Red Hat Hyperconverged Infrastructure for Virtualization 1.8

Upgrading Red Hat Hyperconverged Infrastructure for Virtualization

How to migrate to a Red Hat Enterprise Linux 8 based RHHI for Virtualization environment

Last Updated: 2022-06-02

Red Hat Hyperconverged Infrastructure for Virtualization 1.8 Upgrading Red Hat Hyperconverged Infrastructure for Virtualization

How to migrate to a Red Hat Enterprise Linux 8 based RHHI for Virtualization environment

Laura Bailey
lbailey@redhat.com

Legal Notice

Abstract

This document explains how to upgrade to the latest version of Red Hat Hyperconverged Infrastructure for Virtualization.

Table of Contents

MAKING OPEN SOURCE MORE INCLUSIVE

Red Hat is committed to replacing problematic language in our code, documentation, and web properties. We are beginning with these four terms: master, slave, blacklist, and whitelist. Because of the enormity of this endeavor, these changes will be implemented gradually over several upcoming releases. For more details, see our CTO Chris Wright's message .

CHAPTER 1. OVERVIEW OF UPGRADING RHHI FOR VIRTUALIZATION

Upgrading involves moving from one version of a product to a newer release of the same product. This section shows you how to upgrade to Red Hat Hyperconverged Infrastructure for Virtualization 1.8 from versions 1.5, 1.6 and 1.7.

From a component standpoint, this involves the following:

- Upgrading the Hosted Engine virtual machine to Red Hat Virtualization Manager version 4.4.
- Upgrading the physical hosts to Red Hat Virtualization 4.4.

CHAPTER 2. MAJOR CHANGES IN VERSION 1.8

Be aware of the following differences between Red Hat Hyperconverged Infrastructure for Virtualization 1.8 and previous versions:

Changed behavior

- Red Hat Hyperconverged Infrastructure for Virtualization 1.8 and Red Hat Virtualization 4.4 are based on Red Hat Enterprise Linux 8. Read about the key differences in Red Hat Enterprise Linux 8 in Considerations in adopting RHEL 8 .

- Cluster upgrades now require at least 10 percent free space on gluster disks in order to reduce the risk of running out of space mid-upgrade. This is available post the upgrade to Red Hat Hyperconverged Infrastructure for Virtualization 1.8. (BZ#1783750)

- “Hosts” and “Additional Hosts” tabs in the Web Console have been combined into a new "Hosts" tab that shows information previously shown on both. (BZ#1762804)

- Readcache and readcachesize options have been removed from VDO volumes, as they are not supported on Red Hat Enterprise Linux 8 based operating systems. (BZ#1808081)

- The Quartz scheduler is replaced with the standard Java scheduler to match support with Red Hat Virtualization. (BZ#1797487)

Enhancements

- The Administrator Portal can now upgrade all hosts in a cluster with one click. This is available post the upgrade to Red Hat Hyperconverged Infrastructure for Virtualization 1.8. (BZ#1721366)

- At-rest encryption using Network-Bound Disk Encryption (NBDE) is now supported on new Red Hat Hyperconverged Infrastructure for Virtualization deployments. (BZ#1821248, BZ#1781184)

- Added support for IPv6 networking. Environments with both IPv4 and IPv6 addresses are not supported. (BZ#1721383)

- New roles, playbooks, and inventory examples are available to simplify and automate the following tasks:

  - Upgrading (BZ#1500728, BZ#1832654)

  - Backing up and restoring configuration (BZ#1850488)

  - Replacing hosts (BZ#1840123)

  - Blacklisting multipath devices (BZ#1807808)

  - Creating the gluster logical network (BZ#1832966)

  - Deploying on IPv6 networks. (BZ#1688217)

- Added an option to select IPv4 or IPv6 based deployment in the web console. (BZ#1688798)

- **fsync** in the replication module now uses **eager-lock** functionality which improves the performance of small-block of size approximately equal to 4k write-heavy workloads by more than 50 percent on Red Hat Hyperconverged Infrastructure for Virtualization 1.8. (BZ#1836164)

- The web console now supports blacklisting multipath devices. (BZ#1814120)

- New fencing policies **skip_fencing_if_gluster_bricks_up** and **skip_fencing_if_gluster_quorum_not_met** are now added and enabled by default. (BZ#1775552)

- Red Hat Hyperconverged Infrastructure for Virtualization now ensures that the "performance.strict-o-direct" option in Red Hat Gluster Storage is enabled before creating a storage domain. (BZ#1807400)

- Red Hat Gluster Storage volume options can now be set for all volumes in the Administrator Portal by using "all" as the volume name. (BZ#1775586)

- Read-only fields are no longer included in the web console user interface, making the interface simpler and easier to read. (BZ#1814553)

CHAPTER 3. UPGRADING FROM RED HAT HYPERCONVERGED INFRASTRUCTURE FOR VIRTUALIZATION 1.5 AND 1.6

3.1. UPGRADE WORKFLOW

To upgrade to Red Hat Hyperconverged Infrastructure for Virtualization 1.8, the primary requirement is to upgrade to Red Hat Hyperconverged Infrastructure for Virtualization 1.7 with the latest version of Red Hat Virtualization 4.3. The upgrade process for RHHI for Virtualization versions 1.5, 1.6, and 1.7 to RHHI for Virtualization 1.8 is as follows:

RHHI for Virtualization 1.5 (based on RHV 4.2)

Perform the upgrade from 1.5 to 1.7 and then upgrade to RHHI for Virtualization 1.8.

RHHI for Virtualization 1.6 (based on RHV 4.3)

Perform the upgrade from 1.6 to 1.7 and then upgrade to RHHI for Virtualization 1.8.

RHHI for Virtualization 1.7 (based on RHV 4.3.8 or later)

Update the current set-up to latest Red Hat Virtualization 4.3, then upgrade to RHHI for Virtualization 1.8.

3.2. UPGRADING TO RED HAT HYPERCONVERGED INFRASTRUCTURE FOR VIRTUALIZATION 1.7

Follow Upgrading to RHHI for Virtualization 1.7 guide to upgrade from RHHI for Virtualization 1.5, 1.6 to 1.7 and to upgrade RHHI for Virtualization 1.7 to the latest Red Hat Virtualization 4.3.z version.

CHAPTER 4. UPGRADING TO RED HAT HYPERCONVERGED INFRASTRUCTURE FOR VIRTUALIZATION 1.8

4.1. UPGRADE WORKFLOW

The procedure to upgrade to Red Hat Hyperconverged Infrastructure for Virtualization (RHHI for Virtualization) 1.8 is not a direct upgrade from the previous versions of RHHI for Virtualization using ‘yum update’ as RHHI for Virtualization 1.7 uses Red Hat Enterprise Linux 7 platform, whereas the new version 1.8 uses the Red Hat Enterprise Linux 8 platform.

As this is not a direct upgrade, engine backup is performed along with gluster configurations and the nodes are reinstalled. The configurations are then restored with new gluster volume created for the hosted engine. Newly installed nodes are allowed to synchronize with other nodes in the cluster and the procedure is repeated across all the nodes one after the other.

The connected hosts and virtual machines can continue to work while the Manager is being upgraded.

4.2. PREREQUISITES

- Red Hat recommends minimizing workload on the virtual machines, this will help to shorten the upgrade window. If there are highly write intensive workloads, the time taken to sync data will be longer leading to a longer upgrade window.
- If there are scheduled geo-replication sessions on the storage domains, Red Hat recommends to remove these schedules to avoid overlapping with the upgrade window.
- If geo-replication is in progress, wait for the sync to complete to start the upgrade.
- All data center’s and clusters in the environment must have the cluster compatibility level set to version 4.3 before starting the procedure.

4.3. RESTRICTIONS

- If the previous version of RHHI for Virtualization environment did not have deduplication and compression enabled, this feature can not be enabled during upgrade to RHHI for Virtualization 1.8.
- Network-Bound Disk Encryption (NBDE) is supported only with new deployments of RHHI for Virtualization 1.8. This feature can not be enabled during upgrade.

4.4. PROCEDURE

This section describes the procedure to upgrade to RHHI for Virtualization 1.8 from RHHI for Virtualization 1.7.

IMPORTANT

The playbooks mentioned in this section are only available in RHHI for Virtualization 1.7 environment, make sure the RHHI for virtualization versions 1.5 and 1.6 are upgraded to the latest version of RHHI for Virtualization 1.7.

4.4.1. Creating a new gluster volume for Red Hat Virtualization 4.4 Hosted Engine deployment

Procedure

Create a new gluster volume for the new Red Hat Virtualization 4.4 Hosted Engine deployment with bricks for each host under the existing *engine* brick mount which is **/gluster_bricks/engine**

- Use the free space in the existing engine brick mount path **/gluster_bricks/engine** on each host to create the new replica 3 volume.

```
# gluster volume create newengine replica 3 host1:/gluster_bricks/engine/newengine
host2:/gluster_bricks/engine/newengine host3:/gluster_bricks/engine/newengine
# gluster volume set newengine group virt
# gluster volume set newengine storage.owner-uid 36
# gluster volume set newengine storage.owner-gid 36
# gluster volume set newengine cluster.granular-entry-heal on
# gluster volume set newengine performance.strict-o-direct on
# gluster volume set newengine network.remote-dio off
# gluster volume start newengine
```

Verify

- Status of the bricks can be verified with the following command:

```
# gluster volume status newengine
```

4.4.2. Backing up the Gluster configurations

Prerequisites

The **tasks/backup.yml** and **archive_config.yml** playbooks are available with the latest version of RHV 4.3.z at **/etc/ansible/roles/gluster.ansible/playbooks/hc-ansible-deployment**.

IMPORTANT

If *tasks/backup.yml* and *archive_config.yml* are not available at **/etc/ansible/roles/gluster.ansible/playbooks/hc-ansible-deployment**, you can create these playbooks from Understanding the **archive_config_inventory.yml** file.

Procedure

1. Edit **archive_config_inventory.yml** inventory file at /etc/ansible/roles/gluster.ansible/playbooks/hc-ansible-deployment/archive_config_inventory.yml

   Hosts

   Host FQDN of all the hosts in the cluster.

   Common Variables

   The default value is correct for the common variables **backup_dir**, **nbde_setup** and **upgrade**.

   ```
   all:
     hosts:
   ```

```
    host1.example.com:
    host2.example.com:
    host3.example.com:
  vars:
    backup_dir: /archive
    nbde_setup: false
    upgrade: true
```

2. Run the **archive_config.yml** playbook using your updated inventory file with the backupfiles tag.

   ```
   # cd /etc/ansible/roles/gluster.ansible/playbooks/hc-ansible-deployment

   # ansible-playbook -i archive_config_inventory.yml archive_config.yml --tags backupfiles
   ```

3. The backup configuration tar file is generated on all the hosts under **/root** with name **rhvh-node-<HOSTNAME>-backup.tar.gz**. Copy this backup configuration tar file from all the hosts to a different machine(backup host).

Verify

- Verify that the backup configuration files are generated on all hosts and are copied to the different machine(backup host).

4.4.3. Migrating the virtual machines

1. Click on **Compute** → **Hosts** → Select the first host.
2. Click on the hostname → Select **Virtual Machines** tab.
3. Select all **Virtual Machines** → Migrate.
4. Wait for all Virtual Machines to migrate to other hosts in the cluster.

4.4.4. Backing up the Hosted Engine configurations

1. Enable **Global Maintenance** for Hosted Engine. Run the following command on one of the active hosts in the cluster deployed with **hosted-engine --deploy**.

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

2. Log in to the Hosted Engine VM using SSH and stop the **ovirt-engine** service.

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

3. Run the following command in Hosted Engine VM to create a backup of the engine from the Hosted Engine VM.

   ```
   # engine-backup --mode=backup --scope=all --file=<backup-file.tar.gz> --log=<logfile>

   Example:
   # engine-backup --mode=backup --scope=all --file=engine-backup.tar.gz --log=backup.log
   Start of engine-backup with mode 'backup'
   ```

```
scope: all
archive file: engine-backup.tar.gz
log file: backup.log
Backing up:
Notifying engine
- Files
- Engine database 'engine'
- DWH database 'ovirt_engine_history'
Packing into file 'engine-backup.tar.gz'
Notifying engine
Done.
```

4. Copy the backup file from the Hosted Engine VM to a different machine (backup host).

```
# scp <backup-file.tar.gz> root@backup-host.example.com:/backup/
```

5. Shut down the Hosted Engine VM by running **poweroff** command from the Hosted Engine VM.

4.4.5. Checking self-heal status

1. Check for any pending self-heal on all the replica 3 volumes and wait for the heal to complete. Run the following command on one of the hosts.

```
# gluster volume heal <volume> info summary
```

2. Once confirmed there are no pending self-heals, stop the glusterfs brick process and unmount all the bricks on the first host(the current host you are working on) to maintain file system consistency. Run the following on the first host:

```
# pkill glusterfsd; pkill glusterfs
# systemctl stop glusterd
# umount /gluster_bricks/*
```

4.4.6. Reinstalling the first host with Red Hat Virtualization Host 4.4

1. Use the Installing Red Hat Virtualization Host guide to re-install the host with Red Hat Virtualization Host 4.4 ISO, formatting only the OS disk.

IMPORTANT

Make sure that the installation does not format the other disks, as bricks are created on top of these disks.

2. Subscribe to Red Hat Virtualization Host(RHVH) 4.4 repositories once the node is up post RHVH 4.4 installation or install the RHV 4.4 appliance downloaded from customer portal.

```
# yum install rhvm-appliance
```

See Configuring software repository access to subscribe to Red Hat Virtualization Host.

4.4.7. Copying the backup files to the newly installed host

- Copy the engine backup and host configuration tar files from the backup host to the newly installed host and untar the content.

```
# scp root@backuphost.example.com:/backupdir/engine-backup.tar.gz /root/
# scp root@backuphost.example.com:/backupdir/rhvh-node-host1.example.com-
backup.tar.gz /root/
```

4.4.8. Restoring gluster configuration files to the newly installed host

NOTE

Ensure to remove the existing LVM filter before restoring the backup and regenerate the LVM filter after restoration.

1. Remove the existing LVM filter, to allow using the existing Physical Volumes (PVs).

```
# sed -i /^filter/d /etc/lvm/lvm.conf
```

2. Extract the contents of gluster configuration files.

```
# mkdir /archive
# tar -xvf /root/rhvh-host-host1.example.com.tar.gz -C /archive/
```

3. Edit the **archive_config_inventory.yml** file to restore the configuration files. The **archive_config_inventory.yml** file is available at **/etc/ansible/roles/gluster.ansible/playbooks/hc-ansible-deployment/archive_config_inventory.yml**

```
all:
  hosts:
    host1.example.com:
  vars:
    backup_dir: /archive
    nbde_setup: false
    upgrade: true
```

IMPORTANT

Use only one host under the *hosts* section of the restoration playbook.

4. Execute the playbook to restore the configuration files.

```
# cd /etc/ansible/roles/gluster.ansible/playbooks/hc-ansible-deployment
# ansible-playbook -i archive_config_inventory.yml archive_config.yml --tags restorefiles
```

5. Regenerate new LVM filters for the newly identified PVs.

```
# vdsm-tool config-lvm-filter -y
```

4.4.9. Deploying hosted engine on the newly installed host

Deploy hosted engine with option **hosted-engine --deploy --restore-from-file=<engine-backup.tar.gz>** pointing to the backed-up archive from the engine.

The hosted engine can be deployed interactively using **hosted-engine --deploy** command, providing storage corresponding to newly created engine volume.

The hosted engine can also be deployed using **ovirt-ansible-hosted-engine-setup** role in an automated way and Red Hat recommends to use the automated way to avoid errors. The following procedure explains the automated way of deploying Hosted Engine VM:

1. Create the playbook for Hosted Engine deployment in the newly installed host at **/etc/ansible/roles/gluster.ansible/playbooks/hc-ansible-deployment/he.yml**

```
---
- name: Deploy oVirt hosted engine
  hosts: localhost
  roles:
    - role: ovirt.ovirt.hosted_engine_setup
```

2. Update the Hosted Engine related information using the **he_gluster_vars.json** template file at **/etc/ansible/roles/gluster.ansible/playbooks/hc-ansible-deployment/he_gluster_vars.json**.

```
# cat /etc/ansible/roles/gluster.ansible/playbooks/hc-ansible-deployment/he_gluster_vars.json

{
  "he_appliance_password": "password",
  "he_admin_password": "password",
  "he_domain_type": "glusterfs",
  "he_fqdn": "hostedengine.example.com",
  "he_vm_mac_addr": "00:18:15:20:59:01",
  "he_default_gateway": "19.70.12.254",
  "he_mgmt_network": "ovirtmgmt",
  "he_storage_domain_name": "HostedEngine",
  "he_storage_domain_path": "/newengine",
  "he_storage_domain_addr": "host1.example.com",
  "he_mount_options": "backup-volfile-servers=host2.example.com:host3.example.com",
  "he_bridge_if": "eth0",
  "he_enable_hc_gluster_service": true,
  "he_mem_size_MB": "16384",
  "he_cluster": "Default",
  "he_restore_from_file": "/root/engine-backup.tar.gz",
  "he_vcpus": "4"
  }
```

NOTE

In **he_gluster_vars.json file**, there are 2 important values

he_restore_from_file

This value is not given in template and should be added. This value should point to the absolute file name of engine backup archive copied to the local machine.

he_storage_domain_path

This value should refer to the newly created gluster volume.

The previous version of Red Hat Virtualization running on the Hosted Engine VM is down and discarded. MAC address and FQDN corresponding to the older hosted engine VM can be reused for the new engine as well.

3. For static Hosted Engine network configuration, add more options as:

   he_vm_ip_addr

   engine VM IP address

   he_vm_ip_prefix

   engine VM IP prefix

   he_dns_addr

   engine VM DNS server

   he_default_gateway

   engine VM default gateway

   NOTE

   If there are no specific DNS available, include 2 more options as **he_vm_etc_hosts: true** and **he_network_test: ping**.

4. Run the playbook to deploy the Hosted Engine:

```
# cd /etc/ansible/roles/gluster.ansible/playbooks/hc-ansible-deployment
# ansible-playbook he.yml --extra-vars='@he_gluster_vars.json'
```

   IMPORTANT

   If you are using Red Hat Virtualization Host (RHVH) 4.4 SP1 based on Red Hat Enterprise Linux 8.6 (RHEL 8.6), add the **-e 'ansible_python_interpreter=/usr/bin/python3.6'** parameter:

```
# ansible-playbook -e 'ansible_python_interpreter=/usr/bin/python3.6' he.yml --extra-vars='@he_gluster_vars.json'
```

5. Wait for the Hosted Engine deployment to complete.

   NOTE

   If there are any failures during Hosted Engine deployment, find the problem looking at the log messages under **/var/log/ovirt-hosted-engine-setup**, fix the problem. Clean the failed hosted engine deployment using the command **ovirt-hosted-engine-cleanup** and rerun the deployment.

6. Login into the RHV 4.4 Administration Portal on the newly installed RHV manager and ensure all the hosts are in **up** state. Wait for the self-heal on the gluster volumes to complete.

4.4.10. Upgrading the next host

1. Move to the next host(second host), ideally the next in order to *maintenance* mode from RHV Administration Portal and stop the *gluster service*.

    a. Click on **Compute** → **Hosts** → select the next host.

    b. Click on **Management** → Select **Maintenance** → **New Maintenance Host(s)** dialog box opens up.

    c. Select the check box **Stop Gluster** service → **OK**.

2. From the command line of the host unmount gluster bricks.

```
# umount /gluster_bricks/*
```

3. Reinstall this host with RHVH 4.4.

IMPORTANT

Ensure that the installation does not format the other disks, as bricks are created on these disks.

4. Copy the gluster configuration tar files from the backup host to the newly installed host and untar the content.

```
# scp root@backuphost.example.com:/backupdir/rhvh-node-<hostname>-backup.tar.gz /root/
# tar -xvf /root/rhvh-host-<hostname>-backup.tar.gz -C /archive/
```

5. Restore gluster configuration files on the newly installed host by executing the playbook mentioned in step Restoring the configuration files on the newly installed host on this host.

NOTE

Edit the **archive_config_inventory.yml** playbook and execute it on the newly installed host.

6. Reinstall the host in RHV Administration Portal.

    a. Copy the authorized key from the first deployed host in RHV 4.4.

    ```
    # scp root@host1.example.com:/root/.ssh/authorized_keys /root/.ssh/
    ```

    b. In the RHV Administration Portal, the host will be in *Maintenance* mode. Click on **Compute** → Select **Hosts** → Click on **Installation** → Select **Re-install**. New host dialog box will open, select the **Hosted Engine** tab and choose the hosted engine deployment action as **deploy**.

    c. Wait for the host to become **up**.

7. Repeat the steps in Upgrading next host for all the Red Hat Virtualization Host 4.3 hosts in the cluster.

4.4.11. Attaching gluster logical network

(optional)If a separate gluster logical network exists in the cluster, attach that gluster logical network to the required interface on each host.

1. Select **Compute** → **Hosts** → select **host** → Select tab **Network Interfaces**
2. Click on button **Setup Host Networks** → Drag and drop the gluster logical network to the appropriate network interface.

4.4.12. Removing old hosted engine storage domain

1. Identify the old hosted engine storage domain with name *hosted_storage* and no golden star next to it.
   a. Click on **Storage** → **Domains** → Select **hosted_storage** → **Data center** tab → **Maintenance**.
   b. Wait for that storage domain to move into *Maintenance*.
   c. Once the storage domain is in *Maintenance* click on **Detach**, the storage domain will go *unattached*.
   d. Select the unattached storage domain and click on **Remove** button → **OK**.
2. Stop and remove old engine volume.
   a. Click on **Storage** → **Volumes** → Select **old engine volume** → Click on **Stop** button → Confirm **OK**.
   b. Click on the same volume → **Remove** → Confirm **OK**.
3. Remove engine bricks on the hyperconverged hosts.

```
# rm -r /gluster_bricks/engine/engine
```

NOTE

Be cautious when removing the old engine brick as the new engine brick directory is also created on the same mount path as **/gluster_bricks/engine**.

4.4.13. Updating cluster compatibility

- Select **Compute** → **Clusters** → Select the cluster **Default** → **Edit** → update *Compatibility Version* to 4.6 → **OK**.

NOTE

There will be a warning for changing compatibility version as **VMs on the cluster to be restarted** click **OK**.

4.4.14. Updating data center compatibility

1. Select **Compute → Data Centers**.
2. Select the appropriate data center.
3. Click **Edit**.
4. The **Edit Data Center** dialog box opens.

5. Update *Compatibility Version* to **4.6** from the dropdown list.

4.4.15. Adding new gluster volume options available with RHV 4.4

New gluster volume options available with RHV 4.4, apply these volume options on all the volumes.

Execute the following on one of the nodes in the cluster.

```
# for vol in `gluster volume list`; do gluster volume set $vol group virt; done
```

4.4.16. Removing the archives and extracted content

Remove the archives and extracted contents of backup configuration files from all the nodes.

```
# rm -rf /root/rhvh-node-<hostname>-backup.tar.gz
# rm -rf /archive/
```

IMPORTANT

Disable the gluster volume option **cluster.lookup-optimize** on all the gluster volumes after the upgrade.

```
# for volume in `gluster volume list`; do gluster volume set $volume cluster.lookup-
optimize off; done
```

4.4.17. Troubleshooting

1. GFID mismatch leading to HA agents not syncing with each other.

   a. Appropriate **Input/Output error** is seen in **/var/log/ovirt-hosted-engine-ha/broker.log**

   ```
   # grep -i  error /var/log/ovirt-hosted-engine-ha/broker.log

   MainThread::ERROR::2020-07-13
   06:25:16,188::broker::69::ovirt_hosted_engine_ha.broker.broker.Broker::(run) Failed
   initializing the broker: [Errno 5] Input/output error: '/rhev/data-center/mnt/glusterSD/rhsqa-
   grafton10.lab.eng.blr.redhat.com:_newengine/1d94d115-8ddd-41c9-bd9c-
   477347e95ad4/ha_agent/hosted-engine.lockspace'
   ```

   b. Run the following command to check if there is any GFID mismatch on the volume.

   ```
   # grep -i ‘gfid mismatch’ /var/log/glusterfs/rhev*

   Example:
   # grep -i 'gfid mismatch' /var/log/glusterfs/rhev*

   /var/log/glusterfs/rhev-data-center-mnt-glusterSD-rhsqa-
   grafton10.lab.eng.blr.redhat.com:_newengine.log:[2020-07-13 06:14:12.992345] E
   [MSGID: 108008] [afr-self-heal-common.c:392:afr_gfid_split_brain_source] 0-newengine-
   replicate-0: Gfid mismatch detected for <gfid:580f8fe2-a42f-4f62-a5b0-
   7591c3740885>/hosted-engine.metadata>, d6a1fe1d-fc04-48cc-953f-d195d40749c1 on
   newengine-client-1 and c5e89641-e08f-462f-85ab-13518c21b7dc on newengine-client-0.
   ```

c. If there are entries listed with GFID mismatch, resolve the GFID split-brain.

```
# gluster volume heal <volume> split-brain latest-mtime <relative_path_of_file_in_brick>

Example:
# gluster volume heal newengine split-brain latest-mtime /1d94d115-8ddd-41c9-bd9c-
477347e95ad4/ha_agent/hosted-engine.lockspace
```

2. RHV Administration portal shows gluster volume in degraded state with one of the bricks on the upgraded node as **down**.

   a. Check the gluster volume status from the gluster command line on one of the hyperconverged hosts.The brick entry corresponding to the node which was upgraded and rebooted is listed with the brick process and port as **N/A**.
   In the following example, notice that there is no process ID or port information for host **rhvh2.example.com**:

```
# gluster volume status engine

Example:
Status of volume: engine
Gluster process                          TCP Port  RDMA Port
------------------------------------------------------------------------------
Brick rhvh1.example.com:/gluster_bricks/eng
ine/engine                               49158     0
Brick rhvh2.example.com:/gluster_bricks/eng
ine/engine                               N/A       N/A
Brick rhvh3.example.com:/gluster_bricks/eng
ine/engine                               49152     0
Self-heal Daemon on localhost                 N/A       N/A
Self-heal Daemon on rhvh2.example.com         N/A       N/A
Self-heal Daemon on rhvh3.example.com         N/A       N/A

Online  Pid
------------
Y       94365
Y       11052
Y       31153
Y       128608
Y       11838
Y       9806

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

   b. To fix this problem, kill the brick process and restart **glusterfsd** service.

```
 # pkill glusterfsd
 # systemctl restart glusterd
```

   c. Check the **gluster volume status** once again to make sure all the brick entries have got a brick process ID as well as the port information. Wait for a couple minutes for this information to reflect in the RHV administration portal.

```
# gluster volume status engine
```

4.5. VERIFYING THE UPGRADE

Verify that the upgrade has completed successfully.

1. Verify the RHV Manager version.
   - Login in to Administration Portal → **Help**(**?** symbol) on the top right → **About**.
     - The software version should be as **Software Version:4.4.X.X-X.X.el8ev**.

       ```
       Example: Software Version:4.4.1.8-0.7.el8ev
       ```

2. Verify the host version.
   - Run the following command on all the hosts to get the latest version of the host:

     ```
     # nodectl info | grep default
     ```

     ```
     Example:
     # nodectl info | grep default
     default: rhvh-4.4.1.1-0.20200707.0 (4.18.0-193.12.1.el8_2.x86_64)
     ```

CHAPTER 5. UPDATE BETWEEN MINOR RELEASES

To update the current version of Red Hat Hyperconverged Infrastructure for Virtualization 1.8 to the latest version, follow the steps in this section.

5.1. UPDATE WORKFLOW

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

1. Prepare the systems to be updated.
2. Updating the Hosted Engine virtual machine and Red Hat Virtualization Manager 4.4.
3. Update the hyperconverged hosts.

5.2. PREPARING THE SYSTEMS TO UPDATE

This section describes the steps to prepare the systems for the update procedure.

5.2.1. Update subscriptions

You can check which repositories a machine has access to by running the following command as the root user on the Hosted Engine Virtual Machine:

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

Verify that the Hosted Engine virtual machine is subscribed to the following repositories:

- rhel-8-for-x86_64-baseos-rpms
- rhel-8-for-x86_64-appstream-rpms
- rhv-4.4-manager-for-rhel-8-x86_64-rpms
- fast-datapath-for-rhel-8-x86_64-rpms
- jb-eap-7.4-for-rhel-8-x86_64-rpms
- openstack-16.2-cinderlib-for-rhel-8-x86_64-rpms
- rhceph-4-tools-for-rhel-8-x86_64-rpms

Verify that the Hyperconverged host (Red Hat Virtualization Node) is subscribed to the following repository:

- rhvh-4-for-rhel-8-x86_64-rpms

See Enabling the Red Hat Virtualization Manager Repositories for more information on subscribing to the above mentioned repositories.

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

Perform the following steps to check if geo-replication is in progress:

1. 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 starting the update process.

2. Remove all the scheduled geo-replication sessions so that synchronization will not occur during the update.

   a. Click **Storage** → **Domains** → Select the domain and click on the domain name.

   b. Click the **Remote Data Sync Setup** tab → **Setup** button.

   c. New dialog window to set the geo-replication schedule pops-up,set the recurrence to **None**.

5.3. UPDATING THE HOSTED ENGINE VIRTUAL MACHINE AND RED HAT VIRTUALIZATION MANAGER 4.4

This section describes the steps to update the Hosted Engine Virtual Machine and the Red Hat Virtualization Manager 4.4 to move towards updating the hyperconverged hosts.

5.3.1. Updating the Hosted Engine virtual machine

1. Place the cluster into Global Maintenance mode.

   a. Log in to the Web Console of one of the hyperconverged nodes.

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

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

2. On the Manager machine, check if updated packages are available. Log in to the Hosted Engine Virtual Machine and run the following command:

```
# engine-upgrade-check
```

5.3.2. Updating the Red Hat Virtualization Manager

1. Log in to the Hosted Engine virtual machine.

2. Upgrade the setup packages using the following command:

```
# yum update ovirt-engine\*setup\* rh\*vm-setup-plugins
```

3. Update the Red Hat Virtualization Manager with the engine-setup script. The engine-setup script performs the following tasks:

   - Prompts you with configuration questions.

   - Stops the ovirt-engine service.

   - Downloads and installs the updated packages.

   - Backs up and updates the database.

- Performs post-installation configuration,
- Starts the ovirt-engine service.

   a. Run the **engine-setup** script and follow the prompts to upgrade the Manager. This process can take a while and cannot be aborted, Red Hat recommends running it inside a **tmux** session.

   ```
   # engine-setup
   ```

   When the script completes successfully, the following message appears:

   Execution of setup completed successfully.

> IMPORTANT
>
> The update process might take some time. Do not stop the process before it completes.

4. Upgrade all other packages.

   ```
   # yum update
   ```

> IMPORTANT
>
> If any kernel packages are updated:
>
> 1. Disable global maintenance mode
> 2. Reboot the machine to complete the update.

5. Remove the cluster from Global Maintenance mode.

   a. Log in to the Web Console of one of the hyperconverged nodes

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

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

5.4. UPGRADING THE HYPERCONVERGED HOSTS

The upgrade process differs depending on whether your nodes use Red Hat Virtualization version 4.4.1 or version 4.4.2.

Use the following command to verify which version you are using:

```
# cat /etc/os-release | grep "PRETTY_NAME"
```

Then follow the appropriate process for your version:

- Upgrading from Red Hat Virtualization 4.4.2 and later
- Upgrading from Red Hat Virtualization 4.4.1 and earlier

5.4.1. Upgrading from Red Hat Virtualization 4.4.2 and later

1. Upgrade each hyperconverged host in the cluster, one at a time.
   For each hyperconverged host in the cluster:

   a. Upgrade the hyperconverged host.

      i. In the Manager, click **Compute → Hosts** and select a node.

      ii. Click **Installation → Upgrade**.

      iii. Click **OK** to confirm the upgrade.
      The node is upgraded and rebooted.

   b. Verify self-healing is complete.

      i. Click the name of the host.

      ii. Click the **Bricks** tab.

      iii. Verify that the **Self-Heal Info** column shows **OK** beside all bricks.

2. Update cluster compatibility settings to ensure you can use new features.

   a. Log in to the Administrator Portal.

   b. Click **Cluster** and select the cluster name (**Default**).

   c. Click **Edit**.

   d. Change **Cluster compatibility version** to **4.6**.

   > **IMPORTANT**
   >
   > Cluster compatibility is not completely updated until the virtual machines have been rebooted. Schedule a maintenance window and move any application virtual machines to maintenance mode before rebooting all virtual machines on each node.

   e. Click **Compute → Data Centers**.

   f. Click **Edit**.

   g. Change **Compatibility version** to **4.6**.

3. Update data center compatibility settings to ensure you can use new features.

   a. Select **Compute → Data Centers**.

   b. Select the appropriate data center.

   c. Click **Edit**.

   d. The **Edit Data Center** dialog box opens.

   e. Update *Compatibility Version* to **4.6** from the dropdown list.

5.4.2. Upgrading from Red Hat Virtualization 4.4.1 and earlier

1. In the Manager, click **Compute → Hosts** and select a node.
2. Click **Installation** → Check for Upgrade. This will trigger a background check on that host for the presence of host update.
3. Once the update is available, there will be a notification next to the host about the availability of host update.
4. Move the host to *maintenance mode*.
   a. On the RHV Administration Portal, navigate to **Hosts** → Select the host.
   b. Click on **Management → Maintenance → Maintenance Host** dialog box opens.
   c. On the **Maintenance Host** dialog box, check the **Stop Gluster service** box → click **OK**.
5. Once the host is in *maintenance mode*, click **Installation → Upgrade**.
   Upgrade Host dialog box opens, make sure to un-check **Reboot host after upgrade**.
6. Click **OK** to confirm the upgrade.
7. Wait for the upgrade to complete.
8. Remove the existing LVM filter on the upgraded host before rebooting by using the following command:

   ```
   # sed -i /^filter/d /etc/lvm/lvm.conf
   ```

9. Reboot the host.
10. Once the host is rebooted, regenerate the LVM filter:

    ```
    # vdsm-tool config-lvm-filter -y
    ```

11. 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 information column**of all bricks is listed as OK before upgrading the next host.
12. Repeat the above steps on the other hyperconverged hosts.
13. Update cluster compatibility settings to ensure you can use new features.
    a. Log in to the Administrator Portal.
    b. Click **Cluster** and select the cluster name (**Default**).
    c. Click **Edit**.
    d. Change **Cluster compatibility version** to **4.6**.

> IMPORTANT
>
> Cluster compatibility is not completely updated until the virtual machines have been rebooted. Schedule a maintenance window and move any application virtual machines to maintenance mode before rebooting all virtual machines on each node.

e. Click **Compute → Data Centers**.

f. Click **Edit**.

g. Change **Compatibility version** to **4.6**.

14. Update data center compatibility settings to ensure you can use new features.

    a. Select **Compute → Data Centers**.

    b. Select the appropriate data center.

    c. Click **Edit**.

    d. The **Edit Data Center** dialog box opens.

    e. Update *Compatibility Version* to **4.6** from the dropdown list.

> IMPORTANT
>
> Disable the gluster volume option **cluster.lookup-optimize** on all the gluster volumes after the update.
>
> ```
> # for volume in `gluster volume list`; do gluster volume set $volume cluster.lookup-optimize off; done
> ```

Troubleshooting

1. The self healing process should start automatically once each hyperconverged host comes up after a reboot. Check for self-heal status using the command:

   ```
   # gluster volume heal <volname> info summary
   ```

   If there are pending self-heal entries for a long time, check the following:

   a. Gluster network is up.

   ```
   # ip addr show <ethernet-interface>
   ```

   b. All brick processes in the volume are up.

   ```
   # gluster volume status <vol>
   ```

   If there are any brick processes reported to be down, restart the **glusterd** service on the node where the brick is reported to be down:

   ```
   # systemctl restart glusterd
   ```

2. If the Red Hat Virtualization node is unable to boot and drops in to *maintenance shell*, then one of the reasons is due to the unstable LVM filter rejecting some of the physical volumes (PVs).

    a. Log into the *maintenance shell* with the *root* password.

    b. Remove the existing LVM filter configuration:

    ```
    # sed -i /^filter/d /etc/lvm/lvm.conf
    ```

    c. Reboot the host.

    d. Once the node is up, regenerate the LVM filter:

    ```
    # vdsm-tool config-lvm-filter -y
    ```

PART I. REFERENCE MATERIAL

APPENDIX A. WORKING WITH FILES ENCRYPTED USING ANSIBLE VAULT

Red Hat recommends encrypting the contents of deployment and management files that contain passwords and other sensitive information. Ansible Vault is one method of encrypting these files. More information about Ansible Vault is available in the Ansible documentation.

A.1. ENCRYPTING FILES

You can create an encrypted file by using the **ansible-vault create** command, or encrypt an existing file by using the **ansible-vault encrypt** command.

When you create an encrypted file or encrypt an existing file, you are prompted to provide a password. This password is used to decrypt the file after encryption. You must provide this password whenever you work directly with information in this file or run a playbook that relies on the file's contents.

Creating an encrypted file

```
$ ansible-vault create variables.yml
New Vault password:
Confirm New Vault password:
```

The **ansible-vault create** command prompts for a password for the new file, then opens the new file in the default text editor (defined as **$EDITOR** in your shell environment) so that you can populate the file before saving it.

If you have already created a file and you want to encrypt it, use the **ansible-vault encrypt** command.

Encrypting an existing file

```
$ ansible-vault encrypt existing-variables.yml
New Vault password:
Confirm New Vault password:
Encryption successful
```

A.2. EDITING ENCRYPTED FILES

You can edit an encrypted file using the **ansible-vault edit** command and providing the Vault password for that file.

Editing an encrypted file

```
$ ansible-vault edit variables.yml
New Vault password:
Confirm New Vault password:
```

The **ansible-vault edit** command prompts for a password for the file, then opens the file in the default text editor (defined as **$EDITOR** in your shell environment) so that you can edit and save the file contents.

A.3. REKEYING ENCRYPTED FILES TO A NEW PASSWORD

You can change the password used to decrypt a file by using the **ansible-vault rekey** command.

```
$ ansible-vault rekey variables.yml
Vault password:
New Vault password:
Confirm New Vault password:
Rekey successful
```

The **ansible-vault rekey** command prompts for the current Vault password, and then prompts you to set and confirm a new Vault password.

APPENDIX B. UNDERSTANDING THE GLUSTER_VOLUME_INVENTORY.YML FILE

The **gluster_volume_inventory.yml** inventory file is an example file that you can use to create logical volume from the existing volume group if free space is available.

You can create this file at **/etc/ansible/roles/gluster.ansible/playbooks/hc-ansible-deployment/gluster_volume_inventory.yml**

B.1. CONFIGURATION PARAMETERS FOR CREATING LOGICAL VOLUMES

hosts

Backend network FQDN of the hyperconverged hosts. Mention the same set of host names under **cluster_nodes** section.

```
hosts:
 <host1-backend-FQDN>:
 <host2-backend-FQDN>:
 <host3-backend-FQDN>:
```

vgname

The name of the existing volume group (VG) on the host.

gluster_infra_disktype

Disk Aggregation Type. Values taken:

- RAID6
- RAID5
- JBOD

gluster_infra_diskcount

Number of data disks in the RAID set. For JBOD the value is 1.

gluster_infra_stripe_unit_size

RAID stripe size. Ignore this parameter for JBOD.

Example gluster_volume_inventory file

```
hc_nodes:
  hosts:
    host1-backend.example.com:
    host2-backend.example.com:
    host3-backend.example.com:

  # Common configurations
  vars:
    gluster_infra_volume_groups:
      - vgname: gluster_vg_sdb
        pvname: /dev/sdb
```

```
gluster_infra_mount_devices:
  - path: /gluster_bricks/newengine
    lvname: gluster_lv_newengine
    vgname: gluster_vg_sdb

gluster_infra_thick_lvs:
  - vgname: gluster_vg_sdb
    lvname: gluster_lv_newengine
    size: 100G

gluster_infra_disktype: RAID6
gluster_infra_diskcount: 10
gluster_infra_stripe_unit_size: 256
gluster_features_force_varlogsizecheck: false
gluster_set_selinux_labels: true

cluster_nodes:
   - host1-backend.example.com
   - host2-backend.example.com
   - host3-backend.example.com

gluster_features_hci_cluster: "{{ cluster_nodes }}"
gluster_features_hci_volumes:
  - volname: newengine
    brick: /gluster_bricks/newengine/newengine
    arbiter: 0
```

APPENDIX C. UNDERSTANDING THE ARCHIVE_CONFIG_INVENTORY.YML FILE

The **archive_config_inventory.yml** file is an example Ansible inventory file that you can use to backup and restore the configurations of Red Hat Hyperconverged Infrastructure for Virtualization cluster.

You can find this file at **/etc/ansible/roles/gluster.ansible/playbooks/hc-ansible-deployment/archive_config_inventory.yml** on any hyperconverged host.

There are 2 playbooks **archive_config.yml** and **backup.yml**. The **archive_config.yml** is a wrapper playbook, that in turn imports **tasks/backup.yml**.

C.1. CONFIGURATION PARAMETERS FOR BACKUP AND RESTORE IN ARCHIVE_CONFIG_INVENTORY.YML

hosts

The backend FQDN of each host in the cluster that you want to back up.

backup_dir

The directory in which to store backup files.

nbde_setup

Upgrade does not support setting of NBDE, set to false.

upgrade

Set to true.

For example:

```
all:
  hosts:
    host1:
    host2:
    host3:
  vars:
    backup_dir: /archive
    nbde_setup: false
    upgrade: true
```

C.2. CREATING THE ARCHIVE_CONFIG.YML PLAYBOOK FILE

Create the **archive_config.yml** playbook file only if it is **not available** at the location **/etc/ansible/roles/gluster.ansible/playbooks/hc-ansible-deployment**

Add the following content to **archive_config.yml** file:

```
---
- import_playbook: tasks/backup.yml
  tags: backupfiles
```

C.3. CREATING THE TASKS/BACKUP.YML PLAYBOOK FILE

Create the **tasks/backup.yml** playbook file only if it is **not available** at the location **/etc/ansible/roles/gluster.ansible/playbooks/hc-ansible-deployment**

Add the following content to the **backup.yml** file:

```
---
- hosts: all
  tasks:
  - name: Check if backup dir is already available
    stat:
      path: "{{ backup_dir }}"
    register: result

  - fail:
      msg: Backup directory "{{backup_dir}}" exists, remove it and retry
    when: result.stat.isdir is defined

  - name: Create temporary backup directory
    file:
      path: "{{ backup_dir }}"
      state: directory

  - name: Get the hostname
    shell: uname -n
    register: hostname

  - name: Add hostname details to archive
    shell: echo {{ hostname.stdout }} > {{ backup_dir }}/hostname

  - name: Dump the IP configuration details
    shell: ip addr show > {{ backup_dir }}/ipconfig

  - name: Dump the IPv4 routing information
    shell: ip route > {{ backup_dir }}/ip4route

  - name: Dump the IPv6 routing information
    shell: ip -6 route > {{ backup_dir }}/ip6route

  - name: Get the disk layout information
    shell: lsblk > {{ backup_dir }}/lsblk

  - name: Get the mount information for reference
    shell: df -Th > {{ backup_dir }}/mount

  - name: Check for VDO configuration
    stat:
      path: /etc/vdoconf.yml
    register: vdoconfstat

  - name: Copy VDO configuration, if available
    shell: cp -a /etc/vdoconf.yml "{{backup_dir}}"
    when: vdoconfstat.stat.isreg is defined

  - name: Backup fstab
    shell: cp -a /etc/fstab "{{backup_dir}}"
```

```
- name: Backup glusterd config directory
  shell: cp -a /var/lib/glusterd "{{backup_dir}}"

- name: Backup /etc/crypttab, if NBDE is enabled
  shell: cp -a /etc/crypttab "{{ backup_dir }}"
  when: nbde_setup is defined and nbde_setup

- name: Backup keyfiles used for LUKS decryption
  shell: cp -a /etc/sd*keyfile "{{ backup_dir }}"
  when: nbde_setup is defined and nbde_setup

- name: Check for the inventory file generated from cockpit
  stat:
    path: /etc/ansible/hc_wizard_inventory.yml
  register: inventory

- name: Copy the host inventory file generated from cockpit
  shell: cp /etc/ansible/hc_wizard_inventory.yml {{ backup_dir }}
  when: inventory.stat.isreg is defined

- name: Create a tar.gz with all the contents
  archive:
    path: "{{ backup_dir }}/*"
    dest: /root/rhvh-node-{{ hostname.stdout }}-backup.tar.gz
```

APPENDIX D. UNDERSTANDING THE HE_GLUSTER_VARS.JSON FILE

The **he_gluster_vars.json** file is an example Ansible variable file. The variables in this file need to be defined in order to deploy Red Hat Hyperconverged Infrastructure for Virtualization.

You can find an example file at **/etc/ansible/roles/gluster.ansible/playbooks/hc-ansible-deployment/he_gluster_vars.json** on any hyperconverged host.

Example he_gluster_vars.json file

```
{
  "he_appliance_password": "encrypt-password-using-ansible-vault",
  "he_admin_password": "UI-password-for-login",
  "he_domain_type": "glusterfs",
  "he_fqdn": "FQDN-for-Hosted-Engine",
  "he_vm_mac_addr": "Valid MAC address",
  "he_default_gateway": "Valid Gateway",
  "he_mgmt_network": "ovirtmgmt",
  "he_storage_domain_name": "HostedEngine",
  "he_storage_domain_path": "/engine",
  "he_storage_domain_addr": "host1-backend-network-FQDN",
  "he_mount_options": "backup-volfile-servers=host2-backend-network-FQDN:host3-backend-
network-FQDN",
  "he_bridge_if": "interface name for bridge creation",
  "he_enable_hc_gluster_service": true,
  "he_mem_size_MB": "16384",
  "he_cluster": "Default",
  "he_vcpus": "4"
}
```

Red Hat recommends encrypting this file. See Working with files encrypted using Ansible Vault for more information.

D.1. REQUIRED VARIABLES

he_appliance_password

The password for the hosted engine. For a production cluster, use an encrypted value created with Ansible Vault.

he_admin_password

The password for the **admin** account of the hosted engine. For a production cluster, use an encrypted value created with Ansible Vault.

he_domain_type

The type of storage domain. Set to **glusterfs**.

he_fqdn

The FQDN for the hosted engine virtual machine.

he_vm_mac_addr

The MAC address for the appropriate network device of the hosted engine virtual machine. You can skip this option for hosted deployment with static IP configuration as in such cases the MAC address for Hosted Engine is automatically generated.

he_default_gateway

The FQDN of the gateway to be used.

he_mgmt_network

The name of the management network. Set to **ovirtmgmt**.

he_storage_domain_name

The name of the storage domain to create for the hosted engine. Set to **HostedEngine**.

he_storage_domain_path

The path of the Gluster volume that provides the storage domain. Set to **/engine**.

he_storage_domain_addr

The back-end FQDN of the first host providing the **engine** domain.

he_mount_options

Specifies additional mount options.

> For a three node deployment with IPv4 configurations, set:

```
"he_mount_options":"backup-volfile-servers=host2-backend-network-FQDN:host3-backend-network-FQDN"
```

The **he_mount_option** is not required for IPv4 based single node deployment of Red Hat Hyperconverged Infrastructure for Virtualization.

For a three node deployment with IPv6 configurations, set:

```
"he_mount_options":"backup-volfile-servers=host2-backend-network-FQDN:host3-backend-network-FQDN",xlator-option='transport.address-family=inet6'"
```

For a single node deployment with IPv6 configurations, set:

```
"he_mount_options":"xlator-option='transport.address-family=inet6'"
```

he_bridge_if

The name of the interface to use for bridge creation.

he_enable_hc_gluster_service

Enables Gluster services. Set to **true**.

he_mem_size_MB

The amount of memory allocated to the hosted engine virtual machine in megabytes.

he_cluster

The name of the cluster in which the hyperconverged hosts are placed.

he_vcpus

The amount of CPUs used on the engine VM. By default 4 VCPUs are allocated for Hosted Engine Virtual Machine.

D.2. REQUIRED VARIABLES FOR STATIC NETWORK CONFIGURATIONS

DHCP configuration is used on the Hosted Engine VM by default. However, if you want to use static IP or FQDN, define the following variables:

he_vm_ip_addr

Static IP address for Hosted Engine VM (IPv4 or IPv6).

he_vm_ip_prefix

IP prefix for Hosted Engine VM (IPv4 or IPv6).

he_dns_addr

DNS server for Hosted Engine VM (IPv4 or IPv6).

he_default_gateway

Default gateway for Hosted Engine VM (IPv4 or IPv6).

he_vm_etc_hosts

Specifies Hosted Engine VM IP address and FQDN to **/etc/hosts** on the host, boolean value.

Example he_gluster_vars.json file with static Hosted Engine configuration

```
{
  "he_appliance_password": "mybadappliancepassword",
  "he_admin_password": "mybadadminpassword",
  "he_domain_type": "glusterfs",
  "he_fqdn": "engine.example.com",
  "he_vm_mac_addr": "00:01:02:03:04:05",
  "he_default_gateway": "gateway.example.com",
  "he_mgmt_network": "ovirtmgmt",
  "he_storage_domain_name": "HostedEngine",
  "he_storage_domain_path": "/engine",
  "he_storage_domain_addr": "host1-backend.example.com",
  "he_mount_options": "backup-volfile-servers=host2-backend.example.com:host3-
backend.example.com",
  "he_bridge_if": "interface name for bridge creation",
  "he_enable_hc_gluster_service": true,
  "he_mem_size_MB": "16384",
  "he_cluster": "Default",
  "he_vm_ip_addr": "10.70.34.43",
  "he_vm_ip_prefix": "24",
  "he_dns_addr": "10.70.34.6",
  "he_default_gateway": "10.70.34.255",
  "he_vm_etc_hosts": "false",
  "he_network_test": "ping"
}
```

NOTE

If DNS is not available, use **ping** for **he_network_test** instead of **dns**.

```
Example: "he_network_test": "ping"
```