNUS back end that hosts the Block Storage service:

- You include an XML configuration file for the driver settings of each back end.

- You include an SSH server key to communicate with ETERNUS device.

You can orchestrate both tasks with director using a heat template. For more information about the syntax of director heat templates, see Understanding Heat Templates in the *Advanced Overcloud Guide*.

The following template, **eternus-temp.yaml**, outlines the basic syntax for the required heat template.

eternus-temp.yaml

```
heat_template_version: 2014-10-16

description: >
    Add XML configuration file for the driver settings of each back end

parameters:
  server:
    type: string

resources:
  EternusSetup: #  ❶
    type: OS::Heat::SoftwareConfig
    properties:
      group: script
      config: |  #  ❷
        #!/bin/bash
        #  ❸

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

❶ The **EternusSetup** section contains the resource that orchestrates the tasks on the Controller node.

❷ The **config** section contains the commands to run on the Controller node.

❸ Copy the private key information to each Controller node where the Block Storage service is hosted, and add commands to create the XML configuration files for the driver settings of each back end in Section 3.1, "Creating driver definitions for each Fujitsu ETERNUS back end" .

Store this file in the custom heat template directory on the director node, **/home/stack/templates/**.

3.1. CREATING DRIVER DEFINITIONS FOR EACH FUJITSU ETERNUS BACK END

Define driver settings for each ETERNUS back end on separate XML files, not the Block Storage configuration file **/etc/cinder/cinder.conf**. Ensure that each back end has an XML file, with the following settings:

EternusIP

IP address of the SMI-S connection of the ETERNUS device. 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 of **software** role for the connection **EternusIP**.

EternusPassword

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

EternusPool

Name of the storage pool for the volumes from Chapter 2, *Configuring the Fujitsu ETERNUS device*. Use the pool RAID Group name or TPP name in the ETERNUS device.

EternusSnapPool

Name of the storage pool for the volume snapshots from Chapter 2, *Configuring the Fujitsu ETERNUS device*. Use the pool RAID Group name in the ETERNUS device. If you did not create a different pool for snapshots, use the same value as **EternusPool**.

Define a Fibre Channel configuration with the following xml example:

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>
```

Define an ISCSI configuration with the following xml example:

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>
</FUJITSU>
```

To orchestrate the creation of these XML files, include bash commands in the **config** section of the **EternusSetup** resource in the **/home/stack/templates/eternus-temp.yaml** file from Chapter 3, *Preparing the Fujitsu ETERNUS heat template* . Orchestrate the creation of **eternus-fc.xml** and **eternus-iscsi.xml** with the following example command:

```
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>
</FUJITSU>
EOF
```

Use the **sudo cat** command to create the required amount of XML configuration files.

Set the ownership and permissions of these XML files for the **cinder** user and group.

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

For an example of a completed heat template, see Section 3.2, "Example Fujitsu ETERNUS heat template".

3.2. EXAMPLE FUJITSU ETERNUS HEAT TEMPLATE

The following **/home/stack/templates/eternus-temp.yaml** file contains the necessary parameters for declaring the example XML configuration files, such as **eternus-fc.xml** and **eternus-iscsi.xml**:

/home/stack/templates/eternus–temp.yaml

```
heat_template_version: 2014-10-16

description: >
    Add XML configuration file for the driver settings of each back end
```

```
parameters:
  server:
    type: string

resources:
  EternusSetup:
    type: OS::Heat::SoftwareConfig
    properties:
      group: script
      config: |
        #!/bin/bash
        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>
        </FUJITSU>
        EOF
        sudo cat > /etc/cinder/eternus <<EOF
        -----BEGIN RSA PRIVATE KEY----- #
```

```
        MIIEpAIBAAKCAQEAv5yMqonpfniu+l1PJ8qdWZpcf0d4UcHj2uyE7ou7vcZUQ1Cq
        s5Q5pjkCgYAxlTIpfOYA8jvLgc7vMEa/ZbhUgAPlYlisxbffmRsBWyJSt9gwHpcW
        hvaWo6VD/iUKZ3bOcMK0buUwBdFUt5s9B8mXbYsX6bWovlVkyu8DzQfpDiPnV6C8
        ...
        IB+46IdmCUO0DaciuEz5/KQd4AXBNdTOss2od6OzihDJXKjBwPyP1g==
        -----END RSA PRIVATE KEY-----
        EOF

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

 Private Key information in **eternus** file generated on the undercloud in Chapter 2, *Configuring the Fujitsu ETERNUS device*.

CHAPTER 4. CREATING THE FUJITSU ETERNUS ENVIRONMENT FILE

The environment file that you create to configure custom back ends contains the settings for each back end that you want to define. It also contains other settings that are relevant to the deployment of a custom back end. For more information about environment files, see Environment Files in the *Advanced Overcloud Customization* guide.

In addition, the environment file registers the heat template that you created earlier in Chapter 3, *Preparing the Fujitsu ETERNUS heat template* . The installation and echo commands defined in the heat template run on the appropriate nodes during deployment.

The following example environment file contains 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, "Creating driver definitions for each Fujitsu ETERNUS back end" , and Section 3.2, "Example Fujitsu ETERNUS heat template".

eternusbackend-env.yaml

```
resource_registry:
  OS::TripleO::NodeExtraConfig: /home/stack/templates/eternus-temp.yaml  # 1


parameter_defaults: # 2
  CinderEnableIscsiBackend: false
  CinderEnableRbdBackend: false
  CinderEnableNfsBackend: false
  NovaEnableRbdBackend: false
  GlanceBackend: file # 3
  controllerExtraConfig: # 4
    cinder::config::cinder_config:
      FJFC/volume_driver: # 5
        value: cinder.volume.drivers.fujitsu.eternus_dx.eternus_dx_fc.FJDXFCDriver
      FJFC/cinder_eternus_config_file: # 6
        value: /etc/cinder/eternus-fc.xml
      FJFC/volume_backend_name: # 7
        value: FJFC
      FJFC/fujitsu_private_key_path:
        value: /etc/cinder/eternus
      FJISCSI/volume_driver: # 8
        value: cinder.volume.drivers.fujitsu.eternus_dx.eternus_dx_iscsi.FJDXISCSIDriver
      FJISCSI/cinder_eternus_config_file:
        value: /etc/cinder/eternus-iscsi.xml
      FJISCSI/volume_backend_name:
        value: FJISCSI
      FJISCSI/fujitsu_private_key_path:
        value: /etc/cinder/eternus
    cinder_user_enabled_backends: ['FJFC','FJISCSI'] # 9
  CinderVolumeOptVolumes: 10
      - /etc/cinder/eternus-iscsi.xml:/etc/cinder/eternus-iscsi.xml:ro
      - /etc/cinder/eternus-fc.xml:/etc/cinder/eternus-fc.xml:ro
      - /etc/cinder/eternus:/etc/cinder/eternus:ro
  ContainerCinderVolumeImage: registry.connect.redhat.com/fujitsu/rhosp161-fujitsu-cinder-volume-16
  ContainerImageRegistryLogin: True
```

```
ContainerImageRegistryCredentials:
  registry.connect.redhat.com:
    my-username: my-password
  registry.redhat.io:
    my-username: my-password
```

[1] Define custom settings for all nodes before the core Puppet configuration with **NodeExtraConfig**. This ensures the following configuration when the Block Storage service deploys on the overcloud:

- The XML configuration files for each back end are present.

- The private key is generated.

[2] Set the following parameters to **false** to disable the other back end types:

- **CinderEnableIscsiBackend**: other iSCSI back ends.

- **CinderEnableRbdBackend**: Red Hat Ceph Storage.

- **CinderEnableNfsBackend**: NFS.

- **NovaEnableRbdBackend**: ephemeral Red Hat Ceph Storage.

[3] Define the Image service image storage settings with the **GlanceBackend** parameter. The following values are supported:

- **file** stores images on **/var/lib/glance/images** on each Controller node.

- **swift** uses the Object Storage service for image storage.

- **cinder** uses the Block Storage service for image storage.

[4] Define custom settings for all Controller nodes with **controllerExtraConfig**. The **cinder::config::cinder_config** class is for the Block Storage service. Director stores these back end settings in the **/etc/cinder/cinder.conf** file of each node.

[5] Configure a back end definition named **FJFC** with the **FJFC/** string, and declare the **volume_driver** parameter under that back end definition. Set the Fibre Channel ETERNUS driver for the back end with the **volume_driver** parameter, for example **cinder.volume.drivers.fujitsu.eternus_dx.eternus_dx_fc.FJDXFCDriver**.

[6] Set the path to the XML configuration file that the driver uses for the back end with **cinder_eternus_config_file**. Orchestrate the creation of **/etc/cinder/eternus-fc.xml** through the heat template, such as, **/home/stack/templates/eternus-temp.yaml**.

[7] The **volume_backend_name** is the name that the Block Storage service uses to enable the back end.

[8] Configure a new back end definition with the **FJISCSI/** string. Set the iSCSI ETERNUS driver for the back end with the **volume_driver** parameter, for example **cinder.volume.drivers.fujitsu.eternus_dx.eternus_dx_iscsi.FJDXISCSIDriver**.

[9] Set and enable custom back ends with the **cinder_user_enabled_backends** class. Use this class for user-enabled back ends only, such as those defined in the **cinder::config::cinder_config** class.

[10] Make custom configuration files on the host available to a cinder-volume service running in a container with **CinderVolumeOptVolumes**.

After creating the environment file, you can deploy your configuration. For more information about the environment file **/home/stack/templates/eternusbackend-env.yaml**, see Chapter 5, *Deploying the configured Fujitsu ETERNUS back ends*.

CHAPTER 5. DEPLOYING THE CONFIGURED FUJITSU ETERNUS BACK ENDS

After you create the eternusbackend-env.yaml file in **/home/stack/templates/**, complete the following steps:

Procedure

1. Log in as the **stack** user.

2. Deploy the back end configuration with the following command:

```
$ openstack overcloud deploy --templates \
-e [your environment files] \
-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 in the *Director Installation and Usage* guide.

Test the back end after director orchestration is complete. See Chapter 6, *Testing your Fujitsu ETERNUS configuration*.

CHAPTER 6. TESTING YOUR FUJITSU ETERNUS CONFIGURATION

After you configure the Block Storage service to use the new ETERNUS back ends, declare a **volume type** for each back end. Use volume types to specify which back end to use when you create new volumes.

- Create a Fibre Channel back end and map it to the respective back end with the following commands:

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

- Create an iSCSI back end and map it to the respective back end with the following commands:

```
# cinder type-create FJISCSI
# cinder type-key FJISCSI volume_backend_name=FJISCSI
```

For more information about volume types, see Chapter 4, *Creating the Fujitsu ETERNUS environment file*:

- Create a 1GB iSCSI volume named **test_iscsi** to verify your configuration:

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

- Test the Fibre Channel back end:

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