Red Hat Hyperconverged Infrastructure for Virtualization 1.5

Maintaining Red Hat Hyperconverged Infrastructure for Virtualization

Common maintenance tasks for Red Hat Hyperconverged Infrastructure for Virtualization

Last Updated: 2019-02-20

Red Hat Hyperconverged Infrastructure for Virtualization 1.5 Maintaining Red Hat Hyperconverged Infrastructure for Virtualization

Common maintenance tasks for Red Hat Hyperconverged Infrastructure for Virtualization

Laura Bailey
lbailey@redhat.com

Legal Notice

Copyright © 2019 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

Red Hat Hyperconverged Infrastructure for Virtualization (RHHI for Virtualization) combines compute, storage, networking, and management capabilities into a single solution, simplifying deployment and reducing the cost of acquisition and maintenance. This document explains how to perform maintenance tasks specific to Red Hat Hyperconverged Infrastructure for Virtualization.

Table of Contents

PART I. CONFIGURATION TASKS

CHAPTER 1. ADD COMPUTE AND STORAGE RESOURCES

Red Hat Hyperconverged Infrastructure for Virtualization (RHHI for Virtualization) can be scaled in multiples of three nodes to a maximum of twelve nodes.

1.1. EXPANDING THE HYPERCONVERGED CLUSTER BY ADDING A NEW VOLUME ON NEW NODES USING COCKPIT

Follow these instructions to use the Cockpit UI to expand your hyperconverged cluster.

Prerequisites

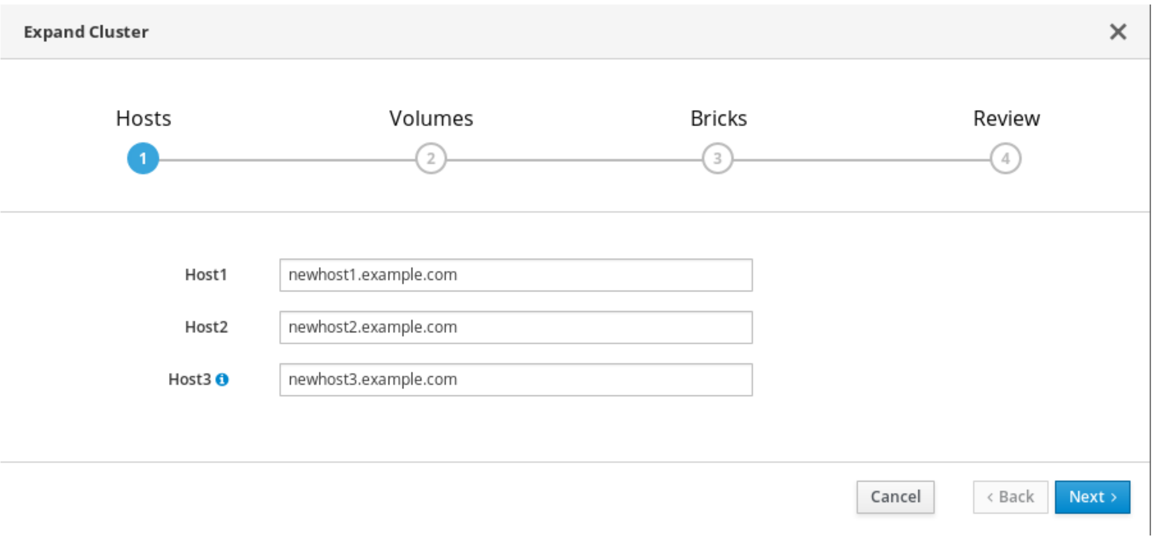
- Verify that your scaling plans are supported.
- If your existing deployment uses certificates signed by a Certificate Authority for encryption, prepare the certificates that will be required for the new nodes.
- Install three physical machines to serve as the new hyperconverged nodes.
  Follow the instructions in Deploying Red Hat Hyperconverged Infrastructure for Virtualization.
- Configure key-based SSH authentication without a password.
  Configure this from the node that is running Cockpit to all new nodes, and from the first new node to all other new nodes.

  > **IMPORTANT**
  >
  > RHHI for Virtualization expects key-based SSH authentication without a password between these nodes for both IP addresses and FQDNs. Ensure that you configure key-based SSH authentication between these machines for the IP address and FQDN of all storage and management network interfaces.

  Follow the instructions in Using key-based authentication to configure key-based SSH authentication without a password.

Procedure

1. Log in to Cockpit.
2. Click **Virtualization** → **Hosted Engine** and then click **Manage Gluster**.
3. Click **Expand Cluster**. The *Gluster Deployment* window opens.
   a. On the *Hosts* tab, enter the FQDN or IP address of the new hyperconverged nodes and click **Next**.

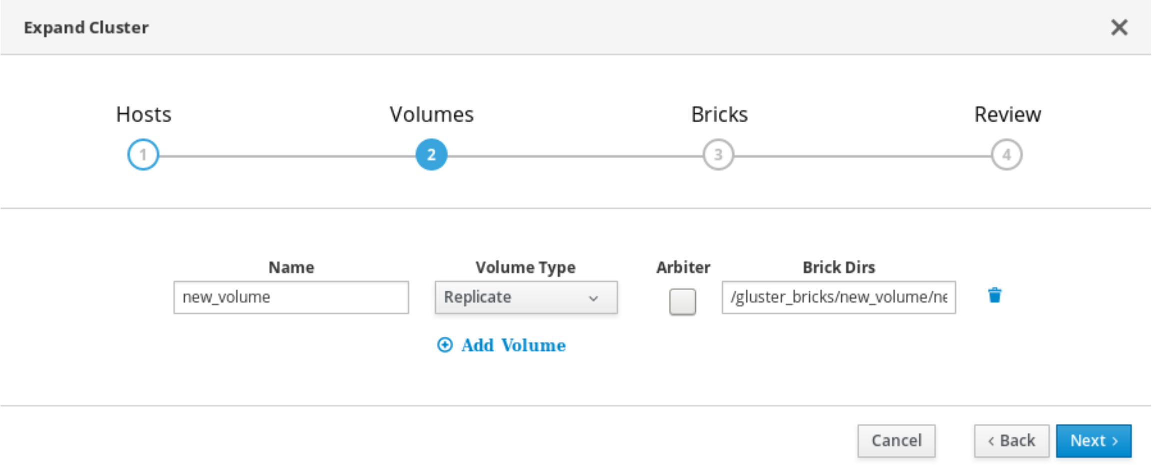


Host1 newhost1.example.com

Host2 newhost2.example.com

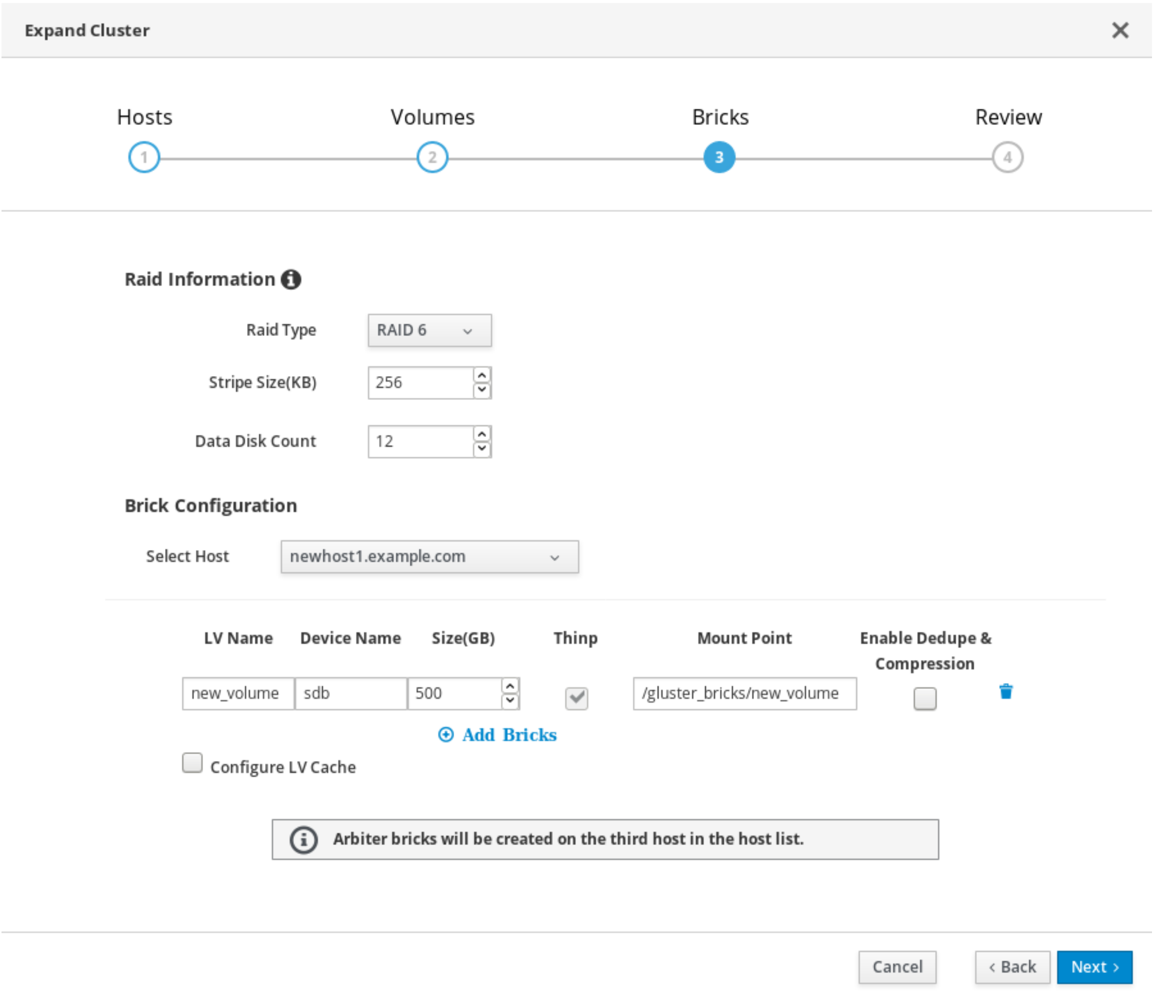
Host3 newhost3.example.com

b. On the *Volumes* tab, specify the details of the volume you want to create.



Name	Volume Type	Arbiter	Brick Dirs
new_volume	Replicate	□	/gluster_bricks/new_volume/ne

c. On the *Bricks* tab, specify the details of the disks to be used to create the Gluster volume.



Raid Information

Raid Type	RAID 6
Stripe Size(KB)	256
Data Disk Count	12

Brick Configuration

Select Host: newhost1.example.com

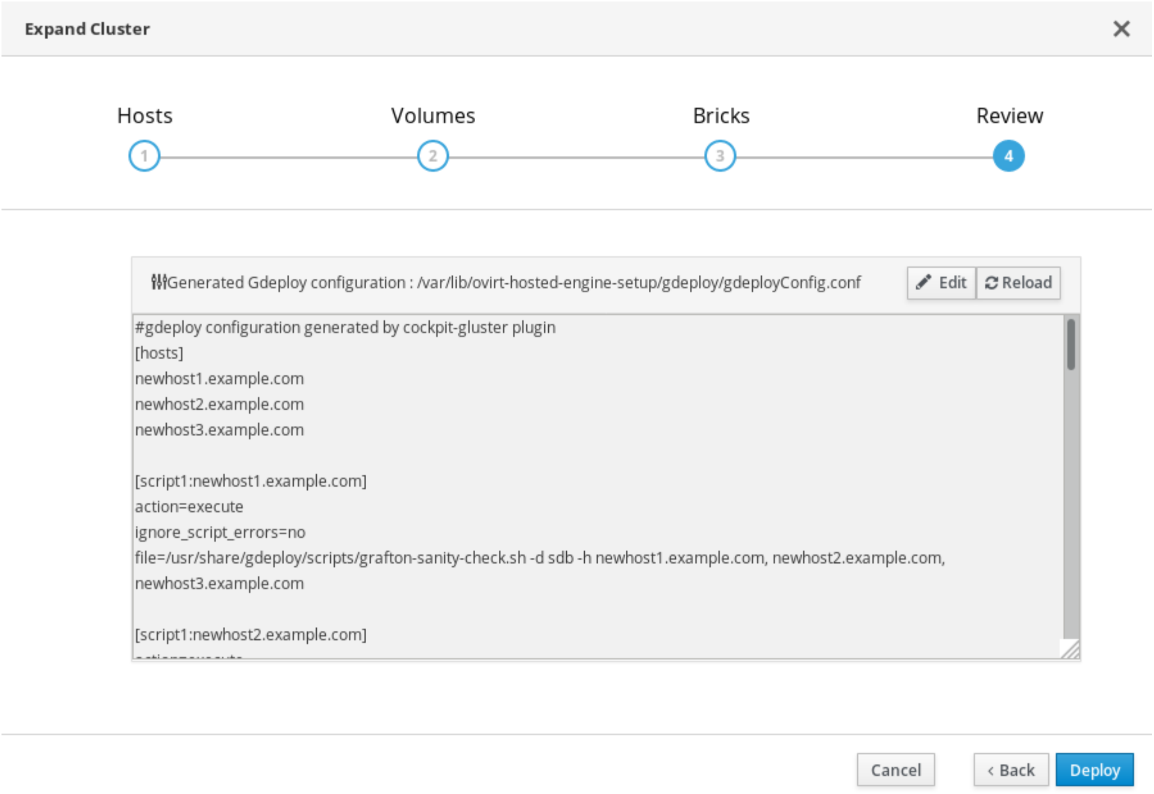
LV Name	Device Name	Size(GB)	Thinp	Mount Point	Enable Dedupe & Compression
new_volume	sdb	500	☑	/gluster_bricks/new_volume	□

⊕ Add Bricks

□ Configure LV Cache

Arbiter bricks will be created on the third host in the host list.

d. On the *Review* tab, check the generated gdeploy file for any problems. When you are satisfied, click **Deploy**.



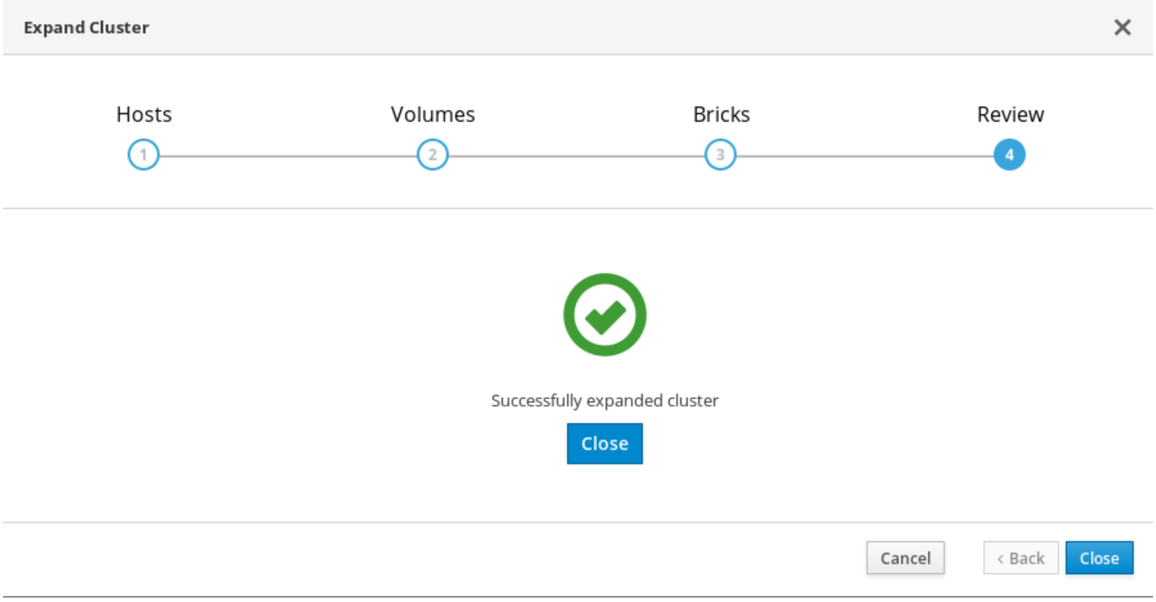
Generated Gdeploy configuration : /var/lib/ovirt-hosted-engine-setup/gdeploy/gdeployConfig.conf

```
#gdeploy configuration generated by cockpit-gluster plugin
[hosts]
newhost1.example.com
newhost2.example.com
newhost3.example.com

[script1:newhost1.example.com]
action=execute
ignore_script_errors=no
file=/usr/share/gdeploy/scripts/grafton-sanity-check.sh -d sdb -h newhost1.example.com, newhost2.example.com,
newhost3.example.com

[script1:newhost2.example.com]
```

Deployment takes some time to complete. The following screen appears when the cluster has been successfully expanded.



Successfully expanded cluster

CHAPTER 2. CONFIGURE HIGH AVAILABILITY USING FENCING POLICIES

Fencing allows a cluster to enforce performance and availability policies and react to unexpected host failures by automatically rebooting virtualization hosts.

Several policies specific to Red Hat Gluster Storage must be enabled to ensure that fencing activities do not disrupt storage services in a Red Hat Hyperconverged (RHHI for Virtualization) Infrastructure deployment.

This requires enabling and configuring fencing at both the cluster level and at the host level. See the following sections for details.

2.1. CONFIGURING FENCING POLICIES IN THE CLUSTER

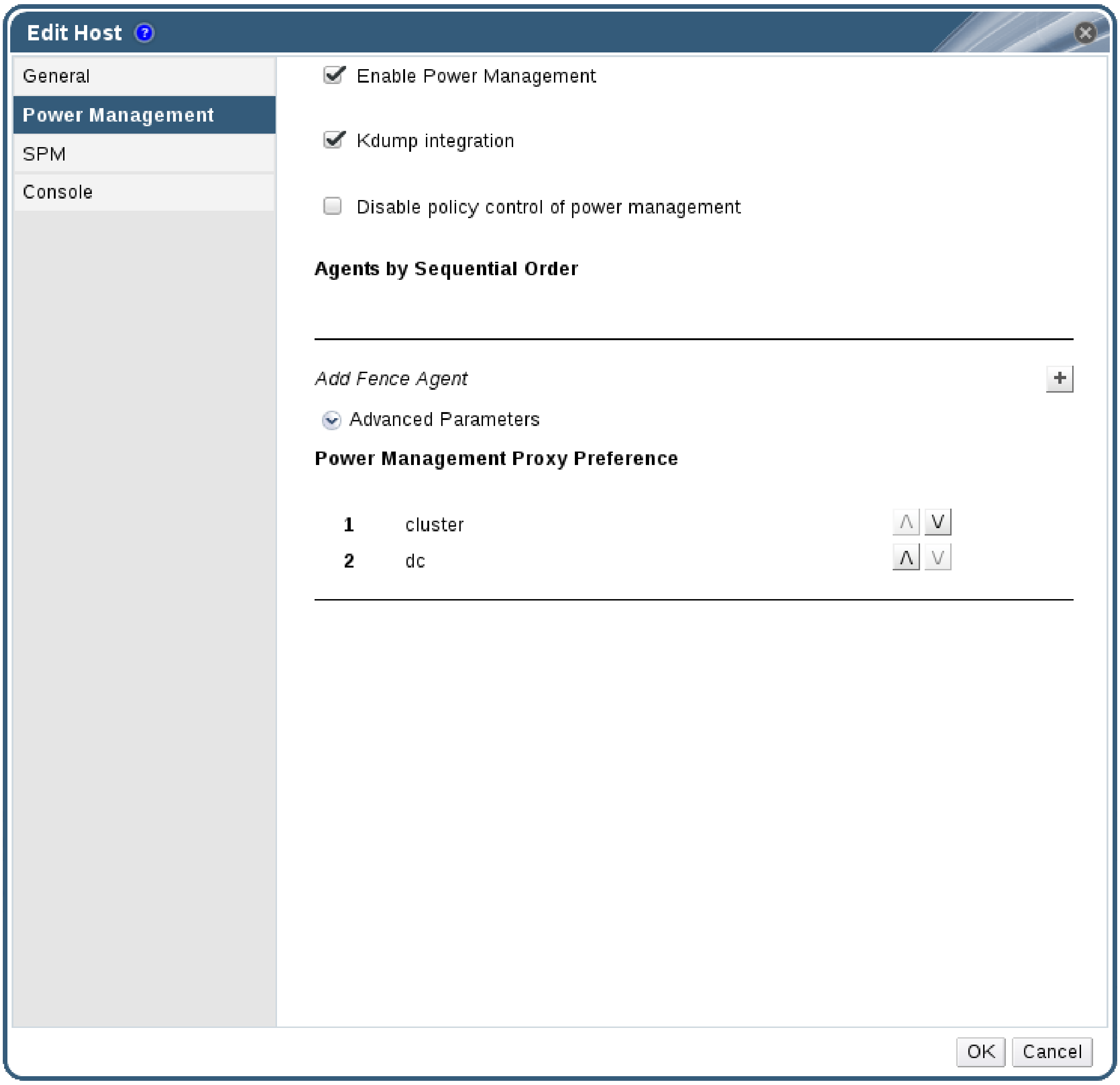
1. In Red Hat Virtualization Manager, click **Compute** → **Clusters**.
2. Select the cluster and click **Edit**. The *Edit Cluster* window opens.
3. Click the **Fencing policy** tab.
4. Check the **Enable fencing** checkbox.
5. Check the checkboxes for at least the following fencing policies:
   - Skip fencing if gluster bricks are up
   - Skip fencing if gluster quorum not met

   See Appendix A, *Fencing Policies for Red Hat Gluster Storage* for details on the effects of these policies.
6. Click **OK** to save settings.

2.2. CONFIGURING FENCING PARAMETERS ON THE HOSTS

1. In Red Hat Virtualization Manager, click **Compute** → **Hosts**.
2. Select the host to configure, and click **Edit** to open the **Edit Host** window.
3. Click the **Power Management** tab.

Figure 2.1. Power Management Settings



☑ Enable Power Management

☑ Kdump integration

□ Disable policy control of power management

Agents by Sequential Order

Add Fence Agent +

Advanced Parameters

Power Management Proxy Preference

1	cluster
2	dc

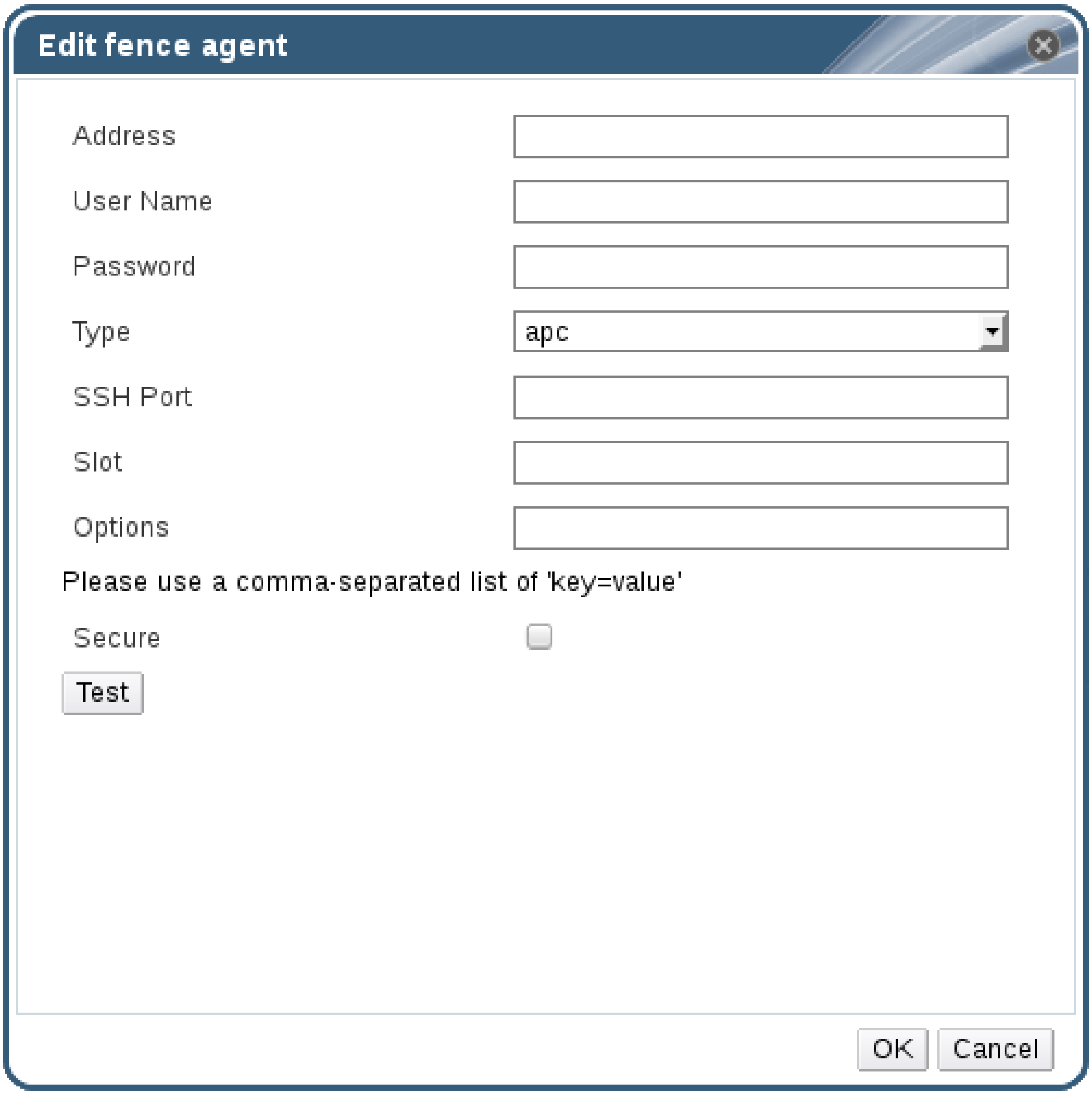
4. Check the **Enable Power Management** check box. This enables other fields on the tab.

5. Check the **Kdump integration** check box to prevent the host from fencing while performing a kernel crash dump.

> **IMPORTANT**
>
> When you enable Kdump integration on an existing host, the host must be reinstalled for kdump to be configured. See Chapter 11, *Reinstalling a virtualization host* for instructions on reinstalling a host.

1. Click the plus (+) button to add a new power management device. The **Edit fence agent** window opens.

Figure 2.2. Edit fence agent



Edit fence agent

Address

User Name

Password

Type: apc

SSH Port

Slot

Options

Please use a comma-separated list of 'key=value'

Secure □

   a. Enter the **Address**, **User Name**, and **Password** of the power management device.

   b. Select the power management device **Type** from the drop-down list.

a. Enter the **SSH Port** number used by the power management device to communicate with the host.

b. Enter the **Slot** number used to identify the blade of the power management device.

c. Enter the **Options** for the power management device. Use a comma-separated list of *key=value* entries.

d. Check the **Secure** check box to enable the power management device to connect securely to the host.

e. Click the **Test** button to ensure the settings are correct. *Test Succeeded, Host Status is: on* displays upon successful verification.

> **WARNING**
>
> Power management parameters (userid, password, options, etc.) are tested by Red Hat Virtualization Manager in two situations: during setup, and when parameter values are manually changed in Red Hat Virtualization Manager. If you choose to ignore alerts about incorrect parameters, or if the parameters are changed on the power management hardware without the corresponding change in Red Hat Virtualization Manager, fencing is likely to fail.

f. Click **OK** to finish adding the fence agent.

1. Click **OK** to save your host configuration.

You are returned to the list of hosts. Note that the exclamation mark (`!`) next to the host’s name has now disappeared, signifying that power management has been successfully configured.

CHAPTER 3. CONFIGURING BACKUP AND RECOVERY OPTIONS

This chapter explains how to add disaster recovery capabilities to your Red Hat Hyperconverged Infrastructure for Virtualization deployment so that you can restore your cluster to a working state after a disk or server failure.

3.1. PREREQUISITES

3.1.1. Prerequisites for geo-replication

Be aware of the following requirements and limitations when configuring geo-replication:

One geo-replicated volume only

Red Hat Hyperconverged Infrastructure for Virtualization (RHHI for Virtualization) supports only one geo-replicated volume. Red Hat recommends backing up the volume that stores the data of your virtual machines, as this is usually contains the most valuable data.

Two different managers required

The source and destination volumes for geo-replication must be managed by different instances of Red Hat Virtualization Manager.

3.1.2. Prerequisites for failover and failback configuration

Versions must match between environments

Ensure that the primary and secondary environments have the same version of Red Hat Virtualization Manager, with identical data center compatibility versions, cluster compatibility versions, and PostgreSQL versions.

No virtual machine disks in the hosted engine storage domain

The storage domain used by the hosted engine virtual machine is not failed over, so any virtual machine disks in this storage domain will be lost.

Execute Ansible playbooks manually from a separate master node

Generate and execute Ansible playbooks manually from a separate machine that acts as an Ansible master node.

3.2. SUPPORTED BACKUP AND RECOVERY CONFIGURATIONS

There are two supported ways to add disaster recovery capabilities to your Red Hat Hyperconverged Infrastructure for Virtualization deployment.

Configure backing up to a secondary volume only

Regularly synchronizing your data to a remote secondary volume helps to ensure that your data is not lost in the event of disk or server failure.
This option is suitable if the following statements are true of your deployment.

- You require only a backup of your data for disaster recovery.
- You do not require highly available storage.
- You do not want to maintain a secondary cluster.

- You are willing to manually restore your data and reconfigure your backup solution after a failure has occurred.

Follow the instructions in Configuring backup to a secondary volume to configure this option.

Configure failing over to and failing back from a secondary cluster

This option provides failover and failback capabilities in addition to backing up data on a remote volume. Configuring failover of your primary cluster's operations and storage domains to a secondary cluster helps to ensure that your data remains available in event of disk or server failure in the primary cluster.
This option is suitable if the following statements are true of your deployment.

- You require highly available storage.
- You are willing to maintain a secondary cluster.
- You do not want to manually restore your data or reconfigure your backup solution after a failure has occurred.

Follow the instructions in Configuring failover to and failback from a secondary cluster to configure this option.

Red Hat recommends that you configure at least a backup volume for production deployments.

3.3. CONFIGURING BACKUP TO A SECONDARY VOLUME

This section covers how to back up a gluster volume to a secondary gluster volume using geo-replication.

To do this, you must:

1. Ensure that all prerequisites are met.
2. Create a suitable volume to use as a geo-replication target.
3. Configure a geo-replication session between the source volume and the target volume.
4. Schedule the geo-replication process.

3.3.1. Prerequisites

3.3.1.1. Enable shared storage on the source volume

Ensure that the volume you want to back up (the source volume) has shared storage enabled. Run the following command on any server that hosts the source volume to enable shared storage.

```
# gluster volume set all cluster.enable-shared-storage enable
```

Ensure that a gluster volume named **`gluster_shared_storage`** is created in the source cluster, and is mounted at **`/var/run/gluster/shared_storage`** on all the nodes in the source cluster. See Setting Up Shared Storage for further information.

3.3.1.2. Match encryption on source and target volumes

If encryption is enabled on the volume that you want to back up, encryption must also be enabled on the volume that will hold your backed up data.

See Configure Encryption with Transport Layer Security (TLS/SSL) for details.

3.3.2. Create a suitable target volume for geo-replication

Prepare a secondary gluster volume to hold the geo-replicated copy of your source volume. This target volume should be in a separate cluster, hosted at a separate site, so that the risk of source and target volumes being affected by the same outages is minimised.

Ensure that the target volume for geo-replication has sharding enabled. Run the following command on any node that hosts the target volume to enable sharding on that volume.

```
# gluster volume set <volname> features.shard enable
```

3.3.3. Configuring geo-replication for backing up volumes

3.3.3.1. Creating a geo-replication session

A geo-replication session is required to replicate data from an active source volume to a passive target volume.

IMPORTANT

Only rsync based geo-replication is supported with Red Hat Hyperconverged Infrastructure for Virtualization.

1. Create a common **pem pub** file.
   Run the following command on a source node that has key-based SSH authentication without a password configured to the target nodes.

   ```
   # gluster system:: execute gsec_create
   ```

2. Create the geo-replication session
   Run the following command to create a geo-replication session between the source and target volumes, using the created **pem pub** file for authentication.

   ```
   # gluster volume geo-replication <SOURCE_VOL> <TARGET_NODE>::
   <TARGET_VOL> create push-pem
   ```

   For example, the following command creates a geo-replication session from a source volume **prodvol** to a target volume called **backupvol**, which is hosted by **backup.example.com**.

   ```
   # gluster volume geo-replication prodvol
   backup.example.com::backupvol create push-pem
   ```

   By default this command verifies that the target volume is a valid target with available space. You can append the **force** option to the command to ignore failed verification.

3. Configure a meta-volume

This relies on the source volume having shared storage configured, as described in Prerequisites.

```
# gluster volume geo-replication <SOURCE_VOL> <TARGET_HOST>::
<TARGET_VOL> config use_meta_volume true
```

IMPORTANT

Do not start the geo-replication session. Starting the geo-replication session begins replication from your source volume to your target volume.

3.3.3.2. Verifying creation of a geo-replication session

1. Log in to Red Hat Virtualization Manager on any source node.
2. Click **Storage** → **Volumes**.
3. Check the **Info** column for the geo-replication icon.
   If this icon is present, geo-replication has been configured for that volume.

   If this icon is not present, try synchronizing the volume.

3.3.3.3. Synchronizing volume state using the Administration Portal

1. Log in to Red Hat Virtualization Manager.
2. Click **Compute** → **Volumes**.
3. Select the volume that you want to synchronize.
4. Click the **Geo-replication** sub-tab.
5. Click **Sync**.

3.3.4. Scheduling regular backups using geo-replication

1. Log in to Red Hat Virtualization Manager on any source node.
2. Click **Storage** → **Domains**.
3. Click the name of the storage domain that you want to back up.
4. Click the **Remote Data Sync Setup** subtab.
5. Click **Setup**.
   The *Setup Remote Data Synchronization* window opens.
   a. In the **Geo-replicated to** field, select the backup target.
   b. In the **Recurrence** field, select a recurrence interval type.
      Valid values are **WEEKLY** with at least one weekday checkbox selected, or **DAILY**.
   c. In the **Hours** and **Minutes** field, specify the time to start synchronizing.

NOTE

This time is based on the Hosted Engine's timezone.

d. Click **OK**.

6. Check the **Events** subtab for the source volume at the time you specified to verify that synchronization works correctly.

3.4. CONFIGURING FAILOVER TO AND FAILBACK FROM A SECONDARY CLUSTER

This section covers how to configure your cluster to fail over to a remote secondary cluster in the event of server failure.

To do this, you must:

1. Configure backing up to a remote volume.
2. Create a suitable cluster to use as a failover target.
3. Prepare a mapping file for the source and target clusters.
4. Prepare a failover playbook.
5. Prepare a cleanup playbook for the primary cluster.
6. Prepare a failback playbook.

3.4.1. Creating a secondary cluster for failover

Install and configure a secondary cluster that can be used in place of the primary cluster in the event of failure.

This secondary cluster can be either of the following configurations:

Red Hat Hyperconverged Infrastructure

See Deploying Red Hat Hyperconverged Infrastructure for details.

Red Hat Gluster Storage configured for use as a Red Hat Virtualization storage domain

See Configuring Red Hat Virtualization with Red Hat Gluster Storage for details. Note that creating a storage domain is not necessary for this use case; the storage domain is imported as part of the failover process.

The storage on the secondary cluster must not be attached to a data center, so that it can be added to the secondary site's data center during the failover process.

3.4.2. Creating a mapping file between source and target clusters

Follow this section to create a file that maps the storage in your source cluster to the storage in your target cluster.

Red Hat recommends that you create this file immediately after you first deploy your storage, and keep it up to date as your deployment changes. This helps to ensure that everything in your cluster fails over safely in the event of disaster.

1. Create a playbook to generate the mapping file.
   Create a playbook that passes information about your cluster to the **oVirt.disaster-recovery** role, using the **site**, **username**, **password**, and **ca** variables.

   Red Hat recommends creating this file in the **/usr/share/ansible/roles/oVirt.disaster-recovery** directory of the server that provides **ansible** and manages failover and failback.

   Example playbook file: dr-ovirt-setup.yml

   ```
   ---
   - name: Collect mapping variables
     hosts: localhost
     connection: local

     vars:
       site: https://example.engine.redhat.com/ovirt-engine/api
       username: admin@internal
       password: my_password
       ca: /etc/pki/ovirt-engine/ca.pem
       var_file: disaster_recovery_vars.yml

     roles:
       - oVirt.disaster-recovery
   ```

2. Generate the mapping file by running the playbook with the **generate_mapping** tag.

   ```
   # ansible-playbook dr-ovirt-setup.yml --tags "generate_mapping"
   ```

   This creates the mapping file, **disaster_recovery_vars.yml**.

3. Edit **disaster_recovery_vars.yml** and add information about the secondary cluster.
   See Appendix A: Mapping File Attributes in the Red Hat Virtualization *Disaster Recovery Guide* for detailed information about attributes used in the mapping file.

3.4.3. Creating a failover playbook between source and target clusters

Create a playbook file that passes the lists of virtualization hosts to use as a failover source and target to the **oVirt.disaster-recovery** role, using the **dr_target_host** and **dr_source_map** variables.

Red Hat recommends creating this file in the **/usr/share/ansible/roles/oVirt.disaster-recovery** directory of the server that provides **ansible** and manages failover and failback.

Example playbook file: dr-rhv-failover.yml

```
---
- name: Failover RHV
  hosts: localhost
  connection: local
  vars:
    dr_target_host: secondary
    dr_source_map: primary
  vars_files:
    - disaster_recovery_vars.yml
```

```
    - passwords.yml
  roles:
    - oVirt.disaster-recovery
```

For information about executing failover, see Failing over to a secondary cluster.

3.4.4. Creating a failover cleanup playbook for your primary cluster

Create a playbook file that cleans up your primary cluster so that you can use it as a failback target.

Red Hat recommends creating this file in the **`/usr/share/ansible/roles/oVirt.disaster-recovery`** directory of the server that provides **`ansible`** and manages failover and failback.

Example playbook file: dr-cleanup.yml

```
---
- name: Clean RHV
  hosts: localhost
  connection: local
  vars:
    dr_source_map: primary
  vars_files:
    - disaster_recovery_vars.yml
  roles:
    - oVirt.disaster-recovery
```

For information about executing failback, see Failing back to a primary cluster.

3.4.5. Create a failback playbook between source and target clusters

Create a playbook file that passes the lists of virtualization hosts to use as a failback source and target to the **`oVirt.disaster-recovery`** role, using the **`dr_target_host`** and **`dr_source_map`** variables.

Red Hat recommends creating this file in the **`/usr/share/ansible/roles/oVirt.disaster-recovery`** directory of the server that provides **`ansible`** and manages failover and failback.

Example playbook file: dr-rhv-failback.yml

```
---
- name: Failback RHV
  hosts: localhost
  connection: local
  vars:
    dr_target_host: primary
    dr_source_map: secondary
  vars_files:
    - disaster_recovery_vars.yml
    - passwords.yml
  roles:
    - oVirt.disaster-recovery
```

For information about executing failback, see Failing back to a primary cluster.

CHAPTER 4. CONFIGURE ENCRYPTION WITH TRANSPORT LAYER SECURITY (TLS/SSL)

Transport Layer Security (TLS/SSL) can be used to encrypt management and storage layer communications between nodes. This helps ensure that your data remains private.

Encryption can be configured using either self-signed certificates or certificates signed by a Certificate Authority.

This document assumes that you want to enable encryption on an existing deployment. However, encryption can also be configured as part of the deployment process. See *Deploying Red Hat Hyperconverged Infrastructure for Virtualization* for details: https://access.redhat.com/documentation/en-us/red_hat_hyperconverged_infrastructure_for_virtualization/1.5/html/deploying_red_hat_hyperconverged_

4.1. CONFIGURING TLS/SSL USING SELF-SIGNED CERTIFICATES

IMPORTANT

Enabling or disabling encryption is a disruptive process that requires virtual machines and the Hosted Engine to be shut down.

1. Shut down all virtual machines
   See *Shutting Down a Virtual Machine* in the Red Hat Virtualization documentation for details: https://access.redhat.com/documentation/en-us/red_hat_virtualization/4.2/html/virtual_machine_management_guide/chap-administrative_tasks.

2. Move all storage domains **except the hosted engine storage domain** into Maintenance mode
   See *Moving Storage Domains to Maintenance Mode*in the Red Hat Virtualization documentation for details: https://access.redhat.com/documentation/en-us/red_hat_virtualization/4.2/html/administration_guide/sect-storage_tasks.

3. Move the hosted engine into global maintenance mode
   Run the following command on the virtualization host that hosts the hosted engine:

   ```
   # hosted-engine --set-maintenance --mode=global
   ```

4. Shut down the hosted engine virtual machine
   Run the following command on the virtualization host that hosts the hosted engine:

   ```
   # hosted-engine --vm-shutdown
   ```

   Verify that the hosted engine has shut down by running the following command:

   ```
    # hosted-engine --vm-status
   ```

5. Stop all high availability services
   Run the following command on all virtualization hosts:

   ```
   # systemctl stop ovirt-ha-agent
   # systemctl stop ovirt-ha-broker
   ```

6. Unmount the hosted engine storage domain from all virtualization hosts

   ```
   # hosted-engine --disconnect-storage
   ```

7. Verify that all volumes are unmounted
   On each virtualization host, verify that all gluster volumes are no longer mounted.

   ```
   # mount
   ```

8. Create a gdeploy configuration file
   Use the template file in Section B.1, “Example gdeploy configuration file for setting up TLS/SSL” to create a new configuration file that will set up TLS/SSL on your deployment.

9. Run gdeploy using your new configuration file
   On the first physical machine, run gdeploy using the configuration file you created in the previous step:

   ```
   # gdeploy -c set_up_encryption.conf
   ```

   This may take some time to complete.

10. Verify that no TLS/SSL errors occurred
    Check the **/var/log/glusterfs/glusterd.log** file on each physical machine to ensure that no TLS/SSL related errors occurred, and setup completed successfully.

11. Start all high availability services
    Run the following commands on all virtualization hosts:

    ```
    # systemctl start ovirt-ha-agent
    # systemctl start ovirt-ha-broker
    ```

12. Move the hosted engine out of Global Maintenance mode

    ```
    # hosted-engine --set-maintenance --mode=none
    ```

    The hosted engine starts automatically after a short wait.

13. Wait for nodes to synchronize
    Run the following command on the first virtualization host to check synchronization status. If engine status is listed as **unknown stale-data**, synchronization requires several more minutes to complete.

    The following output indicates completed synchronization.

    ```
    # hosted-engine --vm-status | grep 'Engine status'
    Engine status   : {"health": "good", "vm": "up", "detail": "up"}
    Engine status   : {"reason": "vm not running on this host",
      "health": "bad", "vm": "down", "detail": "unknown"}
    Engine status   : {"reason": "vm not running on this host",
      "health": "bad", "vm": "down", "detail": "unknown"}
    ```

14. Activate all storage domains
    Activate the master storage domain first, followed by all other storage domains.

For details on activating storage domains, see *Activating Storage Domains from Maintenance Mode* in the Red Hat Virtualization documentation: https://access.redhat.com/documentation/en-us/red_hat_virtualization/4.2/html/administration_guide/sect-storage_tasks.

15. Start all virtual machines
    See *Starting a Virtual Machine* in the Red Hat Virtualization documentation for details: https://access.redhat.com/documentation/en-us/red_hat_virtualization/4.2/html/virtual_machine_management_guide/sect-starting_the_virtual_machine.

4.2. CONFIGURING TLS/SSL USING CERTIFICATE AUTHORITY SIGNED CERTIFICATES

IMPORTANT

Enabling or disabling encryption is a disruptive process that requires virtual machines and the Hosted Engine to be shut down.

IMPORTANT

Ensure that you have appropriate certificates signed by a Certificate Authority before proceeding. Obtaining certificates is outside the scope of this document, but further details are available in the Red Hat Gluster Storage *Administration Guide*: https://access.redhat.com/documentation/en-us/red_hat_gluster_storage/3.4/html/administration_guide/chap-network_encryption#chap-Network_Encryption-Prereqs.

1. Shut down all virtual machines
   See *Shutting Down a Virtual Machine* in the Red Hat Virtualization documentation for details: https://access.redhat.com/documentation/en-us/red_hat_virtualization/4.2/html/virtual_machine_management_guide/chap-administrative_tasks.

2. Move all storage domains **except the hosted engine storage domain** into Maintenance mode
   See *Moving Storage Domains to Maintenance Mode*in the Red Hat Virtualization documentation for details: https://access.redhat.com/documentation/en-us/red_hat_virtualization/4.2/html/administration_guide/sect-storage_tasks.

3. Move the hosted engine into global maintenance mode
   Run the following command on the virtualization host that hosts the hosted engine:

   ```
   # hosted-engine --set-maintenance --mode=global
   ```

4. Shut down the hosted engine virtual machine
   Run the following command on the virtualization host that hosts the hosted engine:

   ```
   # hosted-engine --vm-shutdown
   ```

   Verify that the hosted engine has shut down by running the following command:

   ```
    # hosted-engine --vm-status
   ```

5. Stop all high availability services
   Run the following command on all virtualization hosts:

   ```
   # systemctl stop ovirt-ha-agent
   # systemctl stop ovirt-ha-broker
   ```

6. Unmount the hosted engine storage domain from all virtualization hosts

   ```
   # hosted-engine --disconnect-storage
   ```

7. Verify that all volumes are unmounted
   On each virtualization host, verify that all gluster volumes are no longer mounted.

   ```
   # mount
   ```

8. Configure Certificate Authority signed encryption

   IMPORTANT

   Ensure that you have appropriate certificates signed by a Certificate Authority before proceeding. Obtaining certificates is outside the scope of this document.

   a. Place certificates in the following locations on all nodes.

      /etc/ssl/glusterfs.key
      The node’s private key.

      /etc/ssl/glusterfs.pem
      The certificate signed by the Certificate Authority, which becomes the node’s certificate.

      /etc/ssl/glusterfs.ca
      The Certificate Authority’s certificate.

   b. Stop all volumes

      ```
      # gluster volume stop all
      ```

   c. Restart glusterd on all nodes

      ```
      # systemctl restart glusterd
      ```

   d. Enable TLS/SSL encryption on all volumes

      ```
      # gluster volume set <volname> client.ssl on
      # gluster volume set <volname> server.ssl on
      ```

   e. Specify access permissions on all hosts

      ```
      # gluster volume set <volname> auth.ssl-allow "host1,host2,host3"
      ```

   f. Start all volumes

```
# gluster volume start all
```

9. Verify that no TLS/SSL errors occurred
   Check the **/var/log/glusterfs/glusterd.log** file on each physical machine to ensure that no TLS/SSL related errors occurred, and setup completed successfully.

10. Start all high availability services
    Run the following commands on all virtualization hosts:

    ```
    # systemctl start ovirt-ha-agent
    # systemctl start ovirt-ha-broker
    ```

11. Move the hosted engine out of Global Maintenance mode

    ```
    # hosted-engine --set-maintenance --mode=none
    ```

    The hosted engine starts automatically after a short wait.

12. Wait for nodes to synchronize
    Run the following command on the first virtualization host to check synchronization status. If engine status is listed as **unknown stale-data**, synchronization requires several more minutes to complete.

    The following output indicates completed synchronization.

    ```
    # hosted-engine --vm-status | grep 'Engine status'
    Engine status   : {"health": "good", "vm": "up", "detail": "up"}
    Engine status   : {"reason": "vm not running on this host",
      "health": "bad", "vm": "down", "detail": "unknown"}
    Engine status   : {"reason": "vm not running on this host",
      "health": "bad", "vm": "down", "detail": "unknown"}
    ```

13. Activate all storage domains
    Activate the master storage domain first, followed by all other storage domains.

    For details on activating storage domains, see *Activating Storage Domains from Maintenance Mode* in the Red Hat Virtualization documentation: https://access.redhat.com/documentation/en-us/red_hat_virtualization/4.2/html/administration_guide/sect-storage_tasks.

14. Start all virtual machines
    See *Starting a Virtual Machine* in the Red Hat Virtualization documentation for details: https://access.redhat.com/documentation/en-us/red_hat_virtualization/4.2/html/virtual_machine_management_guide/sect-starting_the_virtual_machine.

CHAPTER 5. CONFIGURE PERFORMANCE IMPROVEMENTS

Some deployments benefit from additional configuration to achieve optimal performance. This section covers recommended additional configuration for certain deployments.

5.1. IMPROVING VOLUME PERFORMANCE BY CHANGING SHARD SIZE

The default value of the **`shard-block-size`** parameter changed from **`4MB`** to **`64MB`** between Red Hat Hyperconverged Infrastructure for Virtualization version 1.0 and 1.1. This means that all new volumes are created with a **`shard-block-size`** value of 64MB. However, existing volumes retain the original **`shard-block-size`** value of 4MB.

There is no safe way to modify the **`shard-block-size`** value on volumes that contain data. Because shard block size applies only to writes that occur after the value is set, attempting to change the value on a volume that contains data results in a mixed shard block size, which results in poor performance.

This section shows you how to safely modify the shard block size on an existing volume after upgrading to Red Hat Hyperconverged Infrastructure for Virtualization 1.1 or higher, in order to take advantage of the performance benefits of a larger shard size.

5.1.1. Prerequisites

- A logical thin pool with sufficient free space to create additional logical volumes that are large enough to contain all existing virtual machines.

- A complete backup of your data. For details on how to achieve this, see Configuring backup and recovery options.

5.1.2. Safely changing the shard block size parameter value

A. Create a new storage domain

1. **Create new thin provisioned logical volumes**

    a. For an arbitrated replicated volume:

        i. Create an **`lv_create_arbitrated.conf`** file with the following contents:

```
[lv10:{<Gluster_Server_IP1>,<Gluster_Server_IP2>}]
action=create
lvname=<lv_name>
ignore_lv_errors=no
vgname=<volgroup_name>
mount=<brick_mountpoint>
lvtype=thinlv
poolname=<thinpool_name>
virtualsize=<size>

[lv11:<Gluster_Server_IP3>]
action=create
lvname=<lv_name>
ignore_lv_errors=no
vgname=<volgroup_name>
```

```
mount=<brick_mountpoint>
lvtype=thinlv
poolname=<thinpool_name>
virtualsize=<size>
```

ii. Run the following command:

```
# gdeploy -c lv_create_arbitrated.conf
```

b. For a normal replicated volume:

i. Create an **`lv_create_replicated.conf`** file with the following contents:

```
[lv3]
action=create
lvname=<lv_name>
ignore_lv_errors=no
vgname=<volgroup_name>
mount=<brick_mountpoint>
lvtype=thinlv
poolname=<thinpool_name>
virtualsize=<size>
```

ii. Run the following command:

```
# gdeploy -c lv_create_replicated.conf
```

2. **Create new gluster volumes on the new logical volumes**

a. For an arbitrated replicated volume

i. Create a **`gluster_arb_volume.conf`** file with the following contents:

```
[volume4]
action=create
volname=data_one
transport=tcp
replica=yes
replica_count=3
key=group,storage.owner-uid,storage.owner-gid,network.ping-
timeout,performance.strict-o-direct,network.remote-
dio,cluster.granular-entry-heal,features.shard-block-
size,server.ssl,client.ssl,auth.ssl-allow
value=virt,36,36,30,on,off,enable,64MB,on,on,"
<Gluster_Server_IP1>;<Gluster_Server_IP2>;
<Gluster_Server_IP3>"
brick_dirs=<Gluster_Server_IP1>:<brick1_mountpoint>,
<Gluster_Server_IP2>:<brick2_mountpoint>,<Gluster_Server_IP3>:
<brick3_mountpoint>
ignore_volume_errors=no
arbiter_count=1
```

ii. Run the following command:

```
# gdeploy -c gluster_arb_volume.conf
```

b. For a normal replicated volume:

   i. Create a gluster_rep_volume.conf file with the following contents:

```
[volume2]
action=create
volname=data
transport=tcp
replica=yes
replica_count=3
key=group,storage.owner-uid,storage.owner-gid,network.ping-
timeout,performance.strict-o-direct,network.remote-
dio,cluster.granular-entry-heal,features.shard-block-size
value=virt,36,36,30,on,off,enable,64MB
brick_dirs=<Gluster_Server_IP1>:<brick1_mountpoint>,
<Gluster_Server_IP2>:<brick2_mountpoint>,<Gluster_Server_IP3>:
<brick3_mountpoint>
ignore_volume_errors=no
```

   ii. Run the following command:

```
# gdeploy -c gluster_rep_volume.conf
```

3. **Create a new storage domain using the new gluster volumes**

   a. Log in to the Administration Portal.

   b. Click **Storage** → **Domains** and then click **New Domain**.

   c. Set the **Storage Type** to **GlusterFS** and provide a **Name** for the domain.

   d. Check the **Use managed gluster volume** option and select the volume to use.

   e. Click **OK** to save.

B. Migrate any virtual machine templates

If your virtual machines are created from templates, copy each template to the new Storage Domain.

In Red Hat Virtualization Manager, click **Compute** → **Templates**. For each template to migrate:

1. Click the name of the template you want to migrate.

2. Click the **Disks** subtab.

3. Select the current disk and click **Copy**. The *Copy Disk(s)* dialog appears.

4. Select the new storage domain as the target domain and click **OK**.

C. Migrate virtual machine disks to the new storage domain

For each virtual machine:

1. Click **Storage** → **Disks**.

2. Select the disk to move and click **Move**. The *Move Disk(s)* dialog opens.

3. Select the new storage domain as the target domain and click **OK**.

You can monitor progress in the `Tasks` tab.

D. Verify that disk images migrated correctly

1. Click **Storage** → **Disks**.
2. For each disk:
   a. Select the disk to check.
   b. Click the **Storage** subtab.
   c. Verify that the domain listed is the new storage domain.

IMPORTANT

Do not skip this step. There is no way to retrieve a disk image after a domain is detached and removed, so be sure that all disk images have correctly migrated before you move on.

E. Remove and reclaim the old storage domain

1. Move the old storage domain into maintenance mode.
2. Detach the old storage domain from the data center.
3. Remove the old storage domain from the data center.

5.2. CONFIGURING A LOGICAL VOLUME CACHE (LVMCACHE) FOR IMPROVED PERFORMANCE

If your main storage devices are not Solid State Disks (SSDs), Red Hat recommends configuring a logical volume cache (lvmcache) to achieve the required performance for Red Hat Hyperconverged Infrastructure for Virtualization deployments.

1. **Create the gdeploy configuration file**
   Create a gdeploy configuration file named `lvmcache.conf` that contains at least the following information. Note that the **ssd** value should be the device name, not the device path (for example, use **sdb** not `/dev/sdb`).

   Example lvmcache.conf file

```
[hosts]
<Gluster_Network_NodeA>
<Gluster_Network_NodeB>
<Gluster_Network_NodeC>

[lv1]
action=setup-cache
ssd=sdb
vgname=gluster_vg_sdb
poolname=gluster_thinpool_sdb
```

```
cache_lv=lvcache
cache_lvsize=220GB
#cachemode=writethrough
```

IMPORTANT

Ensure that disks specified as part of this deployment process do not have any partitions or labels.

IMPORTANT

The default cache mode is **`writethrough`**, but **`writeback`** mode is also supported. To avoid the potential for data loss when implementing lvmcache in **`writeback`** mode, two separate SSD/NVMe devices are highly recommended. By configuring the two devices in a RAID-1 configuration (via software or hardware), the potential of data loss from lost writes is reduced significantly.

2. **Run gdeploy**
   Run the following command to apply the configuration specified in **`lvmcache.conf`**.

```
# gdeploy -c lvmcache.conf
```

For further information about lvmcache configuration, see *Red Hat Enterprise Linux 7 LVM Administration*: https://access.redhat.com/documentation/en-US/Red_Hat_Enterprise_Linux/7/html/Logical_Volume_Manager_Administration/LV.html#lvm_cache_volun

CHAPTER 6. CONFIGURE MONITORING

6.1. CONFIGURING EVENT NOTIFICATIONS

To configure which notifications you want to be displayed in Red Hat Virtualization Manager, see Configuring Event Notifications in the Administration Portal in the Red Hat Virtualization 4.2 *Administration Guide*.

PART II. MAINTENANCE TASKS

CHAPTER 7. BASIC OPERATIONS

Some basic operations are required for many administrative and troubleshooting tasks. This section covers how to safely perform basic tasks like shutting down and starting up the hyperconverged cluster.

7.1. CREATING A SHUTDOWN PLAYBOOK

A hyperconverged environment must be shut down in a particular order. The simplest way to do this is to create a shutdown playbook that can be run from the Hosted Engine virtual machine.

The ovirt.shutdown_env role enables Global Maintenance Mode, and initiates shutdown for all virtual machines and hosts in the cluster. Host shutdown is asynchronous. The playbook terminates before hyperconverged hosts are actually shut down.

Prerequisites

- Ensure that the **ovirt.shutdown_env** ansible role is available on the Hosted Engine virtual machine.

  ```
  # yum install ovirt-ansible-shutdown-env -y
  ```

Procedure

1. Log in to the Hosted Engine virtual machine.

2. Create a shutdown playbook for your environment.
   Use the following template to create the playbook file.

   - Replace **ovirt-engine.example.com** with the FQDN of your Hosted Engine virtual machine.

   - Replace **123456** with the password for the **admin@internal** account.

   Example playbook file: shutdown_rhhi-v.yml

   ```
   ---
   - name: oVirt shutdown environment
     hosts: localhost
     connection: local
     gather_facts: false

     vars:
       engine_url: https://ovirt-engine.example.com/ovirt-engine/api
       engine_user: admin@internal
       engine_password: 123456
       engine_cafile: /etc/pki/ovirt-engine/ca.pem

     roles:
       - ovirt.shutdown_env
   ```

7.2. SHUTTING DOWN RHHI FOR VIRTUALIZATION

A hyperconverged environment must be shut down in a particular order. Use an Ansible playbook to automate this process and ensure that your environment is shut down safely.

Prerequisites

- Create a shutdown playbook as described in Creating a shutdown playbook
- Ensure that the **`ovirt.shutdown_env`** ansible role is available on the Hosted Engine virtual machine.

  ```
  # yum install ovirt-ansible-shutdown-env -y
  ```

Procedure

1. Run the shutdown playbook against the Hosted Engine virtual machine.

   ```
   # ansible-playbook -i localhost <shutdown_rhhi-v.yml>
   ```

7.3. STARTING UP A HYPERCONVERGED CLUSTER

Starting up a hyperconverged cluster is more complex than starting up a traditional compute or storage cluster. Follow these instructions to start up your hyperconverged cluster safely.

1. Power on all hosts in the cluster.
2. Ensure that the required services are available.
   a. Verify that the **`glusterd`** service started correctly on all hosts.

   ```
   # systemctl status glusterd
   ● glusterd.service - GlusterFS, a clustered file-system server
      Loaded: loaded (/usr/lib/systemd/system/glusterd.service;
   enabled; vendor preset: disabled)
     Drop-In: /etc/systemd/system/glusterd.service.d
              └─99-cpu.conf
      Active: active (running) since Wed 2018-07-18 11:15:03 IST;
   3min 48s ago
      [...]
   ```

   If glusterd is not started, start it.

   ```
   # systemctl start glusterd
   ```

   b. Verify that host networks are available and hosts have IP addresses assigned to the required interfaces.

   ```
   # ip addr show
   ```

   c. Verify that all hosts are part of the storage cluster (listed as *Peer in Cluster (Connected)*).

   ```
   # gluster peer status

   Number of Peers: 2
   ```

```
Hostname: 10.70.37.101
Uuid: 773f1140-68f7-4861-a996-b1ba97586257
State: Peer in Cluster (Connected)

Hostname: 10.70.37.102
Uuid: fc4e7339-9a09-4a44-aa91-64dde2fe8d15
State: Peer in Cluster (Connected)
```

d. Verify that all bricks are shown as online.

```
# gluster volume status engine
Status of volume: engine
Gluster process                             TCP Port  RDMA Port
Online  Pid
-------------------------------------------------------------
-----------------
Brick 10.70.37.28:/gluster_bricks/engine/en
gine                                        49153     0
Y       23160
Brick 10.70.37.29:/gluster_bricks/engine/en
gine                                        49160     0
Y       12392
Brick 10.70.37.30:/gluster_bricks/engine/en
gine                                        49157     0
Y       15200
Self-heal Daemon on localhost               N/A       N/A
Y       23008
Self-heal Daemon on 10.70.37.30             N/A       N/A
Y       10905
Self-heal Daemon on 10.70.37.29             N/A       N/A
Y       13568

Task Status of Volume engine
-------------------------------------------------------------
-----------------
There are no active volume tasks
```

3. Start the hosted engine virtual machine.

   a. Run the following command on the host that you want to be the hosted engine node.

```
# hosted-engine --vm-start
```

   b. Verify that the hosted engine virtual machine has started correctly.

```
# hosted-engine --vm-status
```

4. Take the hosted engine virtual machine out of Global Maintenance mode.

   a. Log in to the Administration Portal.

   b. Click **Compute** → **Hosts** and select the hosted engine node.

   c. Click **⋮** → **Disable Global HA Maintenance**.

5. Start any other virtual machines using Cockpit.

   a. Click **Compute** → **Virtualization**.

   b. Select any virtual machines you want to start and click **Run**.

CHAPTER 8. UPGRADING TO RED HAT HYPERCONVERGED INFRASTRUCTURE FOR VIRTUALIZATION 1.5

Upgrading involves moving from one version of a product to a newer major release of the same product. This section shows you how to upgrade to Red Hat Hyperconverged Infrastructure for Virtualization 1.5 from version 1.1.

From a component standpoint, this involves the following:

- Upgrading the Hosted Engine virtual machine to Red Hat Virtualization Manager version 4.2.

- Upgrading the physical hosts to Red Hat Virtualization 4.2.

8.1. MAJOR CHANGES IN VERSION 1.5

Be aware of the following differences between Red Hat Hyperconverged Infrastructure for Virtualization 1.5 and previous versions.

Deduplication and compression support with Virtual Data Optimizer (VDO)

Configuring VDO at deployment time lets you reduce the amount of storage space required for data. You can configure Gluster bricks to use deduplication and compression, and monitor and configure notifications for VDO capacity usage so that you know when your storage is running out of space. The space saved by using Virtual Disk Optimization is displayed on the Brick and Volume detail pages of the Cockpit UI. See Understanding VDO and Monitoring VDO for more information.

Configure disaster recovery with failover and failback

Red Hat Hyperconverged Infrastructure for Virtualization now supports backup, failover, and failback to a remote secondary site. See Configuring backup and recovery options for an overview and information on configuring disaster recovery. See Recovering from disaster for the recovery process.

Scale using the user interface

New nodes can now be prepared and configured in Cockpit. See Expanding the hyperconverged cluster by adding a new volume on new nodes using Cockpit for details.

Upgrade using the user interface

Upgrade your deployment using the Administration Portal. See Upgrading Red Hat Hyperconverged Infrastructure for details.

Manage your storage and virtual machines in Cockpit

You can now view and manage your storage and your virtual machines from Cockpit. The Red Hat Virtualization Adminsitration Console is still required for some more complex tasks, such as geo-replication. See Managing Red Hat Gluster Storage using Cockpit for more information.

Configure different devices and device types

Previous versions of RHHI for Virtualization expected each virtualization host to be set up the same way, with the same device types and device names on each. As of version 1.5, you can specify different device and sizes as appropriate for each host and size arbiter bricks appropriately.

Updated user interfaces

Cockpit and the Administration Portal have seen a number of updates to their user interfaces. Operations are now better organised and easier to find, and a number of new options are available.

- Specify additional hosts during Cockpit setup instead of adding them manually after Hosted Engine deployment.

- Reset brick configuration after reinstalling a virtualization host.

Deploy on a single node

Single node deployments of Red Hat Hyperconverged Infrastructure for Virtualization are now supported. See Deploying RHHI for Virtualization in the data center for support limitations and deployment details.

Convert virtualization hosts

Red Hat Virtualization hosts can now be converted into hyperconverged hosts. See Converting a virtualization cluster to a hyperconverged cluster for details.

8.2. UPGRADE WORKFLOW

Red Hat Hyperconverged Infrastructure for Virtualization is a software solution comprised of several different components. Upgrade the components in the following order to minimize disruption to your deployment:

1. Prepare the systems to be upgraded.

2. Upgrade the Hosted Engine virtual machine.

3. Upgrade the virtualization hosts.

8.3. PREPARING TO UPGRADE

8.3.1. Verify brick mount options

If you have configured Virtual Disk Optimizer for any of the volumes in this deployment, you may be affected by Bug 1649507. This bug incorrectly edited the mount options of brick devices.

Edit the **/etc/fstab** file on all hosts to ensure that the following are true:

- Only bricks on VDO volumes have the **x-systemd.requires=vdo.service** option.

- Bricks on VDO volumes have the **_netdev,x-systemd.device-timeout=0** options.

8.3.2. Update to the latest version of the previous release

Ensure that you are using the latest version (4.1.11) of Red Hat Virtualization Manager 4.1 on the hosted engine virtual machine, and the latest version of Red Hat Virtualization 4.1 on the hosted engine node.

See the Red Hat Virtualization Self-Hosted Engine Guide for the Red Hat Virtualization 4.1 update process.

IMPORTANT

Do not proceed with the following prerequisites until you have updated to the latest version of Red Hat Virtualization 4.1.

8.3.3. Update subscriptions

You can check which repositories a machine has access to by running the following command as the root user:

```
# subscription-manager repos --list-enabled
```

- Verify that the Hosted Engine virtual machine is subscribed to the following repositories:
  - **`rhel-7-server-rhv-4.2-manager-rpms`**
  - **`rhel-7-server-rhv-4-manager-tools-rpms`**
  - **`rhel-7-server-rpms`**
  - **`rhel-7-server-supplementary-rpms`**
  - **`jb-eap-7-for-rhel-7-server-rpms`**
  - **`rhel-7-server-ansible-2-rpms`**
- Verify that the Hosted Engine virtual machine is not subscribed to previous versions of the above repositories.
  - **`rhel-7-server-rhv-4.2-manager-rpms`** replaces the **`rhel-7-server-rhv-4.2-rpms`** repository
  - **`rhel-7-server-rhv-4-manager-tools-rpms`** replaces the **`rhel-7-server-rhv-4-tools-rpms`** repository
- Verify that all virtualization hosts are subscribed to the **`rhel-7-server-rhvh-4-rpms`** repository.

Subscribe a machine to a repository by running the following command on that machine:

```
# subscription-manager repos --enable=<repository>
```

8.3.4. Verify that data is not currently being synchronized using geo-replication

- Click the **Tasks** tab at the bottom right of the Manager. Ensure that there are no ongoing tasks related to Data Synchronization. If data synchronization tasks are present, wait until they are complete before beginning the update.
- Stop all geo-replication sessions so that synchronization will not occur during the update. Click the **Geo-replication** subtab and select the session that you want to stop, then click**Stop**. Alternatively, run the following command to stop a geo-replication session:

  ```
  # gluster volume geo-replication <MASTER_VOL> <SLAVE_HOST>::
  <SLAVE_VOL> stop
  ```

8.4. UPGRADING RED HAT HYPERCONVERGED INFRASTRUCTURE FOR VIRTUALIZATION

8.4.1. Upgrading the Hosted Engine virtual machine

1. **Place the cluster into Global Maintenance mode**
   a. Log in to Cockpit.
   b. Click **Virtualization** → **Hosted Engine**.

c. Click **Put this cluster into global maintenance**.

2. **Upgrade Red Hat Virtualization Manager.**

    a. Log in to the Hosted Engine virtual machine.

    b. Upgrade the setup packages:

    ```
    # yum update ovirt*setup\*
    ```

    c. Run **`engine-setup`** and follow the prompts to upgrade the Manager.
    This process can take a while and cannot be aborted, so Red Hat recommends running it inside a **`screen`** session. See How to use the screen command for more information about this function.

    d. Upgrade all other packages.

    ```
    # yum update
    ```

    e. Reboot the Hosted Engine virtual machine to ensure all updates are applied.

    ```
    # reboot
    ```

3. **Restart the Hosted Engine virtual machine.**

    a. Log in to any virtualization host.

    b. Start the Hosted Engine virtual machine.

    ```
    # hosted-engine --vm-start
    ```

    c. Verify the status of the Hosted Engine virtual machine.

    ```
    # hosted-engine --vm-status
    ```

4. **Remove the cluster from Global Maintenance mode.**

    a. Log in to Cockpit.

    b. Click **Virtualization** → **Hosted Engine**.

    c. Click **Remove this cluster from global maintenance**.

8.4.2. Upgrading the virtualization hosts

IMPORTANT

If you are upgrading a host from Red Hat Virtualization 4.2.7 or 4.2.7-1, ensure that the hosted engine virtual machine is not running on that host during the upgrade process. This is related to a bug introduced in Red Hat Enterprise Linux 7.6, BZ#1641798, which affects these versions of Red Hat Virtualization.

To work around this issue, stop the hosted engine virtual machine before upgrading a host, and start it on another host.

```
[root@host1] # hosted-engine --vm-shutdown
```

```
[root@host2] # hosted-engine --vm-start
```

Perform the following steps on **one virtualization host at a time**.

1. **Upgrade the virtualization host.**
   a. In the Manager, click **Compute** → **Hosts** and select a node.
   b. Click **Installation** → **Upgrade**.
   c. Click **OK** to confirm the upgrade.
      Wait for the upgrade to complete, and for the host to become available again.
2. **Verify self-healing is complete before upgrading the next host.**
   a. Click the name of the host.
   b. Click the **Bricks** tab.
   c. Verify that the **Self-Heal Info** column of all bricks is listed as **`OK`** before upgrading the next host.

Troubleshooting

- If upgrading a virtualization host fails because of a conflict with the **`rhvm-appliance`** package, log in to the virtualization host and follow the steps in RHV: RHV-H Upgrade failed before continuing.

8.4.3. Cleaning up after upgrading

You may need to update the **`virt`** group profile after you upgrade. This is necessary because of BZ1643730, also described in the release notes.

1. Verify that the following parameters are included in the **`/var/lib/glusterd/groups/virt`** file on each server, adding them if necessary.

```
cluster.choose-local=off
client.event-threads=4
server.event-threads=4
performance.client-io-threads=on
```

2. If you needed to update the **`virt`** group profile, run the following command on each volume to apply the updated profile.

   ```
   # gluster volume set <volname> group virt
   ```

CHAPTER 9. MONITORING RED HAT HYPERCONVERGED INFRASTRUCTURE FOR VIRTUALIZATION

9.1. MONITORING VIRTUAL DATA OPTIMIZER (VDO)

Monitoring VDO helps in understanding when the physical storage is running out of space. Physical space in VDO needs to be monitored like thin provisioned storage. VDO devices should use thin provisioning because more logical space will be available and VDO space will be used in a more effective way. By default thin provisioning is enabled and it can be unchecked as required.

You can check available blocks, used space, and device information by clicking on **View Details**.

9.1.1. Monitoring VDO using the command line interface

There are several options for monitoring VDO using the command line interface.

The `vdostats` command

This command displays volume statistics including available blocks, number of blocks used, device name, percentage of physical blocks saved, and percentage of physical blocks on a VDO volume. For more information on vdostats, see the manual page: **`man vdostats`**.

The `vdo status` command

This command reports VDO system and volume status in YAML format.

The `/sys/kvdo/<vdo_volume>/statistics` directory

Files in this directory include volume statistics for VDO. You can read these files instead of using the **`vdostats`** command.

9.1.2. Monitoring VDO using the Cockpit UI

Events related to VDO usage are displayed under the **Notifications** tab. Events provide information about the physical space remaining on the VDO volume, and keep you up to date about whether more physical space is needed.

Table 9.1. Types of Event Notification

Type	Text	Actions
Warn	Warning, low confirmed disk space. StorageDomainName domain has DiskSpace GB of confirmed free space.	• Delete data that is not required. • Replace existing disks with larger disks. • Add more Red Hat Gluster Storage servers and expand the volume across the new servers. • Add more disks and expand the logical volumes underlying the Gluster volume.

CHAPTER 10. ADD VIRTUALIZATION HOSTS TO RED HAT VIRTUALIZATION MANAGER

Follow this process to allow Red Hat Virtualization Manager to manage an existing virtualization host.

1. Log in to Red Hat Virtualization Manager.
2. Click **Compute** → **Hosts**.
3. Click **New**. The **New Host** window opens.
4. On the **General** tab, specify the following details about your virtualization host.
   - **`Host Cluster`**
   - **`Name`**
   - **`Hostname`**
   - **`Password`**
5. On the **General** tab, click the **Advanced Parameters** dropdown, and uncheck the **`Automatically configure host firewall`** checkbox.
6. Click **OK**.

CHAPTER 11. REINSTALLING A VIRTUALIZATION HOST

Some configuration changes require a virtualization host to be reinstalled before the configuration change can take effect. Follow these steps to reinstall a virtualization host.

1. Log in to Red Hat Virtualization Manager.
2. Click **Compute → Hosts**.
3. Select the host and click **Management** > **Maintenance** > **OK** to place this host in Maintenance mode.
4. Click **Installation** > **Reinstall** to open the Reinstall window.
5. On the General tab, uncheck the **Automatically Configure Host firewall** checkbox.
6. On the **Hosted Engine** tab, set the value of **Choose hosted engine deployment action** to **`Deploy`**.
7. Click **OK** to reinstall the host.

CHAPTER 12. REPLACING THE PRIMARY STORAGE HOST

Follow this section to replace the Red Hat Gluster Storage host that initiated the peering process.

IMPORTANT

When self-signed encryption is enabled, replacing a node is a disruptive process that requires virtual machines and the Hosted Engine to be shut down.

1. (Optional) If encryption using a Certificate Authority is enabled, follow the steps under *Expanding Volumes* in the Network Encryption chapter of the Red Hat Gluster Storage 3.4 *Administration Guide*.

2. Move the server to be replaced into Maintenance mode.

   a. In Red Hat Virtualization Manager, click **Compute** → **Hosts** and select the host to replace.

   b. Click **Management** → **Maintenance** and click **OK** to move the host to Maintenance mode.

3. Install the replacement host
   Follow the instructions in *Deploying Red Hat Hyperconverged Infrastructure for Virtualization for Virtualization* to install the physical machine and configure storage on the new host.

   a. Installing virtualization hosts

   b. Configuring Public Key based SSH Authentication

   c. Configuring RHGS for Hosted Engine using the Cockpit UI

4. Prepare the replacement host.

   a. Create a file called **replace_host_prep.conf** based on the template provided in Section B.2, “Example gdeploy configuration file for preparing a replacement host”.

   b. From a host with **gdeploy** installed (usually the server that hosts the Hosted Engine virtual machine), run gdeploy using the new configuration file:

   ```
   # gdeploy -c replace_host_prep.conf
   ```

5. (Optional) If encryption with self-signed certificates is enabled:

   a. Generate the private key and self-signed certificate on the replacement host. See the Red Hat Gluster Storage *Administration Guide* for details: https://access.redhat.com/documentation/en-us/red_hat_gluster_storage/3.4/html/administration_guide/chap-network_encryption#chap-Network_Encryption-Prereqs.

   b. On a healthy host, create a copy of the **/etc/ssl/glusterfs.ca** file.

   ```
   # cp /etc/ssl/glusterfs.ca /etc/ssl/glusterfs.ca.bk
   ```

   c. Append the new host’s certificate to the content of the original **/etc/ssl/glusterfs.ca** file.

d. Distribute the **`/etc/ssl/glusterfs.ca`** file to all hosts in the cluster, including the new host.

e. Run the following command on the replacement host to enable management encryption:

```
# touch /var/lib/glusterd/secure-access
```

f. Include the new host in the value of the **`auth.ssl-allow`** volume option by running the following command for each volume.

```
# gluster volume set <volname> auth.ssl-allow "<old_host1>,
<old_host2>,<new_host>"
```

g. Restart the glusterd service on all hosts.

```
# systemctl restart glusterd
```

h. Follow the steps in Section 4.1, “Configuring TLS/SSL using self-signed certificates” to remount all gluster processes.

6. Add the replacement host to the cluster.
Run the following command from any host already in the cluster.

```
# peer probe <new_host>
```

7. Move the Hosted Engine into Maintenance mode:

```
# hosted-engine --set-maintenance --mode=global
```

8. Stop the ovirt-engine service.

```
# systemctl stop ovirt-engine
```

9. Update the database.

```
# hosted-engine --set-shared-config storage <new_host_IP>:/engine
```

10. Start the ovirt-engine service.

```
# systemctl start ovirt-engine
```

11. Stop all virtual machines except the Hosted Engine.

12. Move all storage domains **except** the Hosted Engine domain into Maintenance mode.

13. Stop the Hosted Engine virtual machine.
Run the following command on the existing server that hosts the Hosted Engine.

```
# hosted-engine --vm-shutdown
```

14. Stop high availability services on all hosts.

```
# systemctl stop ovirt-ha-agent
# systemctl stop ovirt-ha-broker
```

15. Disconnect Hosted Engine storage from the virtualization host.
    Run the following command on the existing server that hosts the Hosted Engine.

    ```
    # hosted-engine --disconnect-storage
    ```

16. Update the Hosted Engine configuration file.
    Edit the storage parameter in the **`/etc/ovirt-hosted-engine/hosted-engine.conf`** file to use the replacement host.

    ```
    storage=<new_server_IP>:/engine
    ```

    NOTE

    To configure the Hosted Engine for new hosts, use the command:

    ```
    # hosted-engine --set-shared-config storage
    <new_server_IP>:/engine
    ```

17. Restart high availability services on all hosts.

    ```
    # systemctl restart ovirt-ha-agent
    # systemctl restart ovirt-ha-broker
    ```

18. Reboot the existing and replacement hosts.
    Wait until all hosts are available before continuing.

19. Take the Hosted Engine out of Maintenance mode.

    ```
    # hosted-engine --set-maintenance --mode=none
    ```

20. Verify that the replacement host is used.
    On all virtualization hosts, verify that the **engine** volume is mounted from the replacement host by checking the IP address in the output of the **`mount`** command.

21. Activate storage domains.
    Verify that storage domains mount using the IP address of the replacement host.

22. Using the RHV Management UI, add the replacement host.
    Specify that the replacement host be used to host the Hosted Engine.

23. Move the replacement host into Maintenance mode.

    ```
    # hosted-engine --set-maintenance --mode=global
    ```

24. Reboot the replacement host.
    Wait until the host is back online before continuing.

25. Activate the replacement host from the RHV Management UI.

Ensure that all volumes are mounted using the IP address of the replacement host.

26. Replace engine volume brick.
    Replace the brick on the old host that belongs to the engine volume with a new brick on the replacement host.

    a. Click **Storage** → **Volumes** and select the volume.

    b. Click the **Bricks** subtab.

    c. Select the brick to replace, and then click **Replace brick**.

    d. Select the host that hosts the brick being replaced.

    e. In the *Replace brick* window, provide the path to the new brick.

27. Remove the old host.

    a. Click **Compute** → **Hosts** and select the old host.

    b. Click **Management** → **Maintenance** to move the host to maintenance mode.

    c. Click **Remove**. The *Remove Host(s)* confirmation dialog appears.

    d. If there are still volume bricks on this host, or the host is non-responsive, check the **Force Remove** checkbox.

    e. Click **OK**.

    f. Detach the old host from the cluster.

    ```
    # gluster peer detach <old_host_IP> force
    ```

28. On the replacement host, run the following command to remove metadata from the previous host.

    ```
    # hosted-engine --clean-metadata --host-id=<old_host_id> --force-clean
    ```

CHAPTER 13. REPLACING A GLUSTER STORAGE HOST

If a Red Hat Gluster Storage host needs to be replaced, there are two options for the replacement host:

1. Replace the host with a new host that has a different fully-qualified domain name by following the instructions in Section 13.1, “Replacing a Gluster Storage Host (Different FQDN)”.

2. Replace the host with a new host that has the same fully-qualified domain name by following the instructions in Section 13.2, “Replacing a Gluster Storage Node (Same FQDN)”.

Follow the instructions in whichever section is appropriate for your deployment.

13.1. REPLACING A GLUSTER STORAGE HOST (DIFFERENT FQDN)

IMPORTANT

When self-signed encryption is enabled, replacing a node is a disruptive process that requires virtual machines and the Hosted Engine to be shut down.

1. Install the replacement host
   Follow the instructions in *Deploying Red Hat Hyperconverged Infrastructure for Virtualization for Virtualization* to install the physical machine.

   a. Installing host physical machines

   b. Configuring Public Key based SSH Authentication

2. Stop any existing geo-replication sessions

   ```
   # gluster volume geo-replication MASTER_VOL SLAVE_HOST::SLAVE_VOL
   stop
   ```

   For further information, see the Red Hat Gluster Storage *Administration Guide*: https://access.redhat.com/documentation/en-us/red_hat_gluster_storage/3.4/html/administration_guide/sect-starting_geo-replication#Stopping_a_Geo-replication_Session.

3. Move the host to be replaced into Maintenance mode
   Perform the following steps in Red Hat Virtualization Manager:

   a. Click **Compute** → **Hosts** and select the Red Hat Gluster Storage host in the results list.

   b. Click **Management** → **Maintenance** and click **OK** to move the host to Maintenance mode.

4. Prepare the replacement host

   a. Configure key-based SSH authentication without a password
      Configure key-based SSH authentication without a password from a physical machine still in the cluster to the replacement host. For details, see https://access.redhat.com/documentation/en-us/red_hat_hyperconverged_infrastructure_for_virtualization/1.5/html/deploying_red_hat_hyper
      configure-key-based-ssh-auth.

   b. Prepare the replacement host

Create a file called **replace_host_prep.conf** based on the template provided in Section B.2, "Example gdeploy configuration file for preparing a replacement host".

From a host with **gdeploy** installed (usually the host that hosts the Hosted Engine), run gdeploy using the new configuration file:

```
# gdeploy -c replace_host_prep.conf
```

5. Create replacement brick directories
Ensure the new directories are owned by the **vdsm** user and the **kvm** group.

```
# mkdir /gluster_bricks/engine/engine
# chmod vdsm:kvm /gluster_bricks/engine/engine
# mkdir /gluster_bricks/data/data
# chmod vdsm:kvm /gluster_bricks/data/data
# mkdir /gluster_bricks/vmstore/vmstore
# chmod vdsm:kvm /gluster_bricks/vmstore/vmstore
```

6. (Optional) If encryption is enabled

   a. Generate the private key and self-signed certificate on the new server using the steps in the Red Hat Gluster Storage *Administration Guide*: https://access.redhat.com/documentation/en-us/red_hat_gluster_storage/3.4/html/administration_guide/chap-network_encryption#chap-Network_Encryption-Prereqs.
   If encryption using a Certificate Authority is enabled, follow the steps under *Expanding Volumes* in the Network Encryption chapter of the Red Hat Gluster Storage 3.4 *Administration Guide*.

   b. Add the new host's certificate to existing certificates.

      i. On a healthy host, make a backup copy of the **/etc/ssl/glusterfs.ca** file.

      ii. Add the new host's certificate to the **/etc/ssl/glusterfs.ca** file on the healthy host.

      iii. Distribute the updated **/etc/ssl/glusterfs.ca** file to all other hosts, including the new host.

   c. Enable management encryption
   Run the following command on the new host to enable management encryption:

```
# touch /var/lib/glusterd/secure-access
```

   d. Include the new host in the value of the **auth.ssl-allow** volume option by running the following command for each volume.

```
# gluster volume set <volname> auth.ssl-allow "<old_host1>,
<old_host2>,<new_host>"
```

   e. Restart the glusterd service on all hosts

```
# systemctl restart glusterd
```

f. If encryption uses self-signed certificates, follow the steps in Section 4.1, “Configuring TLS/SSL using self-signed certificates” to remount all gluster processes.

7. Add the new host to the existing cluster

   a. Run the following command from one of the healthy hosts:

      ```
      # gluster peer probe <new_host>
      ```

   b. Add the new host to the existing cluster

      i. Click **Compute** → **Hosts** and then click **New** to open the *New Host* dialog.

      ii. Provide a **Name**, **Address**, and **Password** for the new host.

      iii. Uncheck the **Automatically configure host firewall** checkbox, as firewall rules are already configured by gdeploy.

      iv. In the **Hosted Engine** tab of the *New Host* dialog, set the value of **Choose hosted engine deployment action** to **`Deploy`**.

      v. Click **OK**.

      vi. When the host is available, click the name of the new host.

      vii. Click the **Network Interfaces** subtab and then click **Setup Host Networks**. The *Setup Host Networks* dialog appears.

      viii. Drag and drop the network you created for gluster to the IP associated with this host, and click **OK**.
      See the Red Hat Virtualization 4.2 Self-Hosted Engine Guide for further details: https://access.redhat.com/documentation/en-us/red_hat_virtualization/4.2/html/self-hosted_engine_guide/chap-installing_additional_hosts_to_a_self-hosted_environment.

8. Configure and mount shared storage on the new host

   ```
   # cp /etc/fstab /etc/fstab.bk
   # echo "<new_host>:/gluster_shared_storage
   /var/run/gluster/shared_storage/ glusterfs defaults 0 0" >>
   /etc/fstab
   # mount /gluster_shared_storage
   ```

9. Replace the old brick with the brick on the new host

   a. In Red Hat Virtualization Manager, click **Storage** → **Volumes** and select the volume.

   b. Click the **Bricks** subtab.

   c. Select the brick that you want to replace and click **Replace Brick**. The *Replace Brick* dialog appears.

   d. Specify the **Host** and the **Brick Directory** of the new brick.

   e. Verify that brick heal completes successfully.

10. Click **Compute** → **Hosts**.

11. Select the old host and click **Remove**.
    Use `gluster peer status` to verify that that the old host is no longer part of the cluster. If the old host is still present in the status output, run the following command to forcibly remove it:

    ```
    # gluster peer detach <old_host> force
    ```

12. Clean old host metadata.

    ```
    # hosted-engine --clean-metadata --host-id=<old_host_id> --force-
    clean
    ```

13. Set up new SSH keys for geo-replication of new brick.

    ```
    # gluster system:: execute gsec_create
    ```

14. Recreate geo-replication session and distribute new SSH keys.

    ```
    # gluster volume geo-replication <MASTER_VOL> <SLAVE_HOST>::
    <SLAVE_VOL> create push-pem force
    ```

15. Start the geo-replication session.

    ```
    # gluster volume geo-replication <MASTER_VOL> <SLAVE_HOST>::
    <SLAVE_VOL> start
    ```

13.2. REPLACING A GLUSTER STORAGE NODE (SAME FQDN)

IMPORTANT

When self-signed encryption is enabled, replacing a node is a disruptive process that requires virtual machines and the Hosted Engine to be shut down.

1. (Optional) If encryption using a Certificate Authority is enabled, follow the steps under *Expanding Volumes* in the Network Encryption chapter of the Red Hat Gluster Storage 3.4 *Administration Guide*.

2. Move the host to be replaced into Maintenance mode

    a. In Red Hat Virtualization Manager, click **Compute** → **Hosts** and select the Red Hat Gluster Storage host.

    b. Click **Management** → **Maintenance**.

    c. Click **OK** to move the host to Maintenance mode.

3. Prepare the replacement host
   Follow the instructions in *Deploying Red Hat Hyperconverged Infrastructure for Virtualization for Virtualization* to install the physical machine and configure storage on the new host.

    a. Installing host physical machines

    b. Configuring Public Key based SSH Authentication

   c. Configuring RHGS for Hosted Engine using the Cockpit UI

4. Prepare the replacement host

   a. Create a file called **replace_host_prep.conf** based on the template provided in Section B.2, “Example gdeploy configuration file for preparing a replacement host”.

   b. From a host with **gdeploy** installed (usually the host that hosts the Hosted Engine), run gdeploy using the new configuration file:

```
# gdeploy -c replace_host_prep.conf
```

5. (Optional) If encryption with self-signed certificates is enabled

   a. Generate the private key and self-signed certificate on the replacement host. See the Red Hat Gluster Storage *Administration Guide* for details: https://access.redhat.com/documentation/en-us/red_hat_gluster_storage/3.4/html/administration_guide/chap-network_encryption#chap-Network_Encryption-Prereqs.

   b. On a healthy host, make a backup copy of the **/etc/ssl/glusterfs.ca** file:

```
# cp /etc/ssl/glusterfs.ca /etc/ssl/glusterfs.ca.bk
```

   c. Append the new host’s certificate to the content of the **/etc/ssl/glusterfs.ca** file.

   d. Distribute the **/etc/ssl/glusterfs.ca** file to all hosts in the cluster, including the new host.

   e. Run the following command on the replacement host to enable management encryption:

```
# touch /var/lib/glusterd/secure-access
```

6. Replace the host machine
   Follow the instructions in the Red Hat Gluster Storage *Administration Guide* to replace the host: https://access.redhat.com/documentation/en-us/red_hat_gluster_storage/3.4/html/administration_guide/sect-replacing_hosts#Replacing_a_Host_Machine_with_the_Same_Hostname.

7. Restart the glusterd service on all hosts

```
# systemctl restart glusterd
```

8. Verify that all hosts reconnect

```
# gluster peer status
```

9. **(Optional)** If encryption uses self-signed certificates, follow the steps in Section 4.1, “Configuring TLS/SSL using self-signed certificates” to remount all gluster processes.

10. Verify that all hosts reconnect and that brick heal completes successfully

```
# gluster peer status
```

11. Refresh fingerprint

    a. In Red Hat Virtualization Manager, click **Compute** → **Hosts** and select the new host.

    b. Click **Edit**.

    c. Click **Advanced Parameters** on the **General** tab.

    d. Click **fetch** to fetch the fingerprint from the host.

    e. Click **OK**.

12. Click **Installation** → **Reinstall** and provide the root password when prompted.

13. On the **Hosted Engine** tab set the value of **Choose hosted engine deployment action** to **Deploy**.

14. Attach the gluster network to the host

    a. Click **Compute** → **Hosts** and click the name of the host.

    b. Click the **Network Interfaces** subtab and then click **Setup Host Networks**.

    c. Drag and drop the newly created network to the correct interface.

    d. Ensure that the **Verify connectivity between Host and Engine** checkbox is checked.

    e. Ensure that the **Save network configuration** checkbox is checked.

    f. Click **OK** to save.

15. **Verify the health of the network**
    Check the state of the host’s network. If the network interface enters an "Out of sync" state or does not have an IPv4 Address, click **Management** → **Refresh Capabilities**.

CHAPTER 14. RECOVERING FROM DISASTER

This chapter explains how to restore your cluster to a working state after a disk or server failure.

You must have configured disaster recovery options previously in order to use this chapter. See Configuring backup and recovery options for details.

14.1. MANUALLY RESTORING DATA FROM A BACKUP VOLUME

This section covers how to restore data from a remote backup volume to a freshly installed replacement deployment of Red Hat Hyperconverged Infrastructure for Virtualization.

To do this, you must:

1. Install and configure a replacement deployment according to the instructions in Deploying Red Hat Hyperconverged Infrastructure for Virtualization.

14.1.1. Restoring a volume from a geo-replicated backup

1. Install and configure a replacement Hyperconverged Infrastructure deployment
   For instructions, refer to *Deploying Red Hat Hyperconverged Infrastructure for Virtualization*: https://access.redhat.com/documentation/en-us/red_hat_hyperconverged_infrastructure_for_virtualization/1.5/html/deploying_red_hat_hypercon

2. Import the backup of the storage domain
   From the new Hyperconverged Infrastructure deployment, in Red Hat Virtualization Manager:

   a. Click **Storage** → **Domains**.

   b. Click **Import Domain**. The *Import Pre-Configured Domain* window opens.

   c. In the **Storage Type** field, specify **GlusterFS**.

   d. In the **Name** field, specify a name for the new volume that will be created from the backup volume.

   e. In the **Path** field, specify the path to the backup volume.

   f. Click **OK**. The following warning appears, with any active data centers listed below:

   ```
   This operation might be unrecoverable and destructive!

   Storage Domain(s) are already attached to a Data Center.
   Approving this operation might cause data corruption if
   both Data Centers are active.
   ```

   g. Check the **Approve operation** checkbox and click **OK**.

3. Determine a list of virtual machines to import

   a. Determine the imported domain’s identifier by running the following command:

   ```
   # curl -v -k -X GET -u "admin@internal:password" -H "Accept:
   application/xml" https://$ENGINE_FQDN/ovirt-
   engine/api/storagedomains/
   ```

For example:

```
# curl -v -k -X GET -u "admin@example.com:mybadpassword" -H
"Accept: application/xml" https://10.0.2.1/ovirt-
engine/api/storagedomains/
```

b. Determine the list of unregistered disks by running the following command:

```
# curl -v -k -X GET -u "admin@internal:password" -H "Accept:
application/xml" "https://$ENGINE_FQDN/ovirt-
engine/api/storagedomains/DOMAIN_ID/vms;unregistered"
```

For example:

```
# curl -v -k -X GET -u "admin@example.com:mybadpassword" -H
"Accept: application/xml" "https://10.0.2.1/ovirt-
engine/api/storagedomains/5e1a37cf-933d-424c-8e3d-
eb9e40b690a7/vms;unregistered"
```

4. Perform a partial import of each virtual machine to the storage domain

a. Determine cluster identifier
The following command returns the cluster identifier.

```
# curl -v -k -X GET -u "admin@internal:password" -H "Accept:
application/xml" https://$ENGINE_FQDN/ovirt-engine/api/clusters/
```

For example:

```
# curl -v -k -X GET -u "admin@example:mybadpassword" -H "Accept:
application/xml" https://10.0.2.1/ovirt-engine/api/clusters/
```

b. Import the virtual machines
The following command imports a virtual machine without requiring all disks to be available in the storage domain.

```
# curl -v -k -u 'admin@internal:password' -H "Content-type:
application/xml" -d '<action> <cluster id="CLUSTER_ID"></cluster>
<allow_partial_import>true</allow_partial_import> </action>'
"https://ENGINE_FQDN/ovirt-
engine/api/storagedomains/DOMAIN_ID/vms/VM_ID/register"
```

For example:

```
# curl -v -k -u 'admin@example.com:mybadpassword' -H "Content-
type: application/xml" -d '<action> <cluster id="bf5a9e9e-5b52-
4b0d-aeba-4ee4493f1072"></cluster>
<allow_partial_import>true</allow_partial_import> </action>'
"https://10.0.2.1/ovirt-engine/api/storagedomains/8d21980a-a50b-
45e9-9f32-cd8d2424882e/e164f8c6-769a-4cbd-ac2a-
ef322c2c5f30/register"
```

For further information, see the Red Hat Virtualization *REST API Guide*: https://access.redhat.com/documentation/en-us/red_hat_virtualization/4.2/html/rest_api_guide/.

5. Migrate the partially imported disks to the new storage domain
   In Red Hat Virtualization Manager, click **Storage** → **Disks**, and Click the **Move Disk** option. Move the imported disks from the synced volume to the replacement cluster's storage domain. For further information, see the Red Hat Virtualization Administration Guide.

6. Attach the restored disks to the new virtual machines
   Follow the instructions in the Red Hat Virtualization Virtual Machine Management Guide to attach the replacement disks to each virtual machine.

14.2. FAILING OVER TO A SECONDARY CLUSTER

This section covers how to fail over from your primary cluster to a remote secondary cluster in the event of server failure.

1. Configure failover to a remote cluster.

2. Verify that the mapping file for the source and target clusters remains accurate.

3. Run the failover playbook with the **`fail_over`** tag.

```
# ansible-playbook dr-rhv-failover.yml --tags "fail_over"
```

14.3. FAILING BACK TO A PRIMARY CLUSTER

This section covers how to fail back from your secondary cluster to the primary cluster after you have corrected the cause of a server failure.

1. Prepare the primary cluster for failback by running the cleanup playbook with the **`clean_engine`** tag.

```
# ansible-playbook dr-cleanup.yml --tags "clean_engine"
```

2. Verify that the mapping file for the source and target clusters remains accurate.

3. Execute failback by running the failback playbook with the **`fail_back`** tag.

```
# ansible-playbook dr-cleanup.yml --tags "fail_back"
```

14.4. STOPPING A GEO-REPLICATION SESSION USING RHV MANAGER

Stop a geo-replication session when you want to prevent data being replicated from an active source volume to a passive target volume via geo-replication.

1. **Verify that data is not currently being synchronized**
   Click the **Tasks** icon at the top right of the Manager, and review the Tasks page.

   Ensure that there are no ongoing tasks related to Data Synchronization.

If data synchronization tasks are present, wait until they are complete.

2. **Stop the geo-replication session**

    a. Click **Storage** → **Volumes**.

    b. Click the name of the volume that you want to prevent geo-replicating.

    c. Click the **Geo-replication** subtab.

    d. Select the session that you want to stop, then click **Stop**.

14.5. TURNING OFF SCHEDULED BACKUPS BY DELETING THE GEO-REPLICATION SCHEDULE

You can stop scheduled backups via geo-replication by deleting the geo-replication schedule.

1. Log in to Red Hat Virtualization Manager on any source node.

2. Click **Storage** → **Domains**.

3. Click the name of the storage domain that you want to back up.

4. Click the **Remote Data Sync Setup** subtab.

5. Click **Setup**.
   The *Setup Remote Data Synchronization* window opens.

6. In the **Recurrence** field, select a recurrence interval type of **NONE** and click **OK**.

7. (Optional) Remove the geo-replication session
   Run the following command from the geo-replication master node:

   ```
   # gluster volume geo-replication MASTER_VOL SLAVE_HOST::SLAVE_VOL
   delete
   ```

   You can also run this command with the **`reset-sync-time`** parameter. For further information about this parameter and deleting a geo-replication session, see Deleting a Geo-replication Session in the Red Hat Gluster Storage 3.4 *Administration Guide*.

PART III. REFERENCE MATERIAL

APPENDIX A. FENCING POLICIES FOR RED HAT GLUSTER STORAGE

The following fencing policies are required for Red Hat Hyperconverged Infrastructure for Virtualization (RHHI for Virtualization) deployments. They ensure that hosts are not shut down in situations where brick processes are still running, or when shutting down the host would remove the volume's ability to reach a quorum.

These policies can be set in the **New Cluster** or **Edit Cluster** window in Red Hat Virtualization Manager when Red Hat Gluster Storage functionality is enabled.

Skip fencing if gluster bricks are up

Fencing is skipped if bricks are running and can be reached from other peers.

Skip fencing if gluster quorum not met

Fencing is skipped if bricks are running and shutting down the host will cause loss of quorum

These policies are checked after all other fencing policies when determining whether a node is fenced.

Additional fencing policies may be useful for your deployment. For further details about fencing, see the Red Hat Virtualization *Technical Reference*: https://access.redhat.com/documentation/en-us/red_hat_virtualization/4.2/html/technical_reference/fencing.

APPENDIX B. EXAMPLE GDEPLOY CONFIGURATION FILES

B.1. EXAMPLE GDEPLOY CONFIGURATION FILE FOR SETTING UP TLS/SSL

set_up_encryption.conf

```
# IPs that corresponds to the Gluster Network
[hosts]
<Gluster_Network_NodeA>
<Gluster_Network_NodeB>
<Gluster_Network_NodeC>

# STEP-1: Generate Keys, Certificates & CA files
# The following section generates the keys,certicates, creates
# ca file and distributes it to all the hosts
[volume1]
action=enable-ssl
volname=engine
ssl_clients=<Gluster_Network_NodeA>,<Gluster_Network_NodeB>,
<Gluster_Network_NodeC>
ignore_volume_errors=no

# As the certificates are already generated, its enough to stop
# rest of the volumes,set TLS/SSL related volume options, and
# start the volume

# STEP-2: Stop all the volumes
[volume2]
action=stop
volname=vmstore

[volume3]
action=stop
volname=data

# STEP-3: Set volume options on all the volumes to enable TLS/SSL on the
volumes
[volume4]
action=set
volname=vmstore
key=client.ssl,server.ssl,auth.ssl-allow
value=on,on,"<Gluster_Network_NodeA>;<Gluster_Network_NodeB>;
<Gluster_Network_NodeC>"
ignore_volume_errors=no

[volume5]
action=set
volname=data
key=client.ssl,server.ssl,auth.ssl-allow
value=on,on,"<Gluster_Network_NodeA>;<Gluster_Network_NodeB>;
<Gluster_Network_NodeC>"
ignore_volume_errors=no
```

```
# STEP-4: Start all the volumes
[volume6]
action=start
volname=vmstore

[volume7]
action=start
volname=data
```

B.2. EXAMPLE GDEPLOY CONFIGURATION FILE FOR PREPARING A REPLACEMENT HOST

IMPORTANT

If the disks must be replaced as well as the host server, ensure that the **[pv]**, **[vg]**, and **[lv]** sections are not commented out of this file.

For details about how to safely replace a host, see Chapter 13, *Replacing a Gluster Storage Host*.

replace_node_prep.conf

```
# EDITME: @1: Change to IP addresses of the network intended for gluster
traffic
# Values provided here are used to probe the gluster hosts.
[hosts]
10.70.X1.Y1

#EDITME : @2: Change to IP addresses of the network intended for gluster
traffic
#of the node which is going to be replaced.
[script1]
action=execute
ignore_script_errors=no
file=/usr/share/ansible/gdeploy/scripts/grafton-sanity-check.sh -d sdc -h
10.70.X1.Y1

# EDITME: @3: Specify the number of data disks in RAID configuration
[disktype]
raid6

[diskcount]
4

[stripesize]
256

# EDITME : @4:  UNCOMMENT SECTION (RHEL ONLY) :Provide the subscription
details
# Register to RHSM only on the node which needs to be replaced
#[RH-subscription1:10.70.X1.Y1]
#action=register
#username=<username>
#password=<passwd>
#pool=<pool-id>
```

```
#[RH-subscription2]
#action=disable-repos
#repos=

#[RH-subscription3]
#action=enable-repos
#repos=rhel-7-server-rpms,rh-gluster-3-for-rhel-7-server-rpms,rhel-7-
server-rhv-4-mgmt-agent-rpms

#[yum1]
#action=install
#packages=vdsm,vdsm-gluster,ovirt-hosted-engine-setup,screen,gluster-
nagios-addons
#update=yes

[service1]
action=enable
service=chronyd

[service2]
action=restart
service=chronyd

[shell1]
action=execute
command=gluster pool list

[shell2]
action=execute
command=vdsm-tool configure --force

# Disable multipath
[script3]
action=execute
file=/usr/share/ansible/gdeploy/scripts/disable-multipath.sh

#EDIT ME: @5: UNCOMMENT SECTIONS ONLY: if original brick disks have to be
replaced.
#[pv1]
#action=create
#devices=sdc
#ignore_pv_errors=no

#[vg1]
#action=create
#vgname=gluster_vg_sdc
#pvname=sdc
#ignore_vg_errors=no


#[lv2:10.70.X1:Y1]
#action=create
#poolname=gluster_thinpool_sdc
#ignore_lv_errors=no
#vgname=gluster_vg_sdc
```

```
#lvtype=thinpool
#poolmetadatasize=16GB
#size=14TB

#[lv3:10.70.X1:Y1]
#action=create
#lvname=gluster_lv_engine
#ignore_lv_errors=no
#vgname=gluster_vg_sdc
#mount=/gluster_bricks/engine
#size=100GB
#lvtype=thick


#[lv5:10.70.X1:Y1]
#action=create
#lvname=gluster_lv_data
#ignore_lv_errors=no
#vgname=gluster_vg_sdc
#mount=/gluster_bricks/data
#lvtype=thinlv
#poolname=gluster_thinpool_sdc
#virtualsize=12TB


#[lv7:10.70.X1:Y1]
#action=create
#lvname=gluster_lv_vmstore
#ignore_lv_errors=no
#vgname=gluster_vg_sdc
#mount=/gluster_bricks/vmstore
#lvtype=thinlv
#poolname=gluster_thinpool_sdc
#virtualsize=1TB

#[selinux]
#yes

#[lv9:10.70.X1:Y1]
#action=setup-cache
#ssd=sdb
#vgname=gluster_vg_sdc
#poolname=lvthinpool
#cache_lv=lvcache
#cache_lvsize=180GB

[service3]
action=start
service=glusterd
slice_setup=yes

[firewalld]
action=add
ports=111/tcp,2049/tcp,54321/tcp,5900/tcp,5900-
6923/tcp,5666/tcp,16514/tcp,54322/tcp
services=glusterfs
```

```
[script2]
action=execute
file=/usr/share/ansible/gdeploy/scripts/disable-gluster-hooks.sh
```

B.3. EXAMPLE GDEPLOY CONFIGURATION FILE FOR SCALING TO ADDITIONAL NODES

add_nodes.conf

```
# Add the hosts to be added
[hosts]
<Gluster_Network_NodeD>
<Gluster_Network_NodeE>
<Gluster_Network_NodeF>

# If using RHEL 7 as platform, enable required repos
# RHVH has all the packages available
#[RH-subscription]
#ignore_register_errors=no
#ignore_attach_pool_errors=no
#ignore_enable_errors=no
#action=register
#username=<username>
#password=<mypassword>
#pool=<pool-id>
#repos=rhel-7-server-rpms,rh-gluster-3-for-rhel-7-server-rpms,rhel-7-
server-rhv-4-mgmt-agent-rpms
#disable-repos=yes

# If using RHEL 7 as platform, have the following section to install
packages
[yum1]
action=install
packages=vdsm-gluster,ovirt-hosted-engine-setup,screen
update=yes
gpgcheck=yes
ignore_yum_errors=no

# enable chronyd
[service1]
action=enable
service=chronyd

# start chronyd service
[service2]
action=restart
service=chronyd

# Setup glusterfs slice
[service3]
action=restart
service=glusterd
slice_setup=yes
```

```
# Open the required ports and firewalld services
[firewalld]
action=add
ports=111/tcp,2049/tcp,54321/tcp,5900/tcp,5900-
6923/tcp,5666/tcp,16514/tcp,54322/tcp
services=glusterfs

# Disable gluster hook scripts
[script2]
action=execute
file=/usr/share/ansible/gdeploy/scripts/disable-gluster-hooks.sh
```