FUJITSU

Fujitsu ETERNUS OpenStack Volume Driver Backend Deployment Guide for Red Hat OpenStack Platform 16.2

Contents

About this document **3**

Prerequisites **3**

Configuring the Fujitsu ETERNUS device **3**

Preparing the Fujitsu ETERNUS heat template **4**

Creating driver definitions for each Fujitsu ETERNUS back end **5**

Creating the Fujitsu ETERNUS environment file **9**

Creating the Fujitsu ETERNUS container image file **11**

Deploying the configured Fujitsu ETERNUS back ends **13**

Upgrading the configured Fujitsu ETERNUS back ends **13**

Testing your Fujitsu ETERNUS configuration **14**

About this document

This document describes how to configure Red Hat OpenStack Platform 16.2 to use a Fujitsu ETERNUS Disk Storage System as a back end.

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 your 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.

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

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 “Creating driver definitions for each Fujitsu ETERNUS back end”.
9. Create a separate storage pool for volume snapshots. This pool represents the `EternusSnapPool` setting in “Creating driver definitions for each Fujitsu ETERNUS back end”. Skip this step if you set fujitsu_use_eternus_snap_pool to false.
10. Create a `Snap Data Pool Volume (SDPV)` to enable Snap Data Pool (SDP) for the `create a snapshot` function. Skip this step if you use a Thin Provision Pool for `EternusSnapPool` . Skip this step if you set fujitsu_use_eternus_snap_pool to false and you use a Thin Provision Pool for `EternusPool` .
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 -m PEM

ssh-keygen -e -f ./eternus.pub > ./eternus.iet
```

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

15. If you want to use the LUN Expand feature when attaching volume, enable the LUN Expand Mode of the host response to be enabled, and use the host response to create a host for the compute node on ETERNUS DX.

```
set host-response -host-response-name <host_response_name> -lun-expand-mode enable

create host-wwn-name -name <host_name> -wwn <wwn> -host-response-name <host_response_name>
```

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: ①
      type: OS::Heat::SoftwareConfig
      properties:
        group: script
        config: |  ②
          #!/bin/bash
          ③

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

① The `EternusSetup` section contains the resource that orchestrates the tasks on the Controller node.
② The `config` section contains the commands to run on the Controller node.
③ 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 "*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/`.

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 *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 *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_0002</EternusPool>
<EternusSnapPool>raid5_0002</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 *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.

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 mkdir -p /etc/cinder/
        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_0002</EternusPool>
        <EternusSnapPool>raid5_0002</EternusSnapPool>
        </FUJITSU>
        EOF
        sudo cat > /etc/cinder/eternus <<EOF
        -----BEGIN RSA PRIVATE KEY----- ①
        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']
```

① `Private Key` information in `eternus` file generated on the undercloud in *Configuring the Fujitsu ETERNUS device.*

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 *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 *Creating driver definitions for each Fujitsu ETERNUS back end*.

eternusbackend-env.yaml

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


parameter_defaults: ②
  CinderEnableIscsiBackend: false
  GlanceBackend: cinder 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/rhosp162-fujitsu-cinder-volume-17 ⑪
  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.

③ **Optional:** If you want to use cinder back end as the image service back end, set `cinder` to the `GlanceBackend` parameter. For more information about use cinder back end as the image service back end, see Configuring cinder back end for the Image service in the *Advanced Overcloud Customization guide*. For more information about other options of parameter `GlanceBackend`, see image-storage-glance-parameters in *Overcloud Parameters Guide*.

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`.

⑪ Define container image location with the `ContainerCinderVolumeImage` parameter. This value varies depending on the container image registry in your environment. That is, depending on if you download it from the internet or by using Red Hat OpenStack Platform (RHOSP) director or Red Hat Satellite as a local registry.

IMPORTANT:

・If you are newly deploying the overcloud, do not add `ContainerCinderVolumeImage` parameter here in eternusbackend-env.yaml. This is an old way used in Red Hat OpenStack Platform (RHOSP) 13. And you should add the `ContainerImagePrepare` parameter in a newly created container image file in *Creating the Fujitsu ETERNUS container image file*.

・If you have already deploy your overcloud by adding `ContainerCinderVolumeImage` parameter in eternusbackend-env.yaml, you do not need to redeploy your overcloud using container image file in *Creating the Fujitsu ETERNUS container image file*. No matter eternusbackend-env.yaml or the newly created container image file you add this parameter to, this will not affect your final use of ETERNUS Openstack VolumeDrive. You can skip the steps in *Creating the Fujitsu ETERNUS container image file* and simply use the original eternusbackend-env.yaml with the `ContainerCinderVolumeImage` parameter to redeploy and upgrade you overcloud.

If you want to use multipath to configure multiple I/O paths between compute nodes and ETERNUS DX, see Multipath configuration.

If you want to use any other ETERNUS Openstack VolumeDrive's parameter, see parameter description in ETERNUS Openstack VolumeDrive 1.7 User's Guide.

Creating the Fujitsu ETERNUS container image file

To use ETERNUS Openstack VolumeDrive as a Block Storage back end, you should create a container image file to specify the image which will be downloaded when deploy overcloud. Or if you have already deploy your overcloud by adding `ContainerCinderVolumeImage` parameter in eternusbackend-env.yaml, skip the following steps.

Procedure

1. Create a new container images file for your overcloud:

```
$ sudo openstack tripleo container image prepare default ¥
--local-push-destination ¥
--output-env-file containers-prepare-parameter-eternus.yaml
```

2. Add an exclude parameter to the strategy for the main Red Hat OpenStack Platform container images. Use this parameter to exclude the cinder volume container image that the fujitsu eternus container image will replace.

containers-prepare-parameter.yaml

```
parameter_defaults:
  ContainerImagePrepare:
    - push_destination: true
      excludes:
        - cinder-volume
      set:
        namespace: registry.redhat.io/rhosp-rhel8
        name_prefix: openstack-
        name_suffix: ''
        tag: 16.2
        ...
      tag_from_label: "{version}-{release}"
```

3. Add a new strategy to the ContainerImagePrepare parameter that includes the replacement container image for ETERNUS Openstack VolumeDrive:

containers-prepare-parameter-eternus.yaml

```
parameter_defaults:
  ContainerImagePrepare:
    - push_destination: true
      ...
      includes:
        - cinder-volume
      set:
        namespace: registry.connect.redhat.com/fujitsu
        name_prefix: rhosp162-fujitsu-
        name_suffix: '-17'
        tag: latest
        ...
  ContainerImageRegistryLogin: True
  ContainerImageRegistryCredentials:
    registry.redhat.io:
      my-username: my-password
    registry.connect.redhat.com:
      my-username: my-password
```

IMPORTANT:
The containers-prepare-parameter-eternus.yaml file replaces the standard containers-prepare-parameter.yaml file in your overcloud deployment. Do not include the standard containers-prepare-parameter.yaml file in your overcloud deployment. Retain the standard containers-prepare-parameter.yaml file for your undercloud installation and updates.

For more information about using third-party plugin as Block Storage backend, see Deploying a vendor plugin in the *Advanced Overcloud Customization guide*.

Deploying the configured Fujitsu ETERNUS back ends

If you are newly deploying the overcloud, after you create the *eternusbackend-env.yaml* file and *containers-prepare-parameter-eternus.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 commands:

```
openstack overcloud deploy --templates ¥
-e [your environment files] ¥
-e /home/stack/templates/eternusbackend-env.yaml
-e /home/stack/containers-prepare-parameter-eternus.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.
・If you have already deploy your overcloud by adding `ContainerCinderVolumeImage` parameter in eternusbackend-env.yaml, use the original eternusbackend-env.yaml file with the `ContainerCinderVolumeImage` parameter to redeploy your overcloud. In the case, do not use `-e /home/stack/containers-prepare-parameter-eternus.yaml` when you redeploy your overcloud.

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

Upgrading the configured Fujitsu ETERNUS back ends

If you are upgrading your RHOSP, after you create the *eternusbackend-env.yaml* file and *containers-prepare-parameter-eternus.yaml* file in /home/stack/templates/, complete the following steps:

Procedure

1. Log in as the `stack` user.

2. Run the overcloud upgrade preparation and finalization with following commands:

```
openstack overcloud upgrade prepare --templates ¥
-e [your environment files] ¥
-e /home/stack/templates/eternusbackend-env.yaml
-e /home/stack/containers-prepare-parameter-eternus.yaml
```

3. Run the overcloud upgrade finalization with following commands:

```
openstack overcloud upgrade prepare --templates ¥
-e [your environment files] ¥
-e /home/stack/templates/eternusbackend-env.yaml
-e /home/stack/containers-prepare-parameter-eternus.yaml
```

IMPORTANT:

・For more information about upgrading your RHOSP, see Frame for Upgrades(13 to 16.2).

・If you deployed your overcloud by adding `ContainerCinderVolumeImage` parameter in eternusbackend-env.yaml, use the original eternusbackend-env.yaml file with the `ContainerCinderVolumeImage` parameter to upgrade your overcloud. In the case, do not use `-e /home/stack/containers-prepare-parameter-eternus.yaml` when you upgrade your overcloud.

Testing your Fujitsu ETERNUS configuration

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

- Check whether the state of cinder volume service is up with the following commands:

```
# openstack volume service list
```

- 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 set volume_backend_name=FJISCSI
```

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

- Create a 1GB iSCSI volume named test_iscsi and check the state of the volume is available:

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

- Create a 1GB FC volume named test_fc and check the state of the volume is available:

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


FUJITSU LIMITED
http://www.fujitsu.com/storage/