Red Hat Hyperconverged Infrastructure for Virtualization 1.8

Deploying Red Hat Hyperconverged Infrastructure for Virtualization

Instructions for deploying Red Hat Hyperconverged Infrastructure for Virtualization

Last Updated: 2022-02-10

Red Hat Hyperconverged Infrastructure for Virtualization 1.8 Deploying Red Hat Hyperconverged Infrastructure for Virtualization

Instructions for deploying Red Hat Hyperconverged Infrastructure for Virtualization

Laura Bailey
lbailey@redhat.com

Legal Notice

Abstract

This document outlines how to deploy Red Hat Hyperconverged Infrastructure for Virtualization (RHHI for Virtualization) across three physical machines, using Red Hat Gluster Storage 3.5 and Red Hat Virtualization 4.4. This creates a discrete cluster for use in remote office branch office (ROBO) environments, where a remote office synchronizes data to a central data center on a regular basis, but can remain fully functional if connectivity to the central data center is lost.

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 .

PART I. PLAN

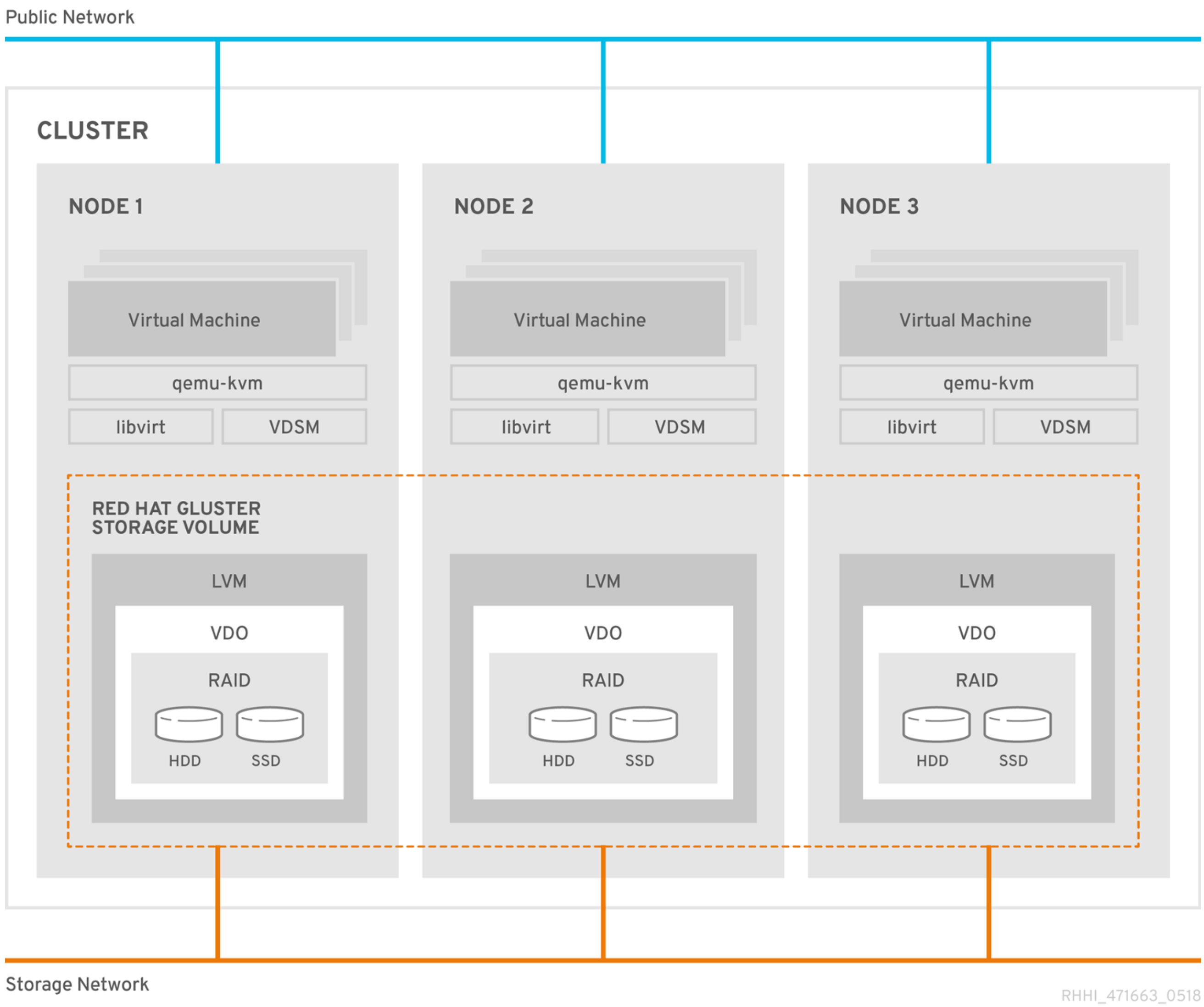
CHAPTER 1. ARCHITECTURE

Red Hat Hyperconverged Infrastructure for Virtualization (RHHI for Virtualization) combines compute, storage, networking, and management capabilities in one deployment.

RHHI for Virtualization is deployed across a number of physical machines to create a discrete cluster or *pod* using Red Hat Gluster Storage 3.5 and Red Hat Virtualization 4.4.

The dominant use case for this deployment is in remote office branch office (ROBO) environments, where a remote office synchronizes data to a central data center on a regular basis, but does not require connectivity to the central data center to function.

The following diagram shows the basic architecture of a single cluster, deployed across three physical machines.



1.1. UNDERSTANDING VDO

As of Red Hat Hyperconverged Infrastructure for Virtualization 1.6, you can configure a Virtual Data Optimizer (VDO) layer to provide data reduction and deduplication for your storage.

VDO is supported only when enabled on new installations at deployment time, and cannot be enabled on deployments upgraded from earlier versions of RHHI for Virtualization.

VDO performs following types of data reduction to reduce the space required by data.

Deduplication

Eliminates zero and duplicate data blocks. VDO finds duplicated data using the UDS (Universal Deduplication Service) Kernel Module. Instead of writing the duplicated data, VDO records it as a reference to the original block. The logical block address is mapped to the physical block address by VDO.

Compression

Reduces the size of the data by packing non-duplicate blocks together into fixed length (4 KB) blocks before writing to disk. This helps to speed up the performance for reading data from storage.

At best, data can be reduced to 15% of its original size.

Because reducing data has additional processing costs, enabling compression and deduplication reduces write performance. As a result, VDO is not recommended for performance sensitive workloads. Red Hat strongly recommends that you test and verify that your workload achieves the required level of performance with VDO enabled before deploying VDO in production, especially if you are using it in combination with other technology that reduces performance, such as disk encryption.

If you plan to use RAID hardware in the layer below VDO, Red Hat strongly recommends using SSD/NVMe disks to avoid performance issues. If there is no use of the RAID hardware layer below VDO, spinning disks can be used.

CHAPTER 2. SUPPORT REQUIREMENTS

Review this section to ensure that your planned deployment meets the requirements for support by Red Hat.

2.1. OPERATING SYSTEM

Red Hat Hyperconverged Infrastructure for Virtualization (RHHI for Virtualization) uses Red Hat Virtualization Host 4.4 as a base for all other configuration. Red Hat Enterprise Linux hosts are not supported.

See Requirements in the Red Hat Virtualization *Planning and Prerequisites Guide* for details on requirements of Red Hat Virtualization.

2.1.1. Browser requirements

Support for the web console and Red Hat Virtualization Administrator Portal varies based on the web browser you are using to access them.

Generally, use the most recent possible version of Mozilla Firefox, Google Chrome, or Microsoft Edge.

For details on browser support for the web console, see Logging in to the web console .

For details on browser support for the Administrator Portal, see Browser requirements for Red Hat Virtualization.

2.2. PHYSICAL MACHINES

Red Hat Hyperconverged Infrastructure for Virtualization (RHHI for Virtualization) requires **at least 3 physical machines**. Scaling to 6, 9, or 12 physical machines is also supported; see Scaling for more detailed requirements.

Each physical machine must have the following capabilities:

- at least 2 NICs (Network Interface Controllers) per physical machine, for separation of data and management traffic (see Section 2.5, "Networking" for details)
- for small deployments:
  - at least 12 cores
  - at least 64GB RAM
  - at most 48TB storage
- for medium deployments:
  - at least 12 cores
  - at least 128GB RAM
  - at most 64TB storage
- for large deployments:
  - at least 16 cores

- at least 256GB RAM
- at most 80TB storage

2.3. VIRTUAL MACHINES

The number of virtual machines that you are able to run on your hyperconverged deployment depends greatly on what those virtual machines do, and what load they are under. Test your workload's CPU, memory, and throughput requirements and provision your hyperconverged environment accordingly.

See Virtualization limits for Red Hat Virtualization for information about maximum numbers of virtual machines and virtual CPUs, and use the RHHI for Virtualization Sizing Tool for assistance planning your deployment.

NOTE

OpenShift Container Storage on top of Red Hat Hyperconverged Infrastructure for Virtualization (hyperconverged nodes that host virtual machines installed with Red Hat OpenShift Container Platform) is not a supported configuration.

2.4. HOSTED ENGINE VIRTUAL MACHINE

The Hosted Engine virtual machine requires at least the following:

- 1 dual core CPU (1 quad core or multiple dual core CPUs recommended)
- 4GB RAM that is not shared with other processes (16GB recommended)
- 25GB of local, writable disk space (50GB recommended)
- 1 NIC with at least 1Gbps bandwidth

For more information, see Requirements in the Red Hat Virtualization 4.4 *Planning and Prerequisites Guide*.

2.5. NETWORKING

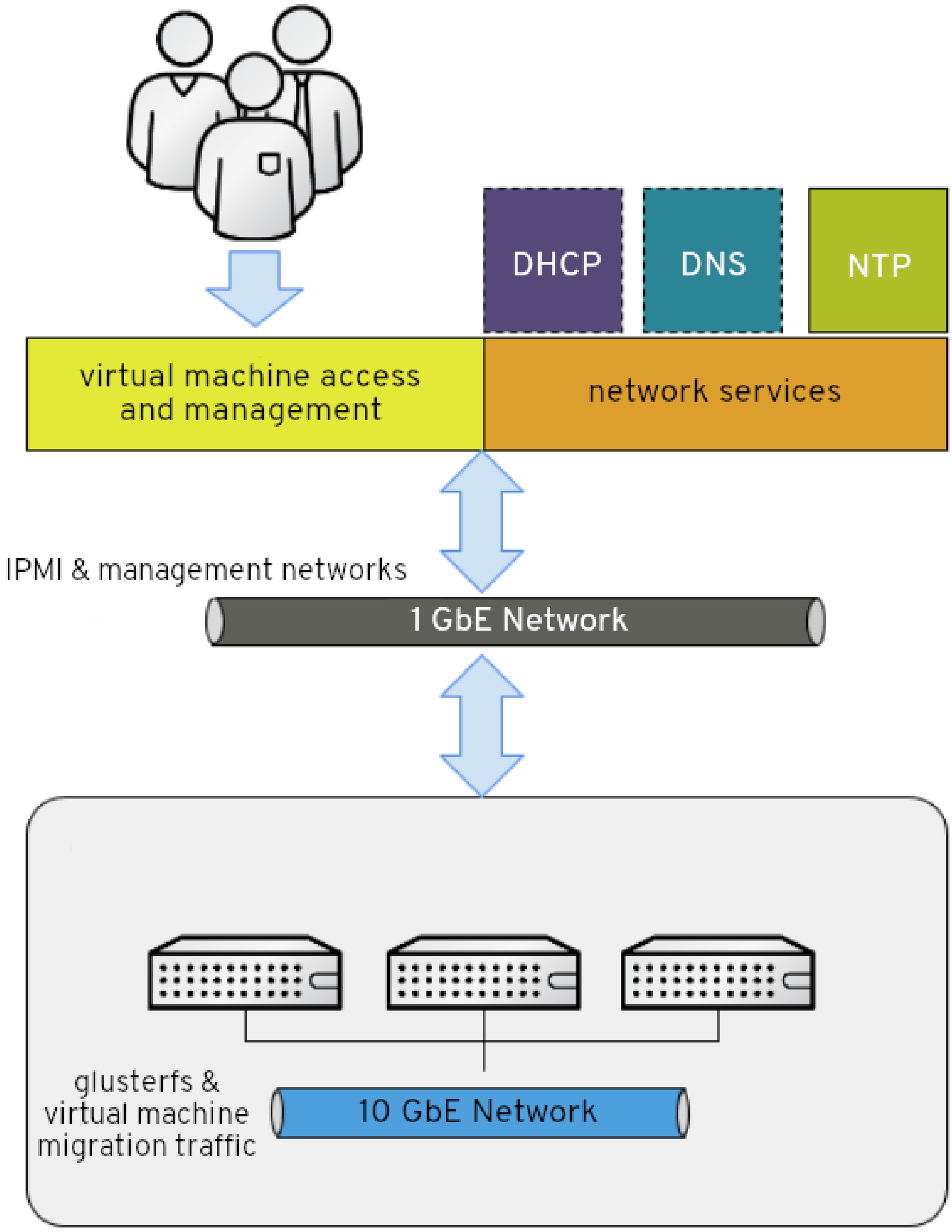
Fully-qualified domain names that are forward and reverse resolvable by DNS are required for all hyperconverged hosts and for the Hosted Engine virtual machine.

If external DNS is not available, for example in an isolated environment, ensure that the **/etc/hosts** file on each node contains the front and back end addresses of all hosts and the Hosted Engine node.

IPv6 is supported in IPv6-only environments (including DNS and gateway addresses). Environments with both IPv4 and IPv6 addresses are not supported.

Red Hat recommends usage of separate networks: a **front-end management network** for virtual machine traffic and a **back-end storage network** for gluster traffic and virtual machine migration.

Figure 2.1. Network diagram



Red Hat recommends each node to have two Ethernet ports, one for each network. This ensures optimal performance. For high availability, place each network on a separate network switch. For improved fault tolerance, provide a separate power supply for each switch.

Front-end management network

- Used by Red Hat Virtualization and virtual machines.
- Requires at least one 1Gbps Ethernet connection.

- IP addresses assigned to this network must be on the same subnet as each other, and on a different subnet to the back-end storage network.
- IP addresses on this network can be selected by the administrator.

Back-end storage network

- Used by storage and migration traffic between hyperconverged nodes.
- Requires at least one 10Gbps Ethernet connection.
- Requires maximum latency of 5 milliseconds between peers.

Network fencing devices that use Intelligent Platform Management Interfaces (IPMI) require a separate network.

If you want to use DHCP network configuration for the Hosted Engine virtual machine, then you must have a DHCP server configured prior to configuring Red Hat Hyperconverged Infrastructure for Virtualization.

If you want to configure disaster recoveryby using geo-replication to store copies of data, ensure that you configure a reliable time source.

Before you begin the deployment process, determine the following details:

- IP address for a gateway to the hyperconverged host. This address must respond to ping requests.
- IP address of the front-end management network.
- Fully-qualified domain name (FQDN) for the Hosted Engine virtual machine.
- MAC address that resolves to the static FQDN and IP address of the Hosted Engine.

2.6. STORAGE

A hyperconverged host stores configuration, logs and kernel dumps, and uses its storage as swap space. This section lists the minimum directory sizes for hyperconverged hosts. Red Hat recommends using the default allocations, which use more storage space than these minimums.

- / (root) - 6GB
- /**home** - 1GB
- /**tmp** - 1GB
- /**boot** - 1GB
- /**var** - 15GB
- /**var**/**crash** - 10GB
- /**var**/**log** - 8GB

IMPORTANT

Red Hat recommends increasing the size of **/var/log** to at least 15GB to provide sufficient space for the additional logging requirements of Red Hat Gluster Storage.

Follow the instructions in Growing a logical volume using the Web Console to increase the size of this partition after installing the operating system.

- **/var/log/audit** - 2GB
- **swap** - 1GB (see Recommended swap size for details)
- Anaconda reserves 20% of the thin pool size within the volume group for future metadata expansion. This is to prevent an out-of-the-box configuration from running out of space under normal usage conditions. Overprovisioning of thin pools during installation is also not supported.
- **Minimum Total - 64GB**

2.6.1. Disks

Red Hat recommends Solid State Disks (SSDs) for best performance. If you use Hard Drive Disks (HDDs), you should also configure a smaller, faster SSD as an LVM cache volume. The cache device must use the same block size as the other volumes.

Do not host the bricks of a Gluster volume across disks that have different block sizes. Ensure that you verify the block size of any VDO devices used to host bricks before creating a volume, as the default block size for a VDO device changed from 512 bytes in version 1.6 to 4 KB in version 1.7. Check the block size (in bytes) of a disk by running the following command:

```
# blockdev --getss <disk_path>
```

2.6.2. RAID

RAID5 and RAID6 configurations are supported. However, RAID configuration limits depend on the technology in use.

- SAS/SATA 7k disks are supported with RAID6 (at most 10+2)
- SAS 10k and 15k disks are supported with the following:
  - RAID5 (at most 7+1)
  - RAID6 (at most 10+2)

RAID cards must use flash backed write cache.

Red Hat further recommends providing at least one hot spare drive local to each server.

If you plan to use RAID hardware in the layer below VDO, Red Hat strongly recommends using SSD/NVMe disks to avoid performance issues. If there is no use of the RAID hardware layer below VDO, spinning disks can be used.

2.6.3. JBOD

As of Red Hat Hyperconverged Infrastructure for Virtualization 1.6, JBOD configurations are fully supported and no longer require architecture review.

2.6.4. Logical volumes

The logical volumes that comprise the **engine** gluster volume must be thick provisioned. This protects the Hosted Engine from out of space conditions, disruptive volume configuration changes, I/O overhead, and migration activity.

The logical volumes that comprise the **vmstore** and optional **data** gluster volumes must be thin provisioned. This allows greater flexibility in underlying volume configuration.

If your thin provisioned volumes are on Hard Drive Disks (HDDs), configure a smaller, faster Solid State Disk (SSD) as an lvmcache for improved performance. The cache device must use the same block size as the other volumes.

2.6.5. Red Hat Gluster Storage volumes

Red Hat Hyperconverged Infrastructure for Virtualization is expected to have 3–4 Red Hat Gluster Storage volumes.

- 1 **engine** volume for the Hosted Engine
- 1 **vmstore** volume for virtual machine operating system disk images
- 1 **data** volume for other virtual machine disk images
- 1 **shared_storage** volume for geo-replication metadata

Separate **vmstore** and **data** volumes are recommended to minimize backup storage requirements. Storing virtual machine data separate from operating system images means that only the **data** volume needs to be backed up when storage space is at a premium, since operating system images on the **vmstore** volume can be more easily rebuilt.

2.6.6. Volume types

Red Hat Hyperconverged Infrastructure for Virtualization (RHHI for Virtualization) supports only the following volume types at deployment time:

- Replicated volumes (3 copies of the same data on 3 bricks, across 3 nodes).
- Arbitrated replicated volumes (2 full copies of the same data on 2 bricks and 1 arbiter brick that contains metadata, across three nodes).
- Distributed volume with a single brick (1 copy of the data, no replication to other bricks).

> **NOTE**
>
> Distributed volume with a single brick is supported only for single node deployment of Red Hat Hyperconverged Infrastructure for Virtualization.

You can create **distributed replicate** or **distributed arbitrated replicate** volumes during the deployment of Red Hat Hyperconverged Infrastructure for Virtualization using Ansible playbooks as mentioned in the guide Automating RHHI for Virtualization deployment.

Note that arbiter bricks store only file names, structure, and metadata. This means that a three-way arbitrated replicated volume requires about 75% of the storage space that a three-way replicated volume would require to achieve the same level of consistency. However, because the arbiter brick stores only metadata, a three-way arbitrated replicated volume only provides the availability of a two-way replicated volume.

For more information on laying out arbitrated replicated volumes, see Creating multiple arbitrated replicated volumes across fewer total nodes in the Red Hat Gluster Storage *Administration Guide*.

2.7. DISK ENCRYPTION

Disk encryption is supported as of Red Hat Hyperconverged Infrastructure for Virtualization 1.8.

The supported method is Network-Bound Disk Encryption (NBDE), which uses a key server to provide decryption keys to encrypted clients at boot time, avoiding the need to enter the decryption password manually.

NBDE support requires at least 1 additional server (physical or virtual) to act as the NBDE key server. For fault tolerance, Red Hat recommends 2 NBDE key servers.

NBDE key servers must not be part of the Red Hat Hyperconverged Infrastructure for Virtualization cluster.

NBDE key servers can use either of the following operating systems:

- Red Hat Enterprise Linux 7.8 and higher
- Red Hat Enterprise Linux 8.2 and higher

Disk encryption generally involves a small reduction in performance. Test this configuration thoroughly before putting it into production to ensure that it meets the performance requirements of your use case, particularly if you are using disk encryption with other technology that creates a slight reduction in speed, such as deduplication and compression using Virtual Disk Optimization.

2.8. VIRTUAL DATA OPTIMIZER (VDO)

A Virtual Data Optimizer (VDO) layer is supported as of Red Hat Hyperconverged Infrastructure for Virtualization 1.6.

VDO support is limited to new deployments only; do not attempt to add a VDO layer to an existing deployment.

Be aware that the default block size for a VDO device changed from 512 bytes in version 1.6 to 4 KB in version 1.7. Do not host the bricks of a Gluster volume across disks that have different block sizes.

Because reducing data has additional processing costs, enabling compression and deduplication reduces write performance. As a result, VDO is not recommended for performance sensitive workloads. Red Hat strongly recommends that you test and verify that your workload achieves the required level of performance with VDO enabled before deploying VDO in production, especially if you are using it in combination with other technology that reduces performance, such as disk encryption.

2.9. SCALING

The number of nodes you can have in an initial deployment depends on your deployment method.

- When you use the web console, you can deploy either 1 or 3 hyperconverged nodes.
  In this case, you cannot create a volume that spans more than 3 nodes at creation time; you must create a 3-node volume first and then expand it across more nodes after deployment.

- When you use Ansible automation, you can deploy up to the maximum of 12 hyperconverged nodes, and span volumes across the required number of nodes at deployment time.

1 node deployments cannot be scaled.

Other deployments can be scaled from a minimum of 3 nodes to 6, 9, or 12 nodes.

You can scale your deployment by adding disks and expanding Gluster volumes. Add disks on new or existing nodes and use them to either create new Gluster volumes or expand existing Gluster volumes.

2.10. EXISTING RED HAT GLUSTER STORAGE CONFIGURATIONS

Red Hat Hyperconverged Infrastructure for Virtualization is supported only when deployed as specified in this document. Existing Red Hat Gluster Storage configurations cannot be used in a hyperconverged configuration. If you want to use an existing Red Hat Gluster Storage configuration, refer to the traditional configuration documented in Configuring Red Hat Virtualization with Red Hat Gluster Storage.

2.11. DISASTER RECOVERY

Red Hat strongly recommends configuring a disaster recovery solution. For details on configuring geo-replication as a disaster recovery solution, see *Maintaining Red Hat Hyperconverged Infrastructure for Virtualization*: https://access.redhat.com/documentation/en-us/red_hat_hyperconverged_infrastructure_for_virtualization/1.8/html/maintaining_red_hat_hyperconverg backup-recovery.

2.11.1. Prerequisites for geo-replication

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

Two different managers required

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

2.11.2. 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 machine

Generate and execute Ansible playbooks manually from a separate machine that acts as an Ansible controller node. This node must have the **ovirt-ansible-collection** package, which provides all required disaster recovery Ansible roles.

NOTE

The **ovirt-ansible-collection** package is installed with the Hosted Engine virtual machine by default. However, during a disaster that affects the primary site, this virtual machine may be down. It is safe to use a machine that is outside the primary site to run this playbook, but for testing purposes these playbooks can be triggered from the Hosted Engine virtual machine.

2.12. ADDITIONAL REQUIREMENTS FOR SINGLE NODE DEPLOYMENTS

Red Hat Hyperconverged Infrastructure for Virtualization is supported for deployment on a single node provided that all Support Requirements are met, with the following additions and exceptions.

A single node deployment requires a physical machine with:

- 1 Network Interface Controller
- at least 12 cores
- at least 64GB RAM

Single node deployments cannot be scaled, and are not highly available. This deployment type is lower cost, but removes the option of availability.

CHAPTER 3. RECOMMENDATIONS

The configuration described in this section is not required, but may improve the stability or performance of your deployment.

3.1. GENERAL RECOMMENDATIONS

- Take a full backup as soon as deployment is complete, and store the backup in a separate location. Take regular backups thereafter. See Configuring backup and recovery options for details.
- Avoid running any service that your deployment depends on as a virtual machine in the same RHHI for Virtualization environment. If you must run a required service in the same deployment, carefully plan your deployment to minimize the downtime of the virtual machine running the required service.
- Ensure that hyperconverged hosts have sufficient entropy. Failures can occur when the value in **/proc/sys/kernel/random/entropy_avail** is less than **200**. To increase entropy, install the **rng-tools** package and follow the steps in https://access.redhat.com/solutions/1395493.
- Document your environment so that everyone who works with it is aware of its current state and required procedures.

3.2. SECURITY RECOMMENDATIONS

- Do not disable any security features (such as HTTPS, SELinux, and the firewall) on the hosts or virtual machines.
- Register all hosts and Red Hat Enterprise Linux virtual machines to either the Red Hat Content Delivery Network or Red Hat Satellite in order to receive the latest security updates and errata.
- Create individual administrator accounts, instead of allowing many people to use the default admin account, for proper activity tracking.
- Limit access to the hosts and create separate logins. Do not create a single root login for everyone to use. See Managing user accounts in the web console in the Red Hat Enterprise Linux 8 documentation.
- Do not create untrusted users on hosts.
- Avoid installing additional packages such as analyzers, compilers, or other components that add unnecessary security risk.

3.3. HOST RECOMMENDATIONS

- Standardize the hosts in the same cluster. This includes having consistent hardware models and firmware versions. Mixing different server hardware within the same cluster can result in inconsistent performance from host to host.
- Configure fencing devices at deployment time. Fencing devices are required for high availability.
- Use separate hardware switches for fencing traffic. If monitoring and fencing go over the same switch, that switch becomes a single point of failure for high availability.

3.4. NETWORKING RECOMMENDATIONS

- Bond network interfaces, especially on production hosts. Bonding improves the overall availability of service, as well as network bandwidth. See Network Bonding in the Administration Guide.

- For optimal performance and simplified troubleshooting, use VLANs to separate different traffic types and make the best use of 10 GbE or 40 GbE networks.

- If the underlying switches support jumbo frames, set the MTU to the maximum size (for example, **9000**) that the underlying switches support. This setting enables optimal throughput, with higher bandwidth and reduced CPU usage, for most applications. The default MTU is determined by the minimum size supported by the underlying switches. If you have LLDP enabled, you can see the MTU supported by the peer of each host in the NIC's tool tip in the **Setup Host Networks** window.

- 1 GbE networks should only be used for management traffic. Use 10 GbE or 40 GbE for virtual machines and Ethernet-based storage.

- If additional physical interfaces are added to a host for storage use, uncheck **VM network** so that the VLAN is assigned directly to the physical interface.

3.4.1. Recommended practices for configuring host networks

If your network environment is complex, you may need to configure a host network manually before adding the host to Red Hat Virtualization Manager.

Red Hat recommends the following practices for configuring a host network:

- Configure the network with the Web Console. Alternatively, you can use nmtui or nmcli.

- If a network is not required for a self-hosted engine deployment or for adding a host to the Manager, configure the network in the Administration Portal after adding the host to the Manager. See Creating a New Logical Network in a Data Center or Cluster.

- Use the following naming conventions:

  - VLAN devices: **VLAN_NAME_TYPE_RAW_PLUS_VID_NO_PAD**

  - VLAN interfaces: **physical_device.VLAN_ID** (for example, **eth0.23**, **eth1.128**, **enp3s0.50**)

  - Bond interfaces: **bondnumber** (for example, **bond0**, **bond1**)

  - VLANs on bond interfaces: **bondnumber.VLAN_ID** (for example, **bond0.50**, **bond1.128**)

- Use network bonding. Networking teaming is not supported.

- Use recommended bonding modes:

  - For the bridged network used as the virtual machine logical network (**ovirtmgmt**), see Which bonding modes work when used with a bridge that virtual machine guests or containers connect to?.

  - For any other logical network, any supported bonding mode can be used.

- Red Hat Virtualization’s default bonding mode is **(Mode 4) Dynamic Link Aggregation**. If your switch does not support Link Aggregation Control Protocol (LACP), use **(Mode 1) Active-Backup**. See Bonding Modes for details.

- Configure a VLAN on a physical NIC as in the following example (although **nmcli** is used, you can use any tool):

```
# nmcli connection add type vlan con-name vlan50 ifname eth0.50 dev eth0 id 50
# nmcli con mod vlan50 +ipv4.dns 8.8.8.8 +ipv4.addresses 123.123.0.1/24 +ivp4.gateway
123.123.0.254
```

- Configure a VLAN on a bond as in the following example (although **nmcli** is used, you can use any tool):

```
# nmcli connection add type bond con-name bond0 ifname bond0 bond.options
"mode=active-backup,miimon=100" ipv4.method disabled ipv6.method ignore
# nmcli connection add type ethernet con-name eth0 ifname eth0 master bond0 slave-type
bond
# nmcli connection add type ethernet con-name eth1 ifname eth1 master bond0 slave-type
bond
# nmcli connection add type vlan con-name vlan50 ifname bond0.50 dev bond0 id 50
# nmcli con mod vlan50 +ipv4.dns 8.8.8.8 +ipv4.addresses 123.123.0.1/24 +ivp4.gateway
123.123.0.254
```

- Do not disable **firewalld**.

- Customize the firewall rules in the Administration Portal after adding the host to the Manager. See Configuring Host Firewall Rules.

3.5. SELF-HOSTED ENGINE RECOMMENDATIONS

- Create a separate data center and cluster for the Red Hat Virtualization Manager and other infrastructure-level services, if the environment is large enough to allow it. Although the Manager virtual machine can run on hosts in a regular cluster, separation from production virtual machines helps facilitate backup schedules, performance, availability, and security.

- A storage domain dedicated to the Manager virtual machine is created during self-hosted engine deployment. Do not use this storage domain for any other virtual machines.

- All self-hosted engine nodes should have an equal CPU family so that the Manager virtual machine can safely migrate between them. If you intend to have various families, begin the installation with the lowest one.

- If the Manager virtual machine shuts down or needs to be migrated, there must be enough memory on a self-hosted engine node for the Manager virtual machine to restart on or migrate to it.

PART II. DEPLOY

CHAPTER 4. DEPLOYMENT WORKFLOW

The workflow for deploying Red Hat Hyperconverged Infrastructure for Virtualization is as follows:

1. **Check requirements.**
   Verify that your planned deployment meets support requirements: Requirements, and fill in the installation checklist so that you can refer to it during the deployment process.

2. **Install operating systems.**

   a. Install an operating system on each physical machine that will act as a hyperconverged host: Installing hyperconverged hosts.

   b. (Optional) Install an operating system on each physical or virtual machine that will act as an Network-Bound Disk Encryption (NBDE) key server: Installing NBDE key servers.

3. **Configure authentication between hyperconverged hosts.**
   Configure key-based SSH authentication without a password to enable automated configuration of the hosts: Configure key-based SSH authentication.

4. **(Optional) Configure disk encryption.**

   a. Configure NBDE key servers.

   b. Configure hyperconverged hosts as NBDE clients.

5. **Configure the hyperconverged cluster:**

   a. Configure Red Hat Gluster Storage on hyperconverged hosts using the Web Console.

   b. Deploy the Hosted Engine virtual machine using the web console.

   c. Configure hyperconverged nodes using the RHV Administration Portal.

CHAPTER 5. INSTALLING OPERATING SYSTEMS

5.1. INSTALLING HYPERCONVERGED HOSTS

The supported operating system for hyperconverged hosts is the latest version of Red Hat Virtualization 4.

5.1.1. Installing a hyperconverged host with Red Hat Virtualization 4

5.1.1.1. Downloading the Red Hat Virtualization 4 operating system

1. Navigate to the Red Hat Customer Portal .
2. Click **Downloads** to get a list of product downloads.
3. Click **Red Hat Virtualization**.
4. Click **Download latest**
5. In the **Product Software** tab, click the **Download** button beside the latest Hypervisor Image, for example, **Hypervisor Image for RHV 4.4**.
6. When the file has downloaded, verify its SHA-256 checksum matches the one on the page.

   ```
   $ sha256sum image.iso
   ```

7. Use the downloaded image to create an installation media device.
   See Creating installation media in the Red Hat Enterprise Linux 8 documentation.

5.1.1.2. Installing the Red Hat Virtualization 4 operating system on hyperconverged hosts

Prerequisites

- Be aware that this operating system is only supported for hyperconverged hosts. Do not install an Network-Bound Disk Encryption (NBDE) key server with this operating system.
- Be aware of additional server requirements when enabling disk encryption on hyperconverged hosts. See Disk encryption requirements for details.

Procedure

1. Start the machine and boot from the prepared installation media.
2. From the boot menu, select **Install Red Hat Virtualization 4** and press **Enter**.
3. Select a language and click **Continue**.
4. Accept the default **Localization** options.
5. Click **Installation destination**.
   a. Deselect any disks you do not want to use as installation locations, for example, any disks that will be used for storage domains.

> **WARNING**
>
> Disks with a check mark will be formatted and all their data will be lost. If you are reinstalling this host, ensure that disks with data that you want to retain do not show a check mark.

b. Select the **Automatic partitioning** option.

c. (Optional) If you want to use disk encryption, select **Encrypt my data** and specify a password.

> **WARNING**
>
> Remember this password, as your machine will not boot without it.
>
> This password is used as the **rootpassphrase** for this host during Network-Bound Disk Encryption setup.

d. Click **Done**.

6. Click **Network and Host Name**.

a. Toggle the **Ethernet** switch to **ON**.

b. Select the network interface and click **Configure**

i. On the **General** tab, check the **Connect automatically with priority** checkbox.

ii. (Optional) To use IPv6 networking instead of IPv4, specify network details on the **IPv6 settings** tab.
For static network configurations, ensure that you provide the static IPv6 address, prefix, and gateway, as well as IPv6 DNS servers and additional search domains.

> **IMPORTANT**
>
> You must use either IPv4 or IPv6; mixed networks are not supported.

iii. Click **Save**.

c. Click **Done**.

7. (Optional) Configure Security policy.

8. Click **Begin installation**.

a. Set a root password.

> **WARNING**
>
> Red Hat recommends not creating additional users on hyperconverged hosts, as this can lead to exploitation of local security vulnerabilities.

b. Click **Reboot** to complete installation.

9. Increase the size of the **/var/log** partition.
   You need at least 15 GB of free space for Red Hat Gluster Storage logging requirements. Follow the instructions in Growing a logical volume using the Web Console to increase the size of this partition.

5.2. INSTALLING NETWORK-BOUND DISK ENCRYPTION KEY SERVERS

If you want to use Network-Bound Disk Encryption to encrypt the contents of your disks in Red Hat Hyperconverged Infrastructure for Virtualization, you need to install at least one key server.

The supported operating systems for Network-Bound Disk Encryption (NBDE) key servers are the latest versions of Red Hat Enterprise Linux 7 and 8.

5.2.1. Installing an NBDE key server with Red Hat Enterprise Linux 8

5.2.1.1. Downloading the Red Hat Enterprise Linux 8 operating system

1. Navigate to the Red Hat Customer Portal .
2. Click **Downloads** to get a list of product downloads.
3. Click **Red Hat Enterprise Linux 8**
4. In the **Product Software** tab, click **Download** beside the latest binary DVD image, for example, **Red Hat Enterprise Linux 8.2 Binary DVD**.
5. When the file has downloaded, verify its SHA-256 checksum matches the one on the page.

```
$ sha256sum image.iso
```

6. Use the image to create an installation media device.
   See Creating installation media in the Red Hat Enterprise Linux 8 documentation for details.

5.2.1.2. Installing the Red Hat Enterprise Linux 8 operating system on Network-Bound Disk Encryption key servers

Procedure

1. Start the machine and boot from the prepared installation media.
2. From the boot menu, select **Install Red Hat Enterprise Linux 8**and press **Enter**.

3. Select a language and click **Continue**.

4. Accept the default **Localization** and **Software** options.

5. Click **Installation destination**.

   a. Select the disk that you want to install the operating system on.

   > **WARNING**
   >
   > Disks with a check mark will be formatted and all their data will be lost. If you are reinstalling this host, ensure that disks with data that you want to retain do not show a check mark.

   b. (Optional) If you want to use disk encryption, select **Encrypt my data** and specify a password.

   > **WARNING**
   >
   > Remember this password, as your machine will not boot without it.

   c. Click **Done**.

6. Click **Network and Host Name**.

   a. Toggle the **Ethernet** switch to **ON**.

   b. Select the network interface and click **Configure**

      i. On the **General** tab, check the **Connect automatically with priority** checkbox.

      ii. (Optional) To use IPv6 networking instead of IPv4, specify network details on the **IPv6 settings** tab.
      For static network configurations, ensure that you provide the static IPv6 address, prefix, and gateway, as well as IPv6 DNS servers and additional search domains.

      > **IMPORTANT**
      >
      > You must use either IPv4 or IPv6; mixed networks are not supported.

      iii. Click **Save**.

   c. Click **Done**.

7. (Optional) Configure Security policy.

8. Click **Begin installation**.
    a. Set a root password.
    b. Click **Reboot** to complete installation.
9. From the **Initial Setup** window, accept the licensing agreement and register your system.

5.2.2. Installing an NBDE key server with Red Hat Enterprise Linux 7

5.2.2.1. Downloading the Red Hat Enterprise Linux 7 operating system

1. Navigate to the Red Hat Customer Portal .
2. Click **Downloads** to get a list of product downloads.
3. Click **Versions 7 and below**.
4. In the **Product Software** tab, click **Download** beside the latest binary DVD image, for example, **Red Hat Enterprise Linux 7.8 Binary DVD**.
5. When the file has downloaded, verify its SHA-256 checksum matches the one on the page.

```
$ sha256sum image.iso
```

6. Use the image to create an installation media device.
   See Creating installation media in the Red Hat Enterprise Linux 8 documentation for details.

5.2.2.2. Installing the Red Hat Enterprise Linux 7 operating system on Network-Bound Disk Encryption key servers

Prerequisites

- Be aware that this operating system is only supported for Network-Bound Disk Encryption (NBDE) key servers. Do not install a hyperconverged host with this operating system.

Procedure

1. Start the machine and boot from the prepared installation media.
2. From the boot menu, select **Install Red Hat Enterprise Linux 7** and press **Enter**.
3. Select a language and click **Continue**.
4. Click **Date & Time**.
    a. Select a time zone.
    b. Click **Done**.
5. Click **Keyboard**.
    a. Select a keyboard layout.
    b. Click **Done**.

6. Click **Installation destination**.
   a. Deselect any disks you do not want to use as an installation location.
   b. If you want to use disk encryption, select **Encrypt my data** and specify a password.

      > **WARNING**
      >
      > Remember this password, as your machine will not boot without it.

   c. Click **Done**.
7. Click **Network and Host Name**.
   a. Click **Configure... → General**.
   b. Check the **Automatically connect to this network when it is available** check box.
   c. Click **Done**.
8. Optionally, configure language support, security policy, and kdump.
9. Click **Begin installation**.
   a. Set a root password.
   b. Click **Reboot** to complete installation.
10. From the **Initial Setup** window, accept the licensing agreement and register your system.

CHAPTER 6. INSTALL ADDITIONAL SOFTWARE

You need to perform some additional configuration for access to software and updates.

- Ensure you have access to software updates: Configure software repository access using the web console.
- If your hyperconverged hosts use disk encryption, Install disk encryption software .

6.1. CONFIGURING SOFTWARE ACCESS

6.1.1. Configuring software repository access using the Web Console

Prerequisites

- This process is for hyperconverged hosts based on Red Hat Virtualization 4.

Procedure

1. **On each hyperconverged host:**
   a. Log in to the Web Console.
      Use the management FQDN and port 9090, for example, **https://server1.example.com:9090/**.
   b. Click **Subscriptions**.
   c. Click **Register System**.
      i. Enter your Customer Portal user name and password.
      ii. Click **Done**.
         The Red Hat Virtualization Host subscription is automatically attached to the system.
   d. Enable the Red Hat Virtualization 4 repository to allow later updates to the Red Hat Virtualization Host:

```
# subscription-manager repos \
--enable=rhvh-4-for-rhel-8-x86_64-rpms
```

2. **(Optional) If you use disk encryption, execute the following on each Network-Bound Disk Encryption (NBDE) key server:**
   a. Log in to the NBDE key server.
   b. Register the NBDE key server with Red Hat.

```
# subscription-manager register --username=username --password=password
```

   c. Attach the subscription pool:

```
# subscription-manager attach --pool=pool_id
```

   d. Enable the repositories required for disk encryption software:

i. For NBDE key servers based on Red Hat Enterprise Linux 8:

```
# subscription-manager repos \
--enable="rhel-8-for-x86_64-baseos-rpms" \
--enable="rhel-8-for-x86_64-appstream-rpms"
```

ii. For NBDE key servers based on Red Hat Enterprise Linux 7:

```
# subscription-manager repos --enable="rhel-7-server-rpms"
```

6.2. INSTALLING SOFTWARE

6.2.1. Installing disk encryption software

The Network-Bound Disk Encryption key server requires an additional package to support disk encryption.

Prerequisites

- Configuring software repository access using the web console .

Procedure

1. On each Network-Bound Disk Encryption (NBDE) key server, install the server-side packages.

```
# yum install tang -y
```

CHAPTER 7. MODIFYING FIREWALL RULES

7.1. MODIFYING FIREWALL RULES FOR DISK ENCRYPTION

On Network-Bound Disk Encryption (NBDE) key servers, you need to open ports so that encryption keys can be served.

Procedure

1. On each NBDE key server:
    a. Open ports required to serve encryption keys.

    NOTE

    The default port is **80/tcp**. To use a custom port, see Deploying a tang server with SELinux in enforcing mode in the Red Hat Enterprise Linux 8 documentation.

    ```
    # firewall-cmd --add-port=80/tcp
    # firewall-cmd --add-port=80/tcp --permanent
    ```

    b. Verify that the port appears in the output of the following command.

    ```
    # firewall-cmd --list-ports | grep '80/tcp'
    ```

CHAPTER 8. CONFIGURE PUBLIC KEY BASED SSH AUTHENTICATION WITHOUT A PASSWORD

Configure public key based SSH authentication without a password for the root user on the first hyperconverged host to all hosts, **including itself**. Do this for all storage and management interfaces, and for both IP addresses and FQDNs.

8.1. GENERATING SSH KEY PAIRS WITHOUT A PASSWORD

Generating a public/private key pair lets you use key-based SSH authentication. Generating a key pair that does not use a password makes it simpler to use Ansible to automate deployment and configuration processes.

Procedure

1. Log in to the first hyperconverged host as the root user.
2. Generate an SSH key that does not use a password.
    a. Start the key generation process.

    ```
    # ssh-keygen -t rsa
    Generating public/private rsa key pair.
    ```

    b. Enter a location for the key.
    The default location, shown in parentheses, is used if no other input is provided.

    ```
    Enter file in which to save the key (/home/username/.ssh/id_rsa): <location>/<keyname>
    ```

    c. Specify and confirm an empty passphrase by pressing **Enter** twice.

    ```
    Enter passphrase (empty for no passphrase):
    Enter same passphrase again:
    ```

    The private key is saved in **<location>/<keyname>**. The public key is saved in **<location>/<keyname>.pub**.

    ```
    Your identification has been saved in <location>/<keyname>.
    Your public key has been saved in <location>/<keyname>.pub.
    The key fingerprint is SHA256:8BhZageKrLXM99z5f/AM9aPo/KAUd8ZZFPcPFWqK6+M
    root@server1.example.com
    The key's randomart image is:
    +---[ECDSA 256]---+
    |      . .      +=|
    | . . . =      o.o|
    |  + . * .    o...|
    | = . . *  . + +..|
    |. + . . So o * ..|
    |   . o . .+ =  ..|
    |      o oo ..=. .|
    |        ooo...+  |
    |        .E++oo   |
    +----[SHA256]-----+
    ```

> **WARNING**
>
> **Your identification** in this output is your private key. Never share your private key. Possession of your private key allows someone else to impersonate you on any system that has your public key.

8.2. COPYING SSH KEYS

To access a host using your private key, that host needs a copy of your public key.

Prerequisites

- Generate a public/private key pair with no password.

Procedure

1. Log in to the first host as the root user.
2. Copy your public key to each host that you want to access, including the host on which you execute the command, using both the front-end and the back-end FQDNs.

```
# ssh-copy-id -i <location>/<keyname>.pub <user>@<hostname>
```

Enter the password for **<user>@<hostname>** when prompted.

> **WARNING**
>
> Make sure that you use the file that ends in **.pub**. Never share your private key. Possession of your private key allows someone else to impersonate you on any system that has your public key.

For example, if you are logged in as the root user on **server1.example.com**, you would run the following commands for a **three node** deployment:

```
# ssh-copy-id -i <location>/<keyname>.pub root@server1front.example.com
# ssh-copy-id -i <location>/<keyname>.pub root@server2front.example.com
# ssh-copy-id -i <location>/<keyname>.pub root@server3front.example.com
# ssh-copy-id -i <location>/<keyname>.pub root@server1back.example.com
# ssh-copy-id -i <location>/<keyname>.pub root@server2back.example.com
# ssh-copy-id -i <location>/<keyname>.pub root@server3back.example.com
```

CHAPTER 9. CONFIGURE DISK ENCRYPTION

9.1. CONFIGURING NETWORK-BOUND DISK ENCRYPTION KEY SERVERS

Prerequisites

- You must have installed a Network-Bound Disk Encryption key server (Installing Network-Bound Disk Encryption key servers).

Procedure

1. Start and enable the tangd service:
   Run the following command on each Network-Bound Disk Encryption (NBDE) key server.

   ```
   # systemctl enable tangd.socket --now
   ```

2. Verify that hyperconverged hosts have access to the key server.

   a. Log in to a hyperconverged host.

   b. Request a decryption key from the key server.

      ```
      # curl key-server.example.com/adv
      ```

      If you see output like the following, the key server is accessible and advertising keys correctly.


```
{"payload":"eyJrZXlzIjpbeyJhbGciOiJFQ01SIiwiY3J2IjoiUC01MjEiLCJrZXlfb3BzIjpbImRlcml
2ZUtleSJdLCJrdHkiOiJFQyIsIngiOiJBQ2ZjNVFwVmlhal9wNWcwUlE4VW52dmdNN1AyRT
Rqa21XUEpSM3VRUkFsVWp0eWlfZ0Y5WEV3WmU5TmhIdHhDaG53OXhMSkphajRieVk
1ZVFGNGxhcXQ2IiwieSI6IkFOMmhpcmNpU2tnWG5HV2VHeGN1Nzk3N3B3empCTzZjZ
Wt5TFJZdlh4SkNvb3BfNmdZdnR2bEpJUk4wS211Y1g3WHUwMlNVWlpqTVVxU3EtdGwy
eEQ1SGcifSx7ImFsZyI6IkVTNTEyIiwiY3J2IjoiUC01MjEiLCJrZXlfb3BzIjpbInZlcmlmeSJdLC
JrdHkiOiJFQyIsIngiOiJBQXlXeU8zTTFEWEdIaS1PZ04tRFhHU29yNl9BcUlJdzQ5OHhRTz
dMam1kMnJ5bDN2WUFXTUVyR1I2MVhKdzdvbEhxdEdDQnhqV0I4RzZZV09vLWRpTUx
wIiwieSI6IkFVWkNXUTAxd3lVMXlYR2R0SUMtOHJhVUVadWM5V3JyekFVbUlyQVF5VTR
sWDcxd1RUWTJEeDlMMzliQU9tVk5oRGstS2lQNFZfYUlsZDFqVl9zdHRuVGoifV19","prot
ected":"eyJhbGciOiJFUzUxMiIsImN0eSI6Imp3ay1zZXQranNvbiJ9","signature":"ARiMIYnCj
7-1C-
ZAQ_CKee676s_vYpi9J94WBibroou5MRsO6ZhRohqh_SCbW1jWWJr8btymTfQgBF_Rwz
VNCnllAXt_D5KSu8UDc4LnKU-egiV-
02b61aiWB0udiEfYkF66krIajzA9y5j7qTdZpWsBObYVvuoJvlRo_jpzXJv0qEMi"}
```


9.2. CONFIGURING HYPERCONVERGED HOSTS AS NETWORK-BOUND DISK ENCRYPTION CLIENTS

9.2.1. Defining disk encryption configuration details

1. Log in to the first hyperconverged host.

2. Change into the **hc-ansible-deployment** directory:

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

3. Make a copy of the **luks_tang_inventory.yml** file for future reference.

```
cp luks_tang_inventory.yml luks_tang_inventory.yml.backup
```

4. Define your configuration in the luks_tang_inventory.yml file.
   Use the example **luks_tang_inventory.yml** file to define the details of disk encryption on each host. A complete outline of this file is available in Understanding the luks_tang_inventory.yml file.

5. Encrypt the **luks_tang_inventory.yml** file and specify a password using **ansible-vault**.
   The required variables in **luks_tang_inventory.yml** include password values, so it is important to encrypt the file to protect the password values.

```
# ansible-vault encrypt luks_tang_inventory.yml
```

   Enter and confirm a new vault password when prompted.

9.2.2. Executing the disk encryption configuration playbook

Prerequisites

- Define configuration in the **luks_tang_inventory.yml** playbook: Section 9.2.1, "Defining disk encryption configuration details".
- Hyperconverged hosts must have encrypted boot disks.

Procedure

1. Log in to the first hyperconverged host.
2. Change into the hc-ansible-deployment directory.

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

3. Run the following command as the root user to start the configuration process.

```
# ansible-playbook -i luks_tang_inventory.yml tasks/luks_tang_setup.yml --
tags=blacklistdevices,luksencrypt,bindtang --ask-vault-pass
```

   Enter the vault password for this file when prompted to start disk encryption configuration.

Verify

- Reboot each host and verify that they are able to boot to a login prompt without requiring manual entry of the decryption passphrase.
- Note that the devices that use disk encryption have a path of **/dev/mapper/luks_sdX** when you continue with Red Hat Hyperconverged Infrastructure for Virtualization setup.

Troubleshooting

- The given boot device/**dev/sda2** is not encrypted.

  ```
  TASK [Check if root device is encrypted]
  fatal: [server1.example.com]: FAILED! => {"changed": false, "msg": "The given boot device
  /dev/sda2 is not encrypted."}
  ```

  Solution: Reinstall the hyperconverged hosts using the process outlined in Section 5.1, "Installing hyperconverged hosts", ensuring that you select Encrypt my data during the installation process and follow all directives related to disk encryption.

- The output has been hidden due to the fact that *no_log: true* was specified for this result.

  ```
  TASK [gluster.infra/roles/backend_setup : Encrypt devices using key file]
  failed: [host1.example.com] (item=None) => {"censored": "the output has been hidden due to
  the fact that no_log: true was specified for this result", "changed": true}
  ```

  This output has been censored in order to not expose a passphrase. If you see this output for the **Encrypt devices using key file** task, the device failed to encrypt. You may have provided the incorrect disk in the inventory file.

  Solution: Clean up the deployment attempt using Cleaning up Network-Bound Disk Encryption after a failed deployment. Then correct the disk names in the inventory file.

- Non-zero return code from Tang server

  ```
  TASK [gluster.infra/roles/backend_setup : Download the advertisement from tang server for
  IPv4] * failed: [host1.example.com] (item={url: http://tang-server.example.com}) =>
  {"ansible_index_var": "index", "ansible_loop_var": "item", "changed": true, "cmd": "curl -sfg
  \"http://tang-server.example.com/adv\" -o /etc/adv0.jws", "delta": "0:02:08.703711", "end":
  "2020-06-10 18:18:09.853701", "index": 0, "item": {"url": "http://tang-server.example.com"},
  "msg": "non-zero return code*", "rc": 7, "start": "2020-06-10 18:16:01.149990", "stderr": "",
  "stderr_lines": [], "stdout": "", "stdout_lines": []}
  ```

  This error indicates that the server cannot access the **url** provided, either because the FQDN provided is incorrect or because it cannot be found from the host.

  Solution: Correct the **url** value provided for the NBDE key server or ensure that the **url** value is accessible from the host. Then run the playbook again with the **bindtang** tag:

  ```
  # ansible-playbook -i luks_tang_inventory.yml tasks/luks_tang_setup.yml --ask-vault-pass --
  tags=bindtang
  ```

- For any other playbook failures, use the instructions in Cleaning up Network-Bound Disk Encryption after a failed deployment to clean up your deployment. Review the playbook and inventory files for incorrect values and test access to all servers before executing the configuration playbook again.

CHAPTER 10. CONFIGURE RED HAT GLUSTER STORAGE FOR HOSTED ENGINE USING THE WEB CONSOLE

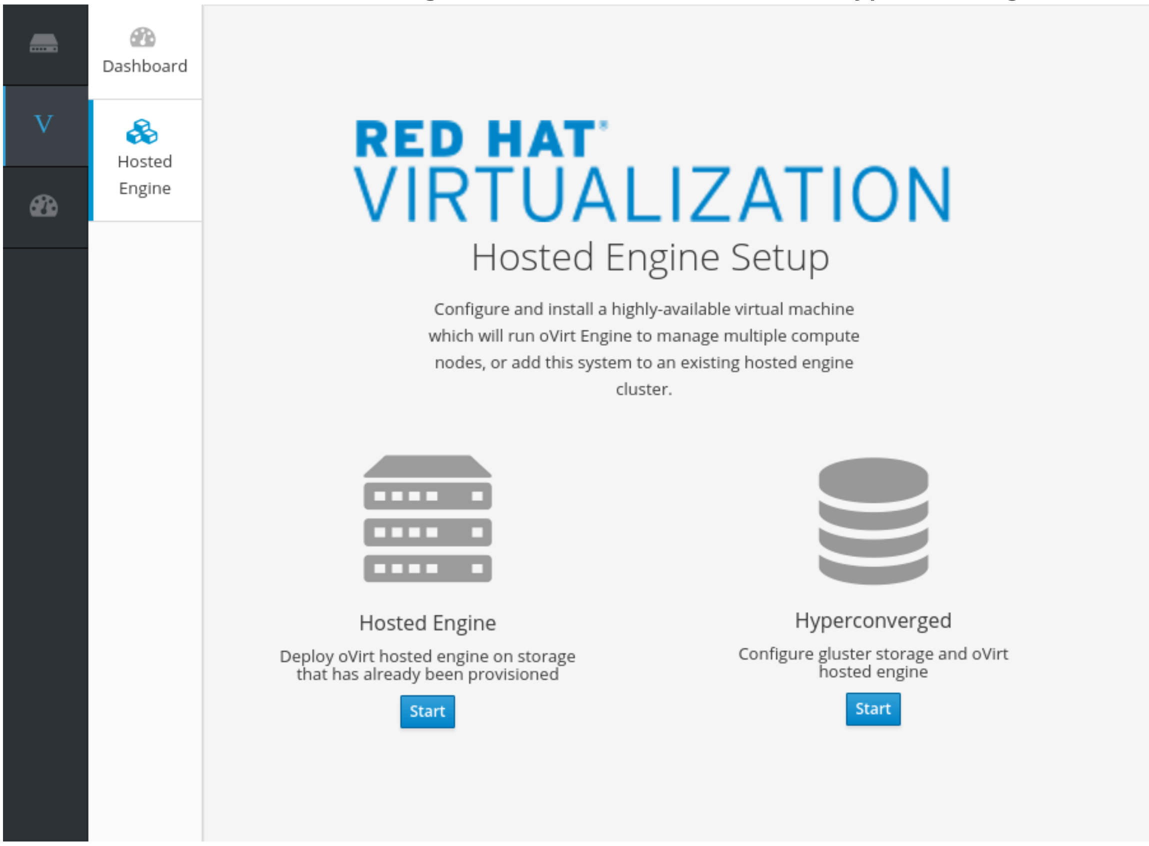
IMPORTANT

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

1. Log into the Web Console
   Browse to the Web Console management interface of the first hyperconverged host, for example, https://node1.example.com:9090/, and log in with the credentials you created in Section 5.1, "Installing hyperconverged hosts".

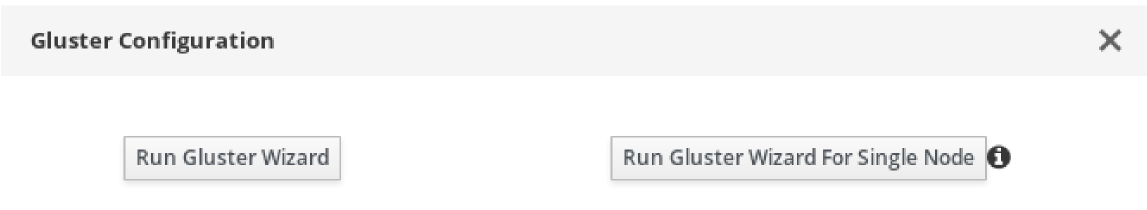
2. Start the deployment wizard

   a. Click Virtualization → Hosted Engine and click Start underneath Hyperconverged.



   The *Gluster Configuration* window opens.

   b. Click the Run Gluster Wizard button.



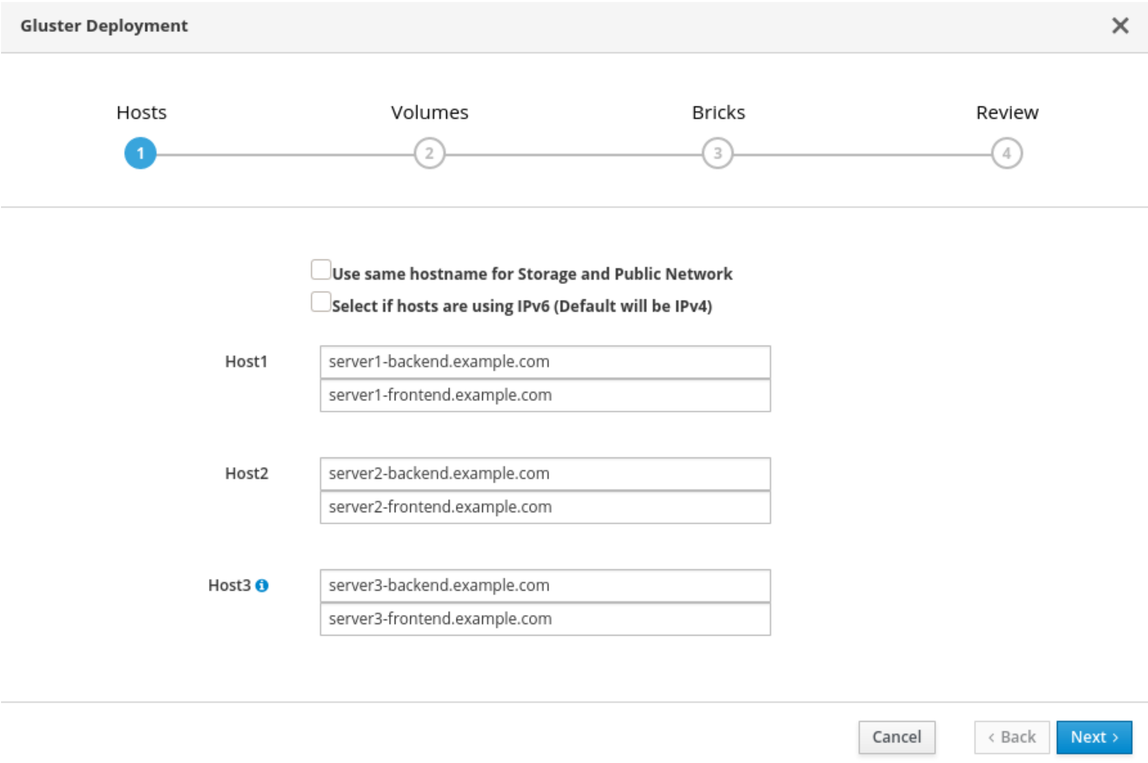
   The *Gluster Deployment* window opens in 3 node mode.

3. Specify hosts

a. If your hosts do not use multiple networks, check the **Use same hostname for storage and public network** checkbox.

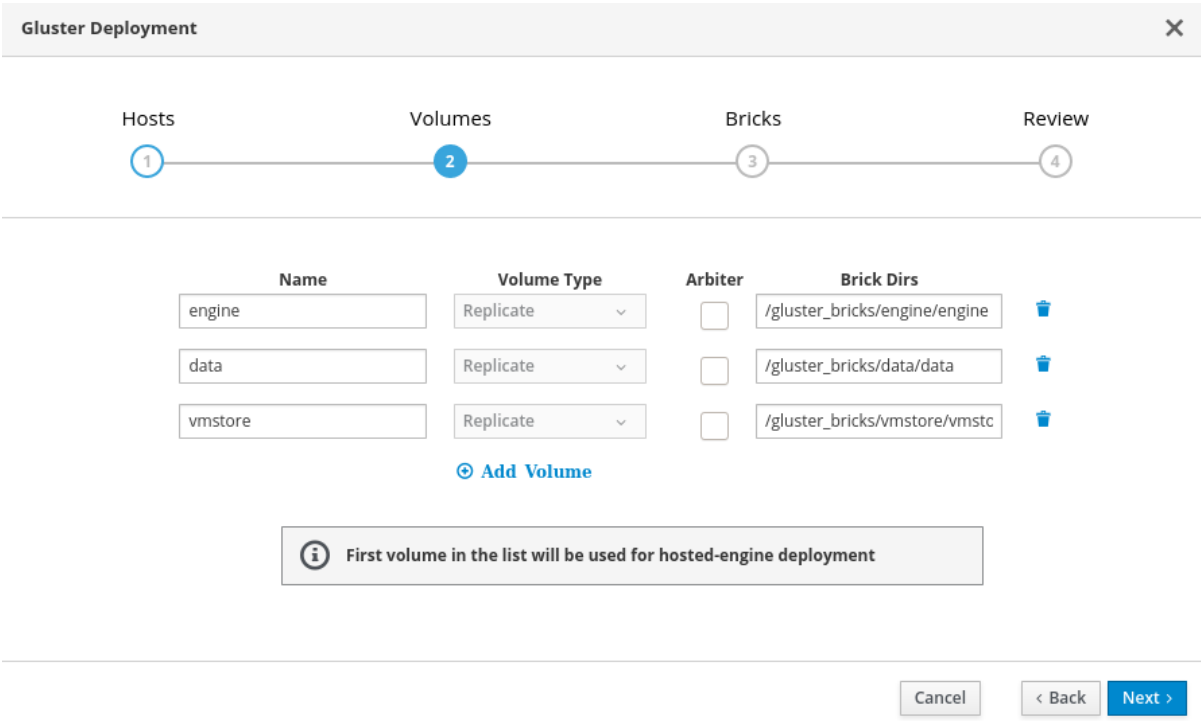
b. If your hosts use IPv6 networking, check the **Select if hosts are using IPv6** checkbox. Your hosts must use FQDNs if you select this option; IPv6 addresses are not supported.

c. Specify the back-end (storage network) and front-end (public network) FQDNs of the hyperconverged hosts.
In the **Host1** field, specify the FQDN of the hyperconverged host that can SSH to other hosts without a password using key-based authentication.



d. Click Next.

4. Specify volumes
Specify the volumes to create.



Name

Specify the name of the volume to be created.

Volume Type

Only replicated volumes are supported for three-node deployments.

Arbiter

Specify whether to create the volume with an arbiter brick. If this box is checked, the third disk stores only metadata.

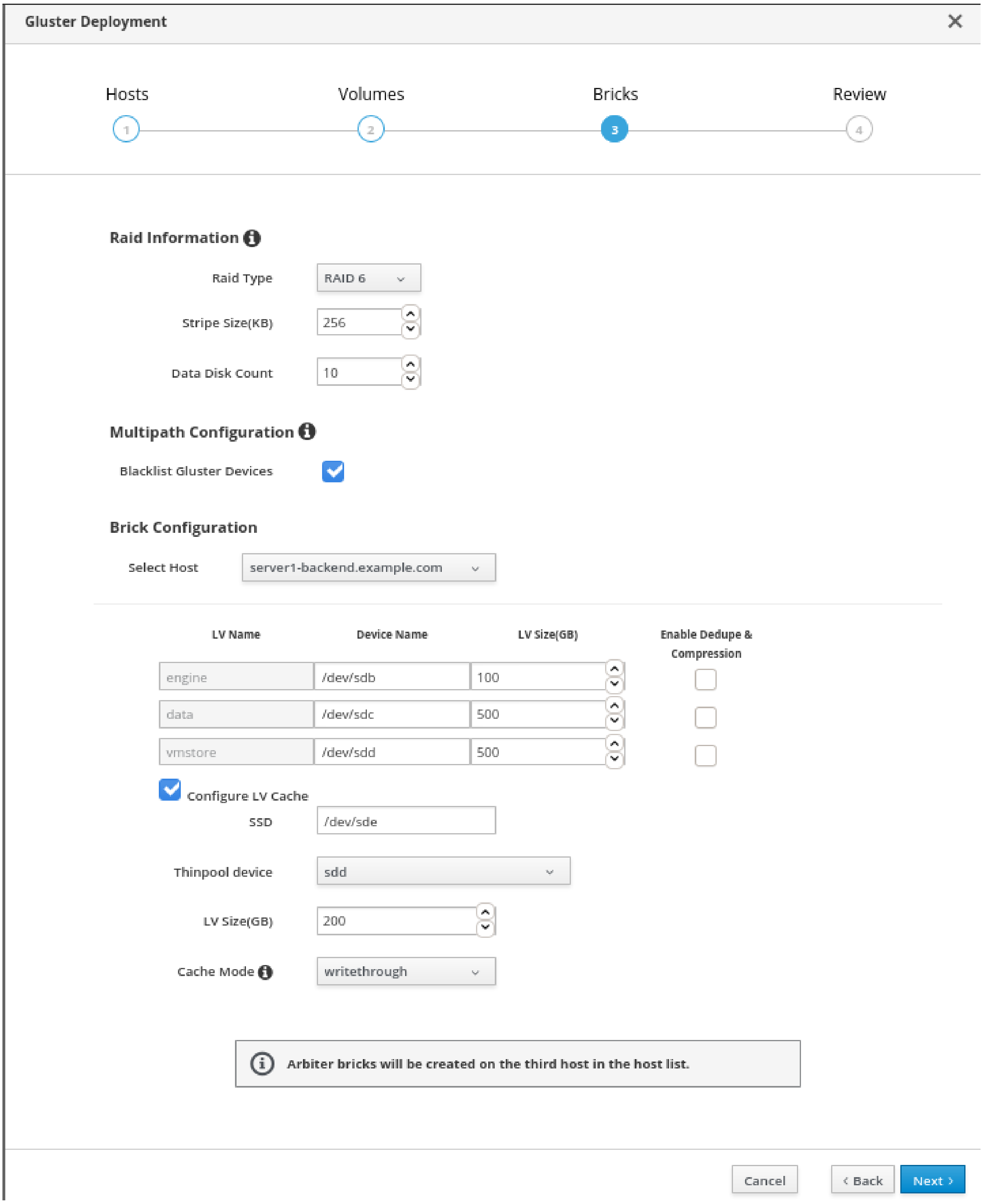
Brick Dirs

The directory that contains this volume's bricks. Use a brick path of the format **gluster_bricks/<volname>/<volname>**.

The default values are correct for most installations.

If you need more volumes, click Add Volumes to add another row and enter your extra volume details.

5. Specify bricks
   Enter details of the bricks to be created.



RAID Type

Specify the RAID configuration of the host. Supported values are raid5, raid6, and jbod. Setting this option ensures that your storage is correctly tuned for your RAID configuration.

Stripe Size

Specify the RAID stripe size in KB. This can be ignored for jbod configurations.

Data Disk Count

Specify the number of data disks in your host’s RAID volume. This can be ignored for jbod configurations.

Blacklist Gluster Devices

Prevents the disk that is specified as a Gluster brick from using a multipath device name. If you want to use a multipath device name, uncheck this checkbox and use the **/dev/mapper/<WWID>** format to specify your device in the **Device** field.

Select Host

If your hosts should use different device names or sizes, use this drop-down menu to change to the host you want to configure.

LV Name

The name of the logical volume to be created. This is pre-filled with the name that you specified on the previous page of the wizard.

Device Name

Specify the raw device you want to use in the format **/dev/sdc**. Use **/dev/mapper/<WWID>** format for multipath devices. Use **/dev/mapper/luks_<name>** format for devices using Network-Bound Disk Encryption.

LV Size

Specify the size of the logical volume to create in GiB. Do not enter units, only the number. This number should be the same for all bricks in a replicated set. Arbiter bricks can be smaller than other bricks in their replication set.

Recommendation for LV size

Logical volume for engine brick must be a thick LV of size 100GB, other bricks created as thin LV reserving 16GB for thinpool metadata and 16GB reserved for spare metadata.

Example:

```
If the host has a disk of size 1TB, then
engine brick size = 100GB ( thick LV )
Pool metadata size = 16GB
Spare metadata size = 16GB
Available space for thinpool = 1TB - ( 100GB + 16GB + 16GB ) = 868 GB
```

Other bricks for volumes can be created with the available thinpool storage space of 868GB, for example, *vmstore* brick with 200GB and *data* brick with 668GB.

Enable Dedupe & Compression

Specify whether to provision the volume using VDO for compression and deduplication at deployment time. The logical size of the brick is expanded to 10 times the size of physical volume as part of VDO space savings.

> NOTE
>
> Ensure to enable **Dedupe & Compression** on all the bricks which are part of the volume.

Configure LV Cache

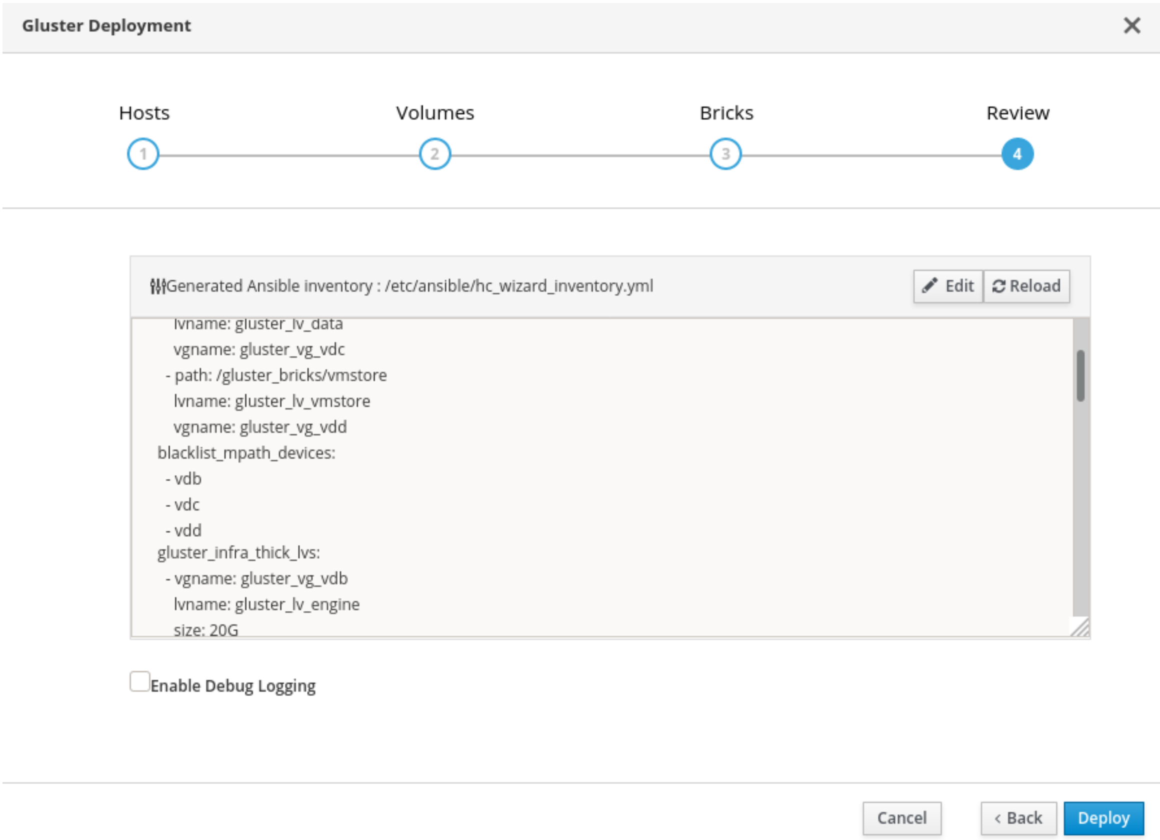
Optionally, check this checkbox to configure a small, fast SSD device as a logical volume cache for a larger, slower logical volume.

- Add the device path to the SSD field.
- Specify the Thinpool device to attach the cache device to.
- Add the size to the LV Size (GB) field.

- Set the Cache Mode used by the device.

> **WARNING**
>
> To avoid data loss when using write-back mode, Red Hat recommends using two separate SSD/NVMe devices. Configuring the two devices in a RAID-1 configuration (via software or hardware), significantly reduces the potential of data loss from lost writes.
>
> For further information about lvmcache configuration, see LVM cache logical volumes in the Red Hat Enterprise Linux 8 documentation.

1. Review and edit configuration

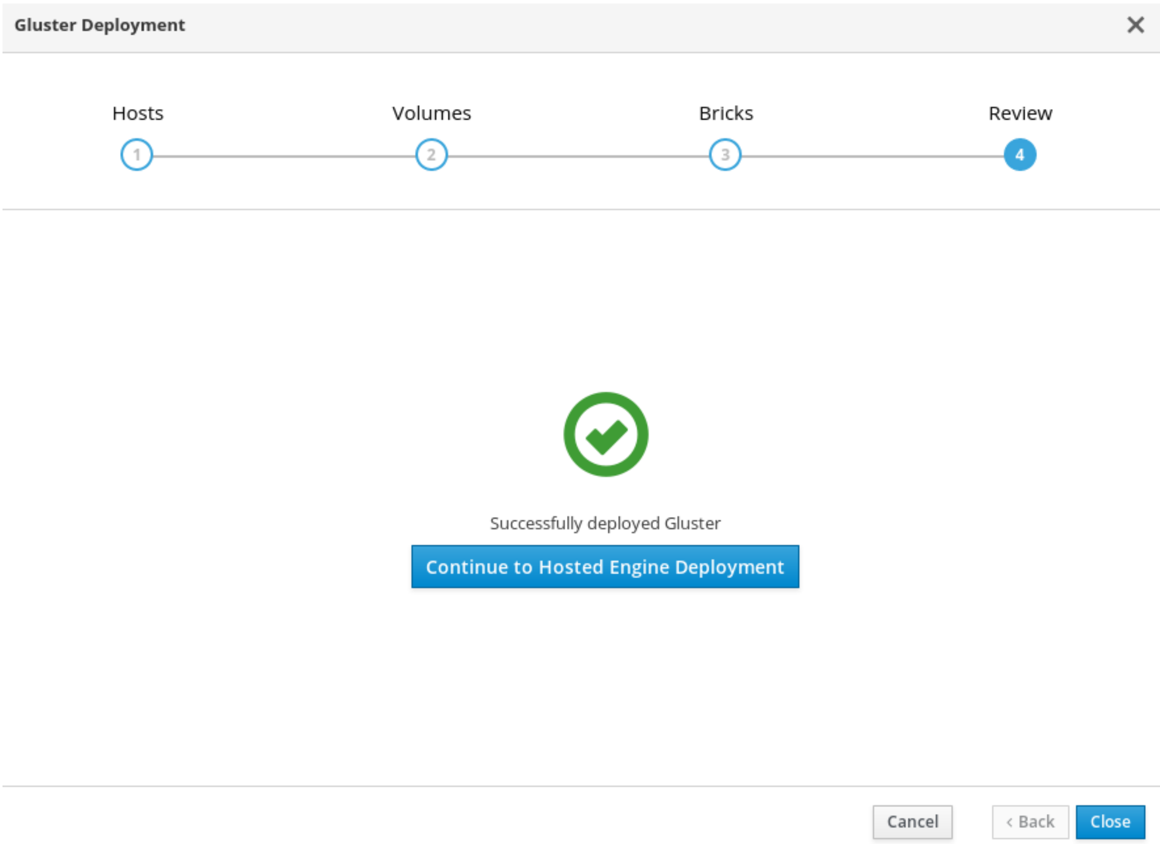


   a. Click Edit to begin editing the generated deployment configuration file.
      Make any changes required and click Save.

   b. Review the configuration file.
      If all configuration details are correct, click Deploy.

2. Wait for deployment to complete
   You can watch the progress of the deployment in the text field.

   The window displays Successfully deployed gluster when complete.



Click Continue to Hosted Engine Deployment and continue the deployment process with the instructions in Chapter 11, *Deploy the Hosted Engine using the Web Console*.

IMPORTANT

If deployment fails, click Clean up to remove any potentially incorrect changes to the system. If your deployment uses Network-Bound Disk Encryption, you must then follow the process in Cleaning up Network-Bound Disk Encryption after a failed deployment.

When cleanup is complete, click Redeploy. This returns you to the *Review and edit configuration* tab so that you can correct any issues in the generated configuration file before reattempting deployment.

CHAPTER 11. DEPLOY THE HOSTED ENGINE USING THE WEB CONSOLE

This section shows you how to deploy the Hosted Engine using the Web Console. Following this process results in Red Hat Virtualization Manager running as a virtual machine on the first physical machine in your deployment. It also configures a Default cluster comprised of the three physical machines, and enables Red Hat Gluster Storage functionality and the virtual-host *tuned* performance profile for each machine in the cluster.

Prerequisites

- The RHV-M Appliance is installed during the deployment process; however, if required, you can install it on the deployment host before starting the installation:

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

Manually installing the Manager virtual machine is not supported.

- Configure Red Hat Gluster Storage for Hosted Engine using the Web Console
- Gather the information you need for Hosted Engine deployment
  Have the following information ready before you start the deployment process.
  - IP address for a pingable gateway to the hyperconverged host
  - IP address of the front-end management network
  - Fully-qualified domain name (FQDN) for the Hosted Engine virtual machine
  - MAC address that resolves to the static FQDN and IP address of the Hosted Engine

Procedure

1. Open the Hosted Engine Deployment wizard
   If you continued directly from the end of Configure Red Hat Gluster Storage for Hosted Engine using the Web Console, the wizard is already open.

   Otherwise:

   a. Click Virtualization → Hosted Engine.

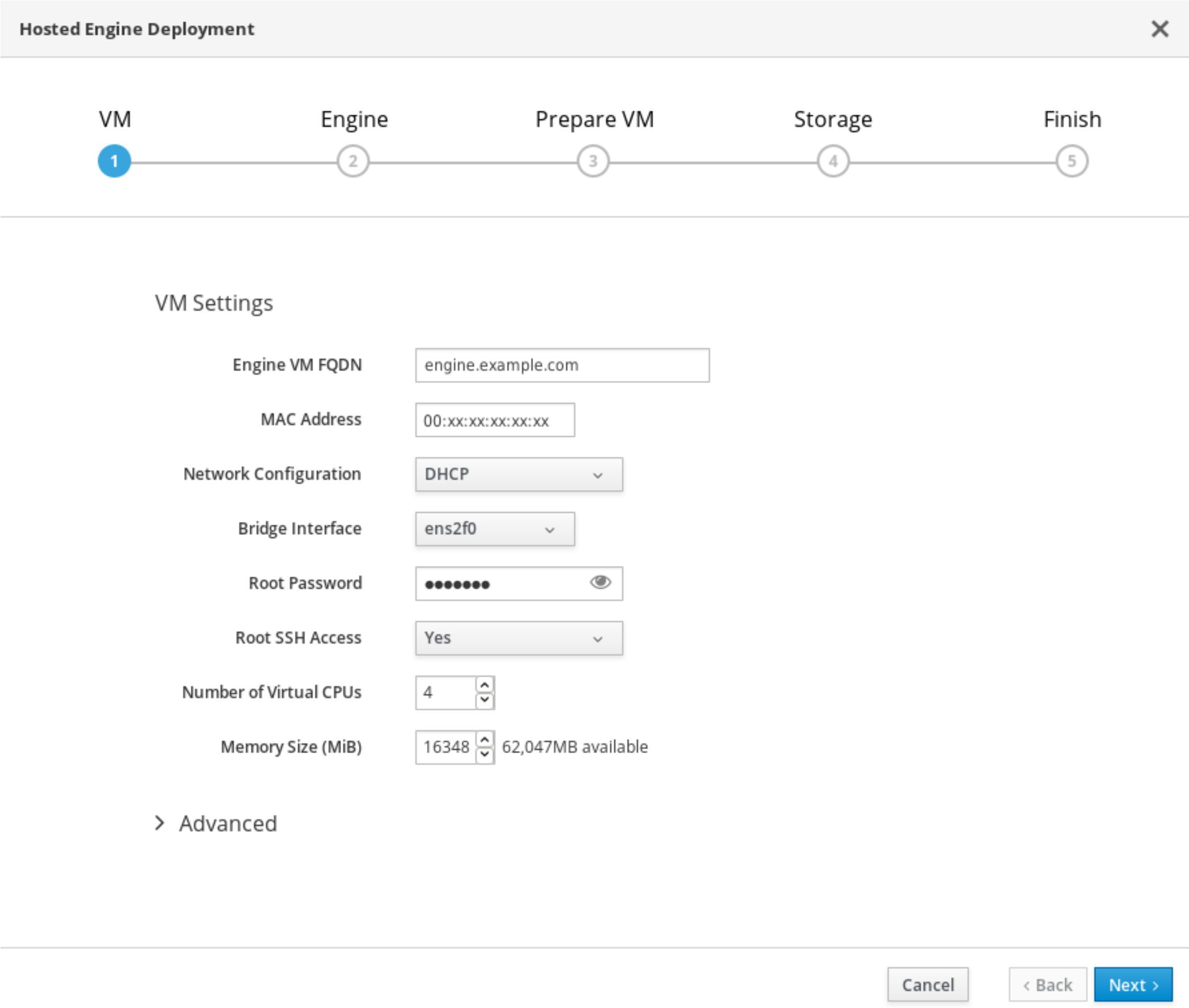
   b. Click Start underneath Hyperconverged.

   c. Click Use existing configuration.

   IMPORTANT

   If the previous deployment attempt failed, click Clean up instead of Use existing configuration to discard the previous attempt and start from scratch. If your deployment uses Network-Bound Disk Encryption, you must then follow the process in Cleaning up Network-Bound Disk Encryption after a failed deployment.

2. Specify virtual machine details



a. Enter the following details:

Engine VM FQDN

The fully qualified domain name to be used for the Hosted Engine virtual machine, for example, **engine.example.com**.

MAC Address

The MAC address associated with the Engine VM FQDN.

IMPORTANT

The pre-populated MAC address must be replaced.

Network Configuration

Choose either DHCP or Static from the Network Configuration drop-down list.

- If you choose DHCP, you must have a DHCP reservation for the Hosted Engine virtual machine so that its host name resolves to the address received from DHCP. Specify its MAC address in the MAC Address field.
- If you choose Static, enter the following details:
  - VM IP Address - The IP address must belong to the same subnet as the host. For example, if the host is in 10.1.1.0/24, the Hosted Engine virtual machine's IP must be in the same subnet range (10.1.1.1-254/24).

- Gateway Address
- DNS Servers

Bridge Interface

Select the Bridge Interface from the drop-down list.

Root password

The root password to be used for the Hosted Engine virtual machine.

Root SSH Access

Specify whether to allow Root SSH Access.The default value of Root SSH Access is set to Yes.

Number of Virtual CPUs

Enter the Number of Virtual CPUs for the virtual machine.

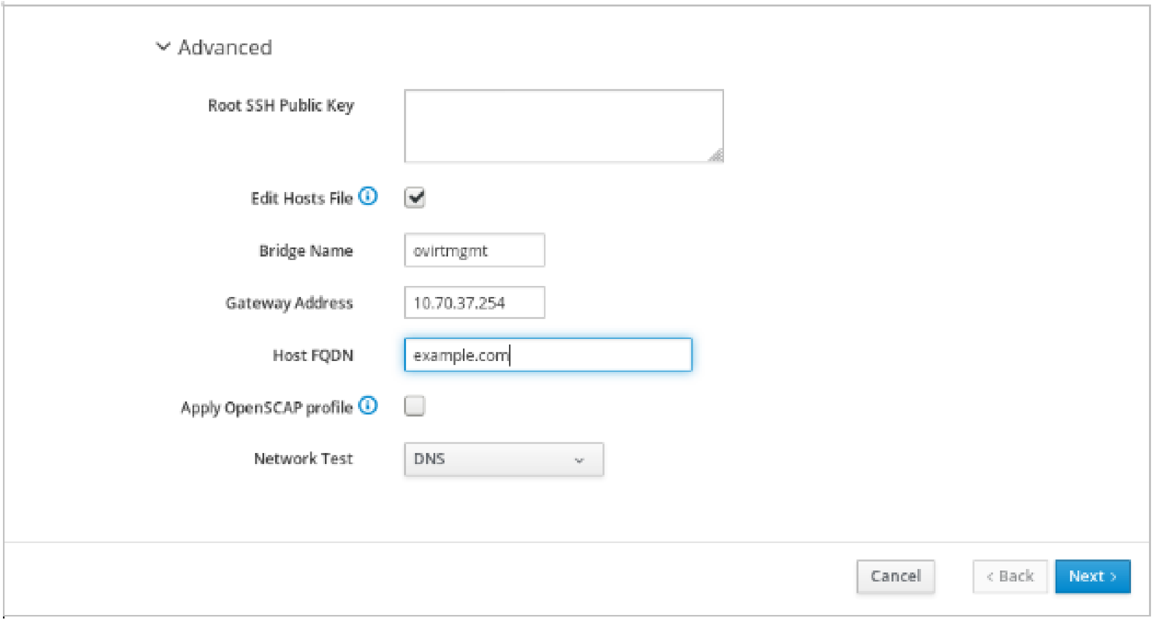
Memory Size (MiB)

Enter the Memory Size (MiB). The available memory is displayed next to the input field.

NOTE

Red Hat recommends to retain the values of Root SSH Access, Number of Virtual CPUs and Memory Size to default values.

b. Optionally expand the Advanced fields.



Root SSH Public Key

Enter a Root SSH Public Key to use for root access to the Hosted Engine virtual machine.

Edit Hosts File

Select or clear the Edit Hosts File check box to specify whether to add entries for the Hosted Engine virtual machine and the base host to the virtual machine's /etc/hosts file. You must ensure that the host names are resolvable.

Bridge Name

Change the management Bridge Name, or accept the default ovirtmgmt.

Gateway Address

Enter the Gateway Address for the management bridge.

Host FQDN

Enter the Host FQDN of the first host to add to the Manager. This is the front-end FQDN of the base host you are running the deployment on.

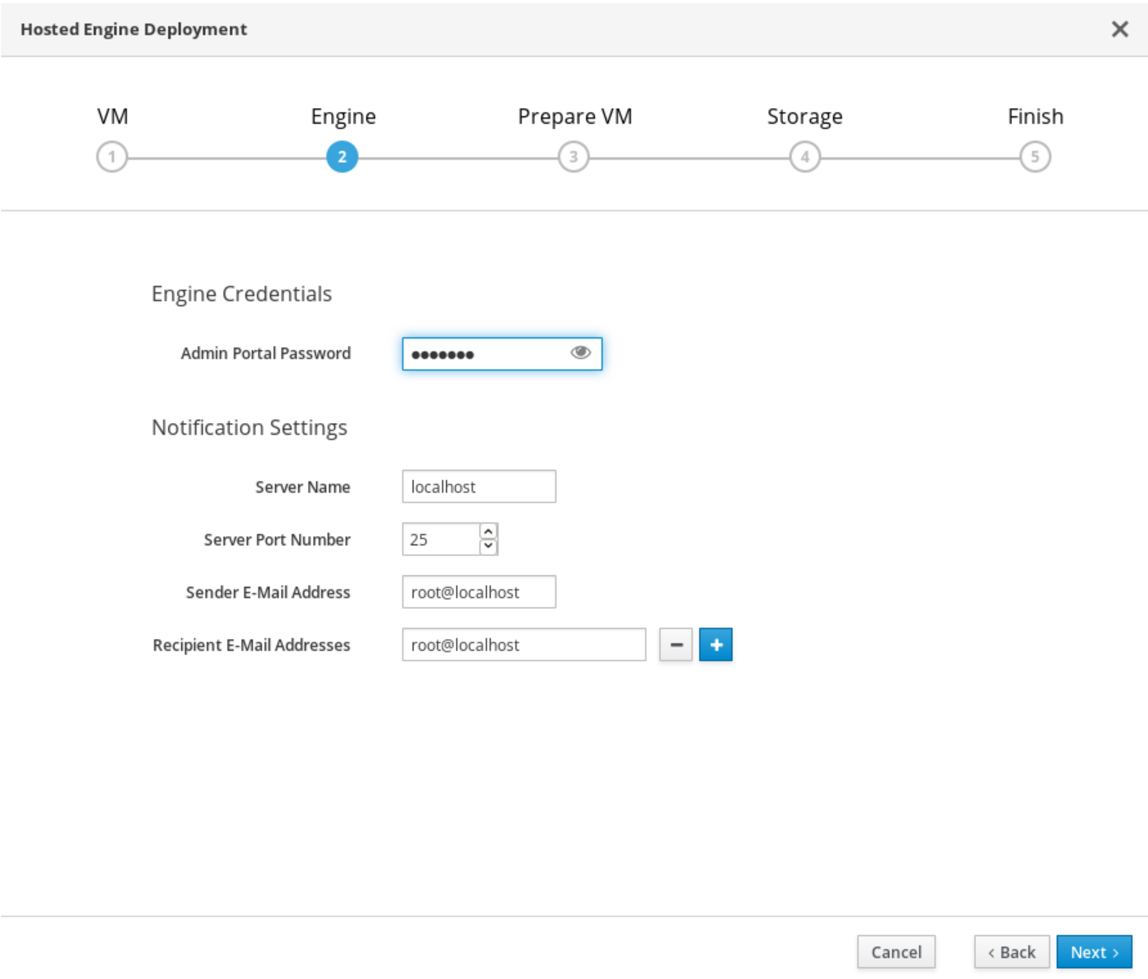
Network Test

If you have a static network configuration or are using an isolated environment with addresses defined in /**etc**/**hosts**, set Network Test toPing.

c. Click Next. Your FQDNs are validated before the next screen appears.

3. Specify virtualization management details

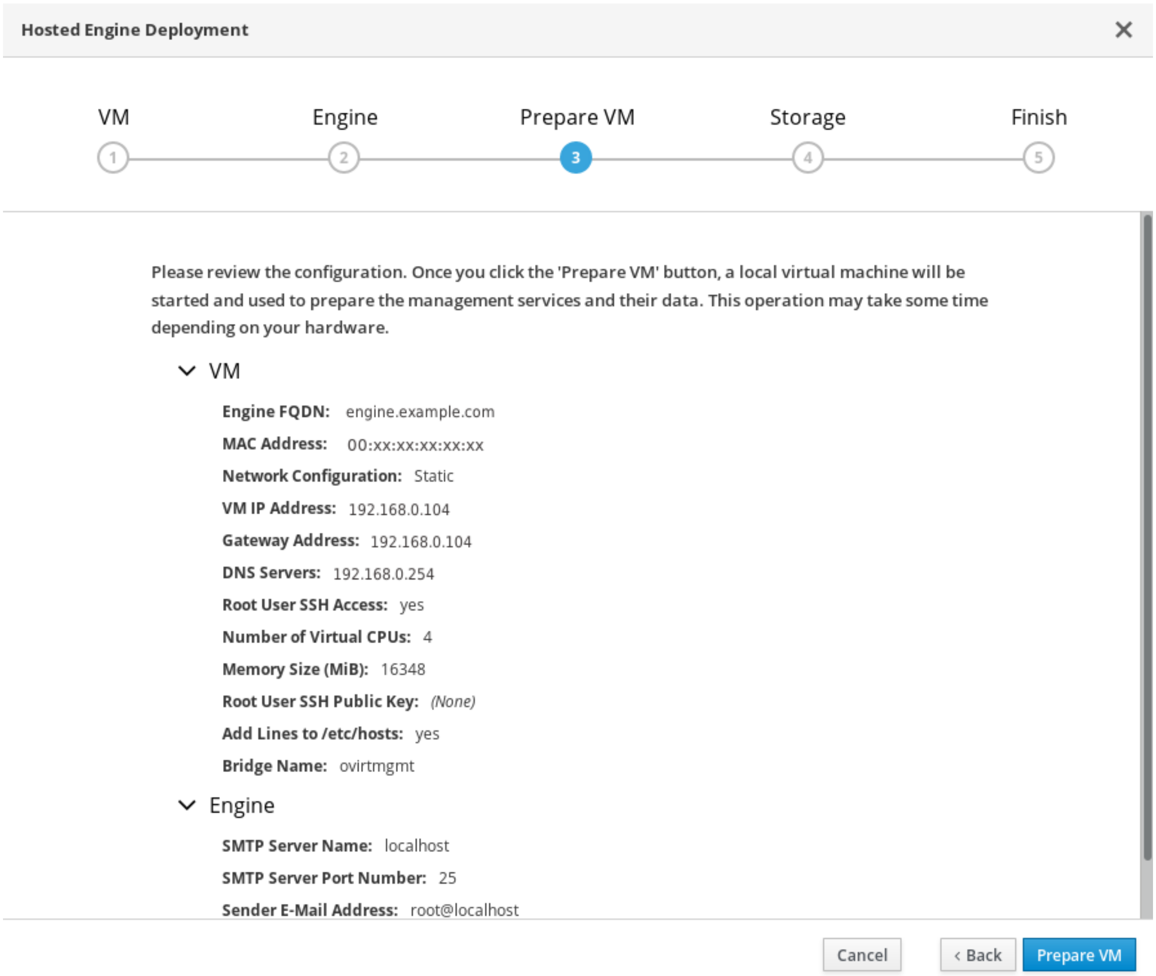
a. Enter the password to be used by the **admin** account in the Administration Portal. You can also specify an email address for notifications, the notifications can also be configured post deployment; see Chapter 15, *Post-deployment configuration suggestions*.



b. Click Next.

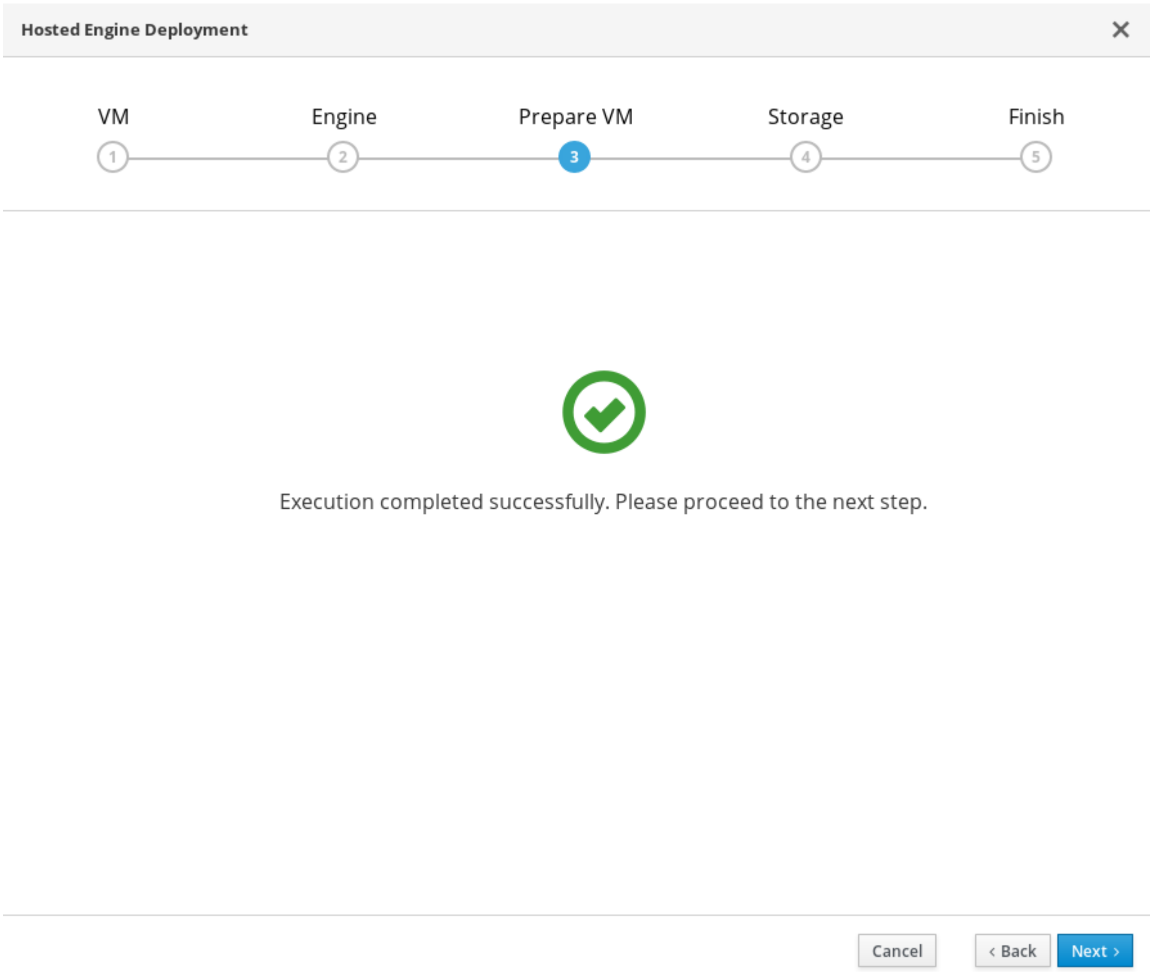
4. Review virtual machine configuration

a. Ensure that the details listed on this tab are correct. Click Back to correct any incorrect information.



b. Click Prepare VM.

c. Wait for virtual machine preparation to complete.



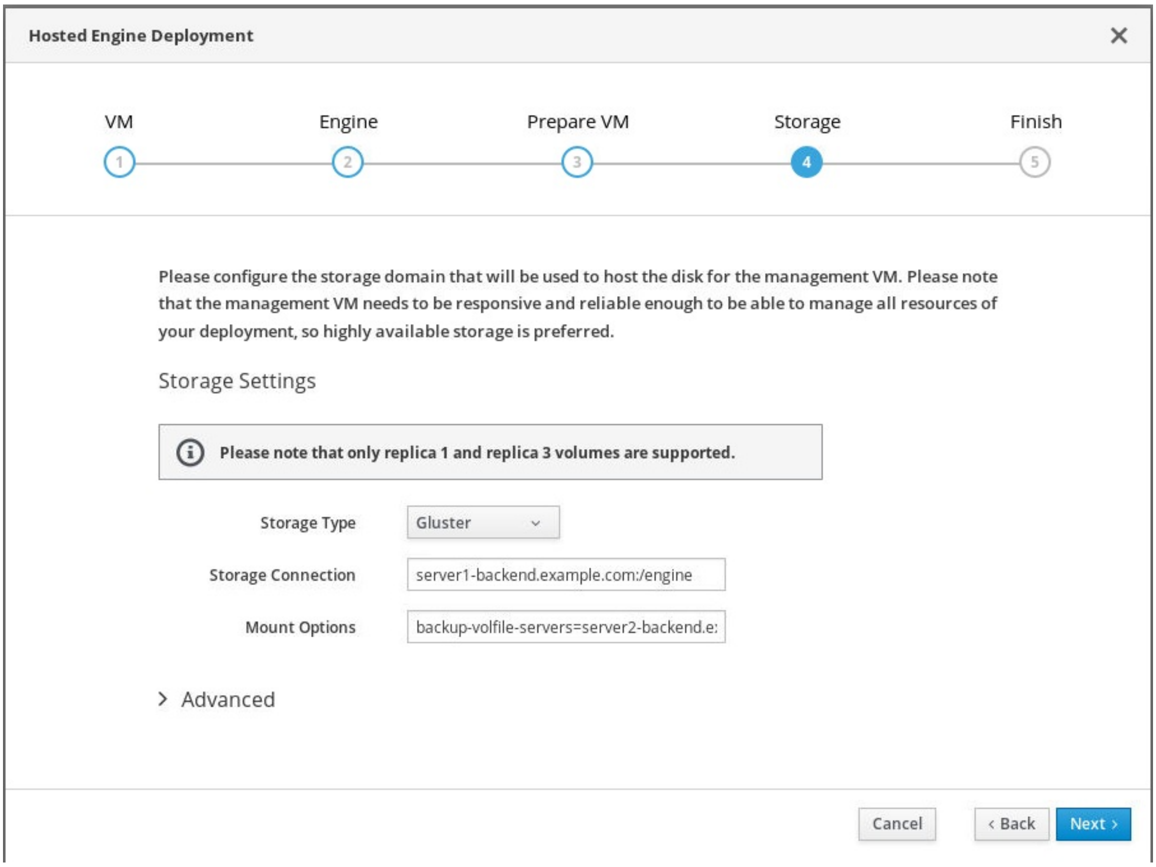
If preparation does not occur successfully, see Viewing Hosted Engine deployment errors.

d. Click Next.

5. Validate storage for the Hosted Engine virtual machine

   a. Ensure that the Mount Options field is populated correctly with**backup-volfile-servers=<host2-ip-address>:<host3-ip-address>** and, if you use IPv6,**xlator-option=transport.address-family=inet6**, for example:

```
backup-volfile-servers=<host2-ip-address>:<host3-ip-address>,xlator-
option=transport.address-family=inet6
```



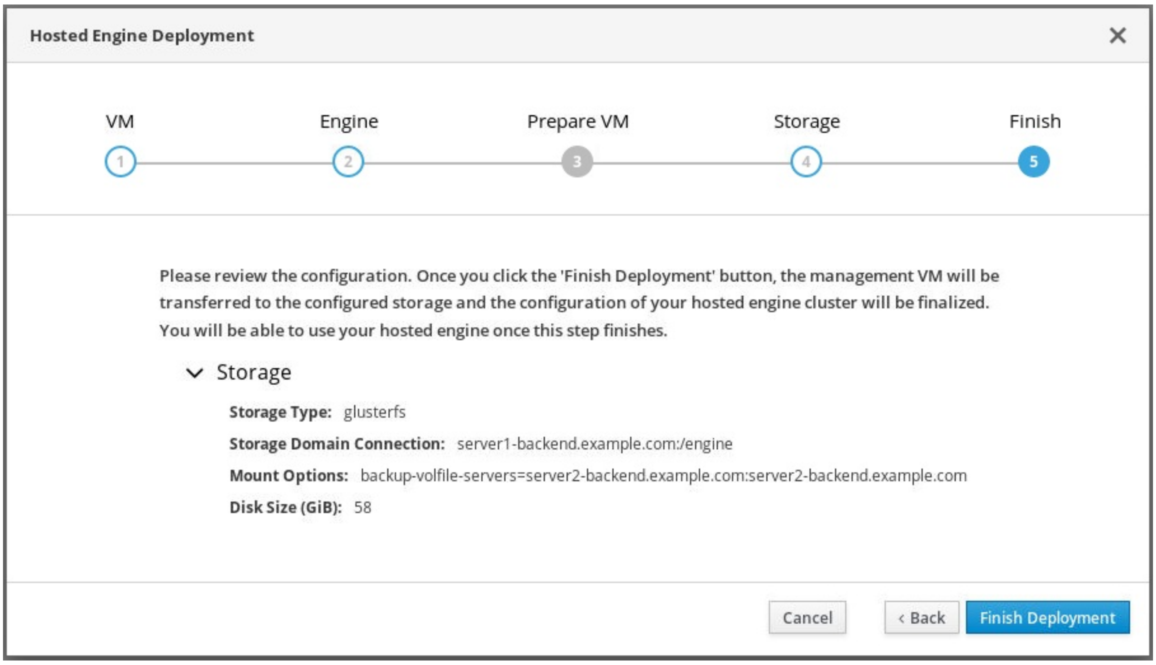
b. Click Next.

6. Finalize Hosted Engine deployment

a. Review your deployment details and verify that they are correct.

NOTE

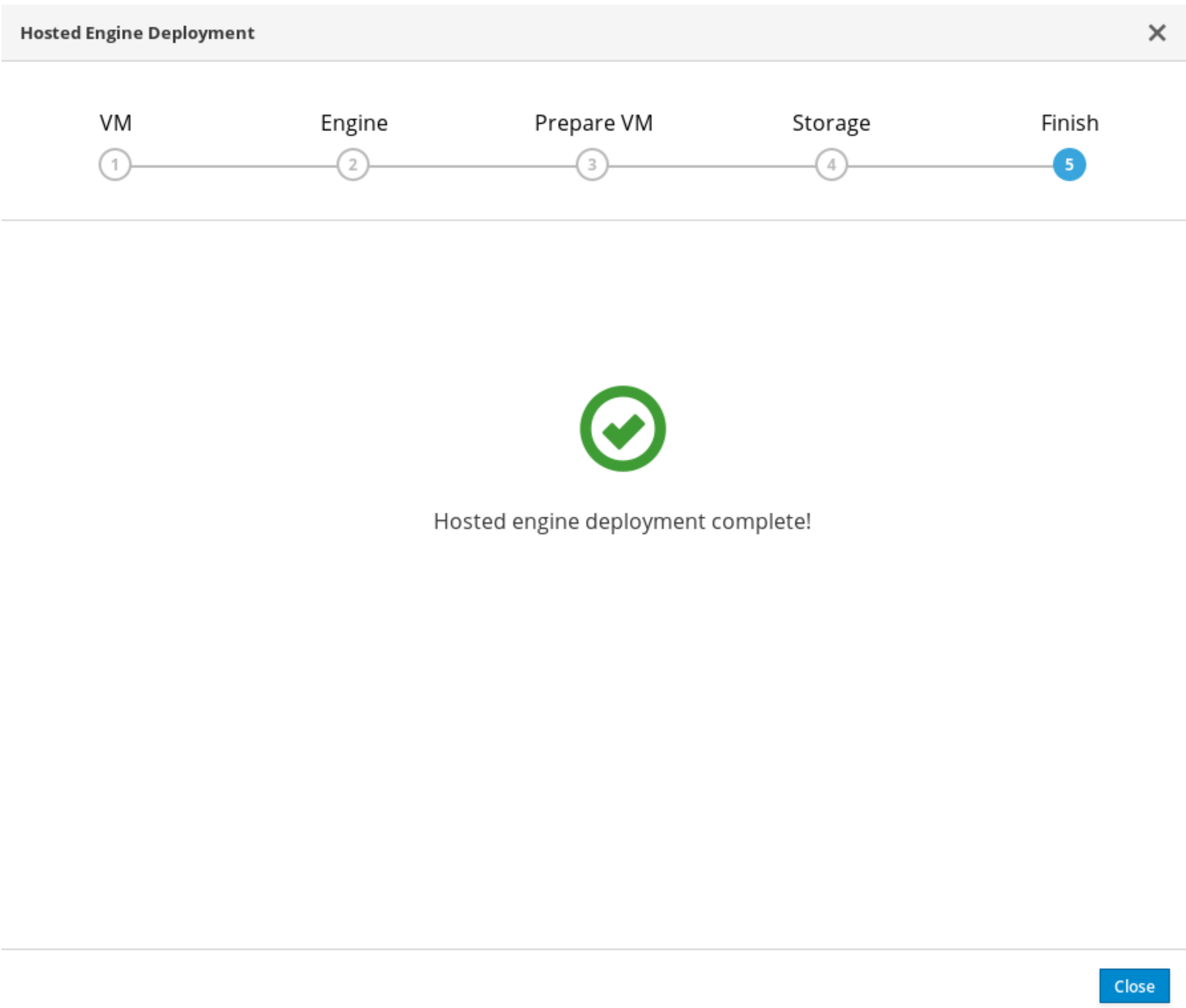
The responses you provided during configuration are saved to an answer file to help you reinstall the hosted engine if necessary. The answer file is created at **/etc/ovirt-hosted-engine/answers.conf** by default. This file should not be modified manually without assistance from Red Hat Support.



b. Click Finish Deployment.

7. Wait for deployment to complete
This can take some time, depending on your configuration details.

The window displays the following when complete.

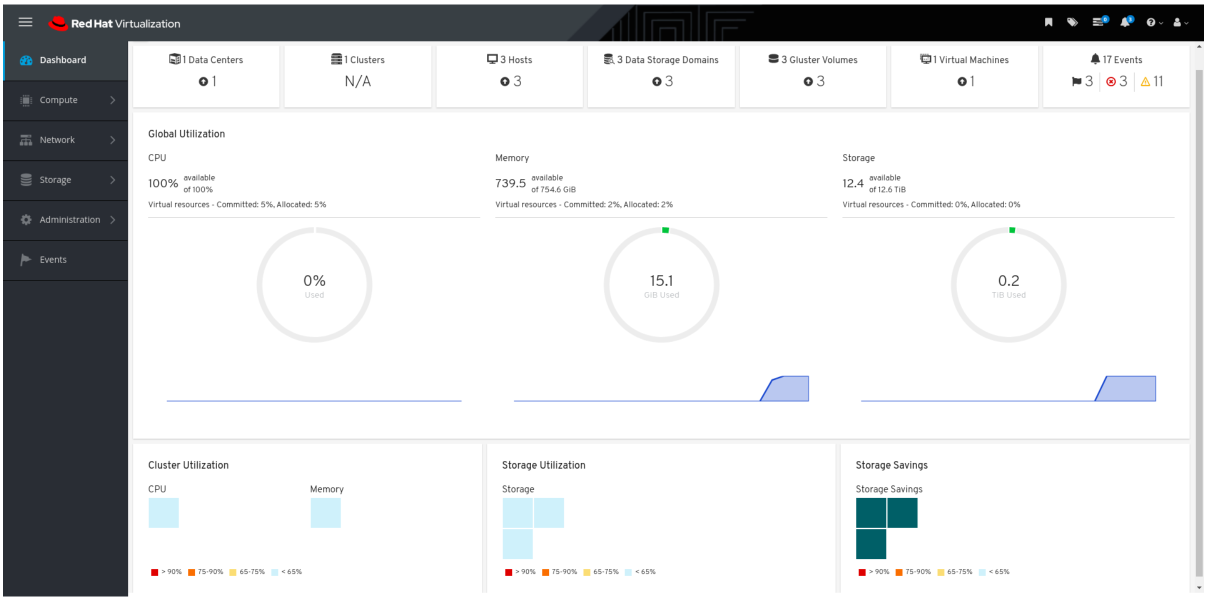


IMPORTANT

After RHHI-V deployment is completed successfully, the other 2 Red Hat Virtualization hosts are rebooted. Wait for these hosts to be up, after which the Red Hat Virtualization Administration portal can be accessible. If deployment does not complete successfully, see Viewing Hosted Engine deployment errors.

Click Close.

8. Verify hosted engine deployment
Browse to the Administration Portal (for example, http://engine.example.com/ovirt-engine) and verify that you can log in using the administrative credentials you configured earlier. Click Dashboard and look for your hosts, storage domains, and virtual machines.



Next steps

- Log in to the Administration Portal to complete configuration

CHAPTER 12. CONFIGURE THE LOGICAL NETWORK FOR GLUSTER TRAFFIC

For creating a separate gluster logical network, in Red Hat Hyperconverged Infrastructure for Virtualization (RHHI for Virtualization) 1.7 users had to perform the steps manually via the Red Hat Virtualization Administration portal. From RHHI for Virtualization 1.8 this process can be automated using the ansible playbook as follows:

12.1. DEFINING THE LOGICAL NETWORK DETAILS FOR GLUSTER TRAFFIC

Prerequisites

- Red Hat Hyperconverged Infrastructure for Virtualization deployment is complete with hosts in up state.

Procedure

1. Log in to the first hyperconverged host.
2. Change into the **hc-ansible-deployment** directory:

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

3. Make a copy of the **gluster_network_inventory.yml** file for future reference.

   ```
   # cp gluster_network_inventory.yml gluster_network_inventory.yml.backup
   ```

4. Define your configuration in the **gluster_network_inventory.yml** file.
   Use the example **gluster_network_inventory.yml** file to define the details on each host. A complete outline of this file is available in Understanding the gluster_network_inventory.yml file.
5. Encrypt the **gluster_network_inventory.yml** file and specify a password using**ansible-vault**.
   The required variables in **gluster_network_inventory.yml** include password values, so it is important to encrypt the file to protect the password values.

   ```
   # ansible-vault encrypt gluster_network_inventory.yml
   ```

   Enter and confirm a new vault password when prompted.

12.2. EXECUTING THE GLUSTER NETWORK PLAYBOOK

Prerequisites

- Define configuration in the **gluster_network_inventory.yml** playbook: Section 12.1, “Defining the logical network details for gluster traffic”.

Procedure

1. Log in to the first hyperconverged host.

2. Change into the hc-ansible-deployment directory.

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

3. Run the following command as the root user to start the configuration process.

```
# ansible-playbook -i gluster_network_inventory.yml  tasks/gluster_network_setup.yml --ask-vault-pass
```

Enter the vault password for this file when prompted to start network configuration.

12.3. VERIFYING THE LOGICAL NETWORK FOR GLUSTER TRAFFIC

Check the following to verify if the logical network for gluster traffic is successfully created and attached to the host.

1. Validate the availability of gluster logical network.
   a. Log in to the Administration Portal.
   b. Click Network → Networks. This should list the newly created **gluster_net** network.
   c. Click on gluster_net → click on Clusters tab, hovering the mouse over Network Role column should display **Migration Gluster.**
2. Validate **gluster_net** is attached to the storage network interface of all the hosts.
   a. Click on Compute → Hosts → click on any host.
   b. Select Network Interfaces tab → click on the drop down button near the label Logical Networks corresponding to storage or backend network, you should see the **gluster_net** as the network name.

12.4. (OPTIONAL) EDITING THE LOGICAL NETWORK FOR JUMBO FRAMES

If you have Jumbo frames (MTU 9000) enabled, you need to edit the default network configuration to ensure jumbo frames are used for storage traffic. The network components (switch) must support Jumbo frames.

The following is the procedure to edit the logical network for Jumbo frames on the storage network, *gluster_net* here:

Prerequisites

- Logical network for gluster traffic is successfully created and is attached to the host.

Procedure

1. Login in to the Administration Portal.
2. Click Networks → Network.
3. Click on gluster_net → Edit.

4. Select custom MTU and make it as 9000.

5. Click OK.

NOTE

Make sure all the network components are enabled with the same MTU.

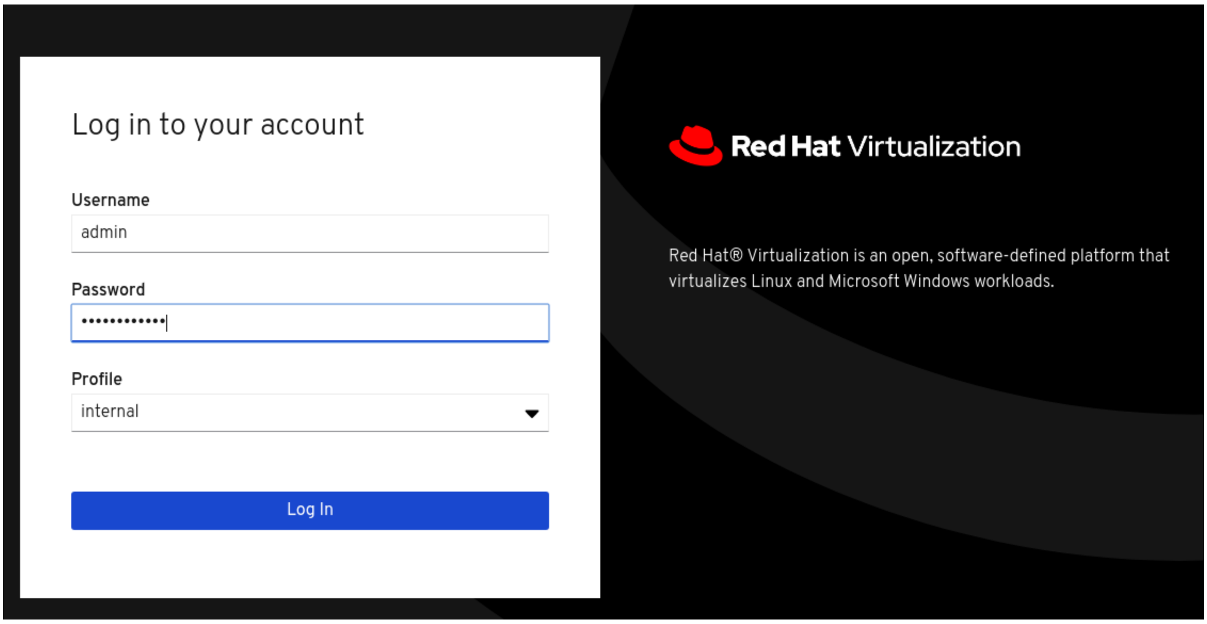
PART III. VERIFY

CHAPTER 13. VERIFY YOUR DEPLOYMENT

After deployment is complete, verify that your deployment has completed successfully.

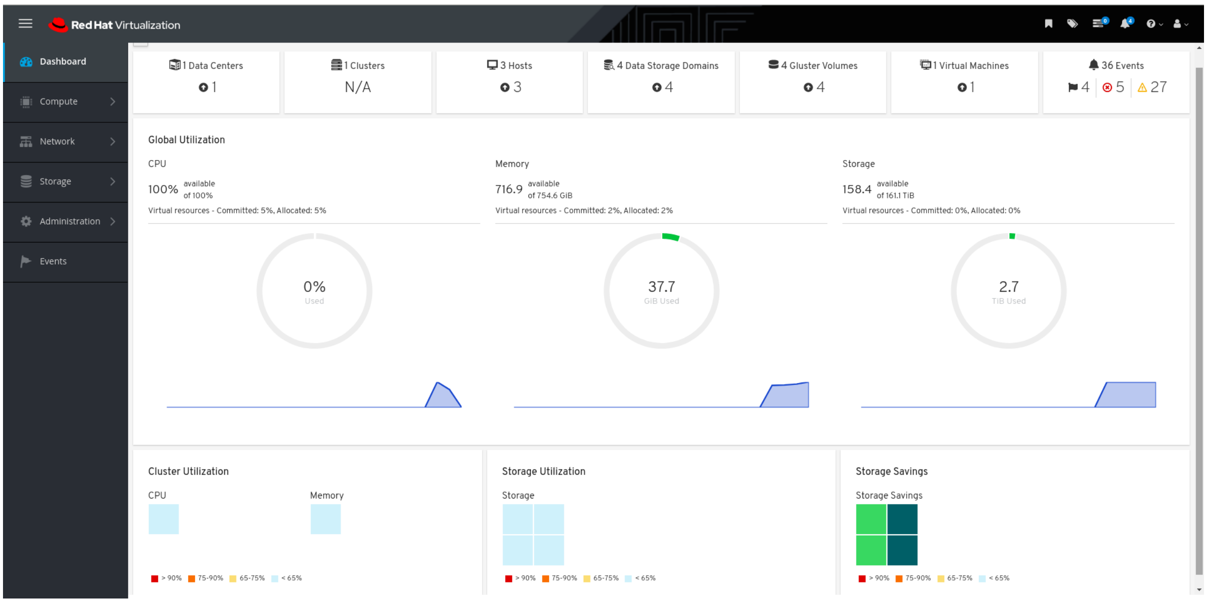
1. Browse to the Administration Portal, for example, http://engine.example.com/ovirt-engine.

   Administration Console Login



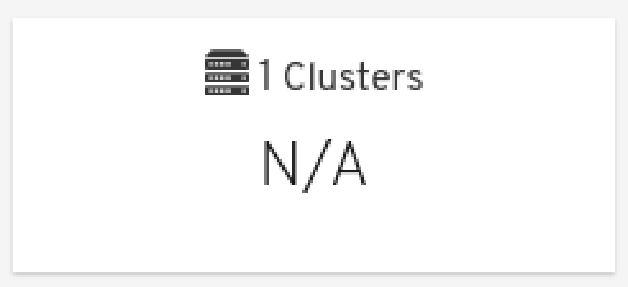
2. Log in using the administrative credentials added during hosted engine deployment. When login is successful, the Dashboard appears.

   Administration Console Dashboard



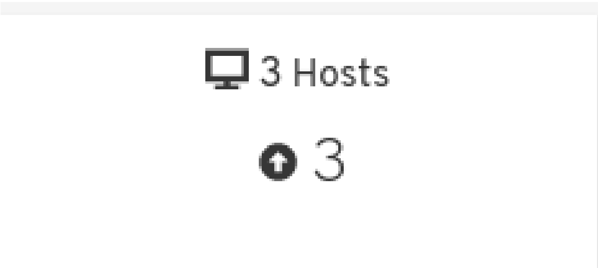
3. Verify that your cluster is available.

   Administration Console Dashboard - Clusters



4. Verify that at least one host is available.
   If you provided additional host details during Hosted Engine deployment, 3 hosts are visible here, as shown.

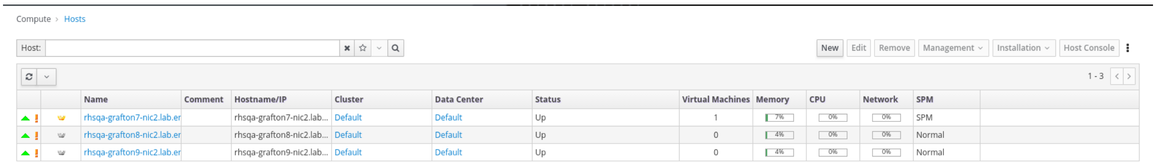
   Administration Console Dashboard - Hosts

   a. Click Compute → Hosts.

   b. Verify that all hosts are listed with a Status of **Up**.

   Administration Console - Hosts

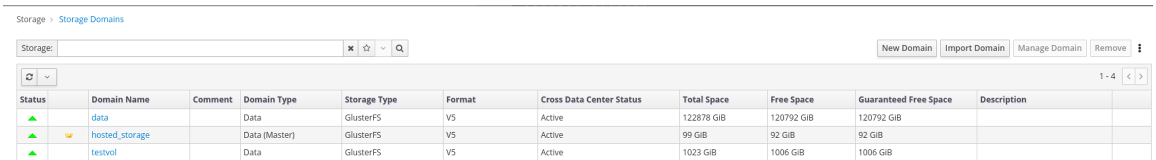
		Name	Comment	Hostname/IP	Cluster	Data Center	Status	Virtual Machines	Memory	CPU	Network	SPM
		rhsqa-grafton7-nic2.lab.er		rhsqa-grafton7-nic2.lab...	Default	Default	Up	1	7%	0%	0%	SPM
		rhsqa-grafton8-nic2.lab.er		rhsqa-grafton8-nic2.lab...	Default	Default	Up	0	4%	0%	0%	Normal
		rhsqa-grafton9-nic2.lab.er		rhsqa-grafton9-nic2.lab...	Default	Default	Up	0	4%	0%	0%	Normal

5. Verify that all storage domains are available.

   a. Click Storage → Domains.

   b. Verify that the **Active** icon is shown in the first column.

   Administration Console - Storage Domains

Status		Domain Name	Comment	Domain Type	Storage Type	Format	Cross Data Center Status	Total Space	Free Space	Guaranteed Free Space	Description
		data		Data	GlusterFS	V5	Active	122878 GiB	120792 GiB	120792 GiB	
		hosted_storage		Data (Master)	GlusterFS	V5	Active	99 GiB	92 GiB	92 GiB	
		testvol		Data	GlusterFS	V5	Active	1023 GiB	1006 GiB	1006 GiB	

PART IV. NEXT STEPS

CHAPTER 14. ENABLING THE RED HAT VIRTUALIZATION MANAGER REPOSITORIES

Register the system with Red Hat Subscription Manager, attach the **Red Hat Virtualization Manager** subscription, and enable Manager repositories.

Procedure

1. Register your system with the Content Delivery Network, entering your Customer Portal user name and password when prompted:

```
# subscription-manager register
```

NOTE

If you are using an IPv6 network, use an IPv6 transition mechanism to access the Content Delivery Network and subscription manager.

2. Find the Red Hat Virtualization Manager subscription pool and record the pool ID:

```
# subscription-manager list --available
```

3. Use the pool ID to attach the subscription to the system:

```
# subscription-manager attach --pool=pool_id
```

NOTE

To view currently attached subscriptions:

```
# subscription-manager list --consumed
```

To list all enabled repositories:

```
# yum repolist
```

4. Configure the repositories:

```
# subscription-manager repos \
    --disable='*' \
    --enable=rhel-8-for-x86_64-baseos-rpms \
    --enable=rhel-8-for-x86_64-appstream-rpms \
    --enable=rhv-4.4-manager-for-rhel-8-x86_64-rpms \
    --enable=fast-datapath-for-rhel-8-x86_64-rpms \
    --enable=jb-eap-7.4-for-rhel-8-x86_64-rpms \
    --enable=openstack-16.2-cinderlib-for-rhel-8-x86_64-rpms \
    --enable=rhceph-4-tools-for-rhel-8-x86_64-rpms
```

5. Enable the pki-deps module.

```
# yum module -y enable pki-deps
```

6. Enable version 12 of the postgresql module.

```
# yum module -y enable postgresql:12
```

7. Synchronize installed packages to update them to the latest available versions.

```
# yum distro-sync
```

Additional resources

For information on modules and module streams, see the following sections in *Installing, managing, and removing user-space* components

- Module streams
- Selecting a stream before installation of packages
- Resetting module streams
- Switching to a later stream

CHAPTER 15. POST-DEPLOYMENT CONFIGURATION SUGGESTIONS

Depending on your requirements, you may want to perform some additional configuration on your newly deployed Red Hat Hyperconverged Infrastructure for Virtualization. This section contains suggested next steps for additional configuration.

Details on these processes are available in *Maintaining Red Hat Hyperconverged Infrastructure for Virtualization*.

15.1. CONFIGURE NOTIFICATIONS

See Configuring Event Notifications in the Administration Portal to configure email notifications.

15.2. (OPTIONAL)CONFIGURE HOST POWER MANAGEMENT

The Red Hat Virtualization Manager 4.4 is capable of rebooting hosts that have entered a non-operational or non-responsive state, as well as preparing to power off under-utilized hosts to save power. This functionality depends on a properly configured power management device.

See Configuring Host Power Management Settings for further information.

15.3. CONFIGURE BACKUP AND RECOVERY OPTIONS

Red Hat recommends configuring at least basic disaster recovery capabilities on all production deployments.

See Configuring backup and recovery options in *Maintaining Red Hat Hyperconverged Infrastructure for Virtualization* for more information.

PART V. TROUBLESHOOT

CHAPTER 16. LOG FILE LOCATIONS

During the deployment process, progress information is displayed in the web browser. This information is also stored on the local file system so that the information logged can be archived or reviewed at a later date, for example, if the web browser stops responding or is closed before the information has been reviewed.

The log file for the Web Console based deployment process (documented in Chapter 10, *Configure Red Hat Gluster Storage for Hosted Engine using the Web Console*) is stored in the **/var/log/cockpit/ovirt-dashboard/gluster-deployment.log** file by default.

The log files for the Hosted Engine setup portion of the deployment process (documented in Chapter 11, *Deploy the Hosted Engine using the Web Console*) are stored in the **/var/log/ovirt-hosted-engine-setup** directory, with file names of the form **ovirt-hosted-engine-setup-<date>.log**.

CHAPTER 17. DEPLOYMENT ERRORS

17.1. ORDER OF CLEANUP OPERATIONS

Depending on where deployment fails, you may need to perform a number of cleanup operations.

Always perform cleanup for tasks in reverse order to the order of the tasks themselves. For example, during deployment, we perform the following tasks in order:

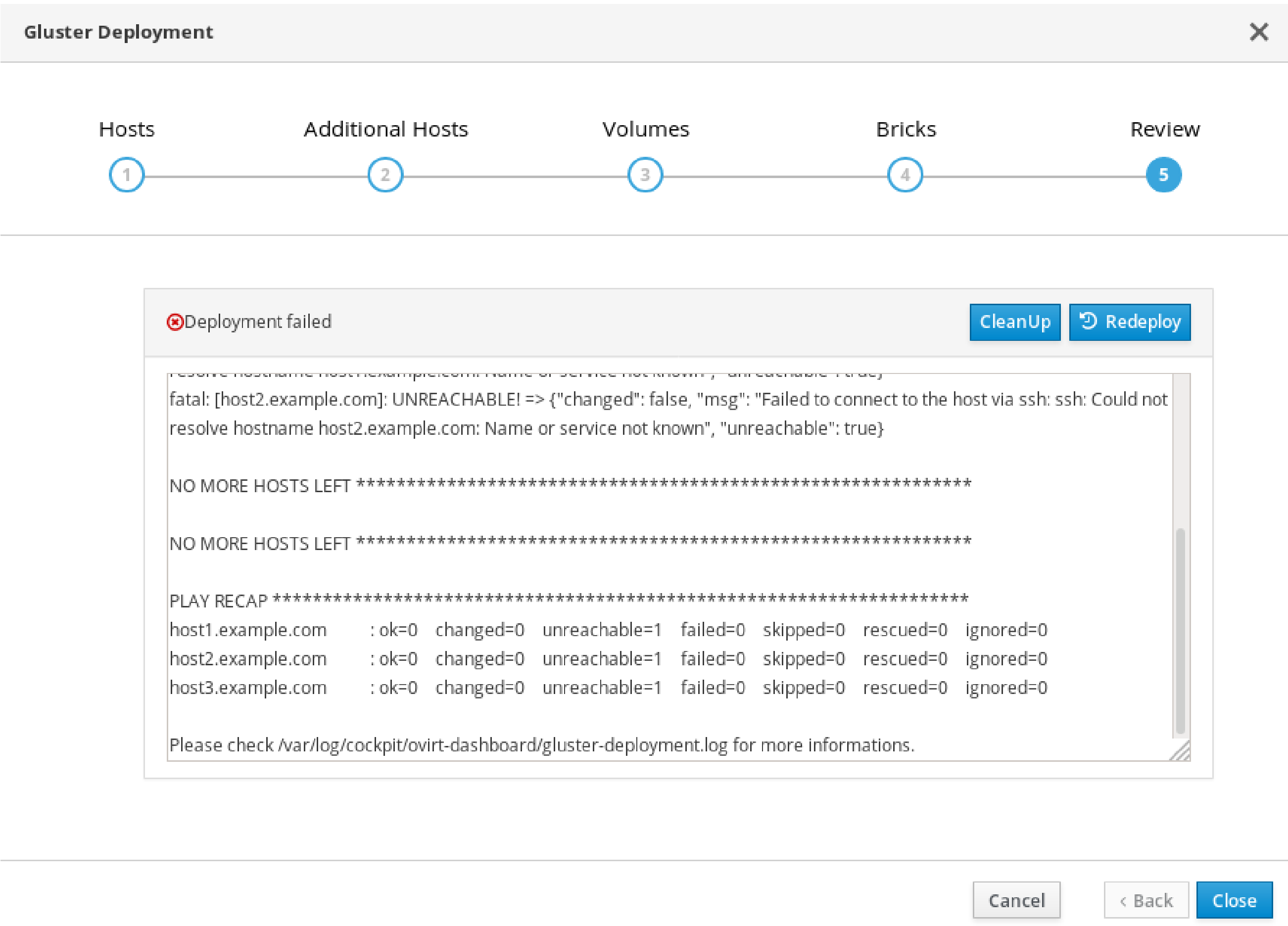
1. Configure Network-Bound Disk Encryption using Ansible.
2. Configure Red Hat Gluster Storage using the Web Console.
3. Configure the Hosted Engine using the Web Console.

If deployment fails at step 2, perform cleanup for step 2. Then, if necessary, perform cleanup for step 1.

17.2. FAILED TO DEPLOY STORAGE

If an error occurs during storage deployment, the deployment process halts and**Deployment failed** is displayed.

Deploying storage failed



- Review the Web Console output for error information.

- Click Clean up to remove any potentially incorrect changes to the system. If your deployment uses Network-Bound Disk Encryption, you must then follow the process in Cleaning up Network-Bound Disk Encryption after a failed deployment.

- Click Redeploy and correct any entered values that may have caused errors. If you need help resolving errors, contact Red Hat Support with details.

- Return to storage deployment to try again.

17.2.1. Cleaning up Network-Bound Disk Encryption after a failed deployment

If you are using Network-Bound Disk Encryption and deployment fails, you cannot just click the Cleanup button in order to try again. You must also run the **luks_device_cleanup.yml** playbook to complete the cleaning process before you start again.

Run this playbook as shown, providing the same **luks_tang_inventory.yml** file that you provided during setup.

```
# ansible-playbook -i luks_tang_inventory.yml /etc/ansible/roles/gluster.ansible/playbooks/hc-ansible-deployment/tasks/luks_device_cleanup.yml --ask-vault-pass
```

17.2.2. Error: VDO signature detected on device

During storage deployment, the **Create VDO with specified size** task may fail with the **VDO signature detected on device** error.

```
TASK [gluster.infra/roles/backend_setup : Create VDO with specified size]
task path: /etc/ansible/roles/gluster.infra/roles/backend_setup/tasks/vdo_create.yml:9
failed: [host1.example.com] (item={u'writepolicy': u'auto', u'name': u'vdo_sdb', u'readcachesize': u'20M', u'readcache': u'enabled', u'emulate512': u'off', u'logicalsize': u'11000G', u'device': u'/dev/sdb', u'slabsize': u'32G', u'blockmapcachesize': u'128M'}) => {"ansible_loop_var": "item", "changed": false, "err": "vdo: ERROR - vdo signature detected on /dev/sdb at offset 0; use --force to override\n", "item": {"blockmapcachesize": "128M", "device": "/dev/sdb", "emulate512": "off", "logicalsize": "11000G", "name": "vdo_sdb", "readcache": "enabled", "readcachesize": "20M", "slabsize": "32G", "writepolicy": "auto"}, "msg": "Creating VDO vdo_sdb failed.", "rc": 5}
```

This error occurs when the specified device is already a VDO device, or when the device was previously configured as a VDO device and was not cleaned up correctly.

- If you specified a VDO device accidentally, return to storage configuration and specify a different non-VDO device.

- If you specified a device that has been used as a VDO device previously:

  a. Check the device type.

  ```
  # blkid -p /dev/sdb
  /dev/sdb: UUID="fee52367-c2ca-4fab-a6e9-58267895fe3f" TYPE="vdo" USAGE="other"
  ```

  If you see **TYPE="vdo"** in the output, this device was not cleaned correctly.

  b. Follow the steps in Manually cleaning up a VDO device to use this device. Then return to storage deployment to try again.

Avoid this error by specifying clean devices, and by using the Clean up button in the storage deployment window to clean up any failed deployments.

17.2.3. Manually cleaning up a VDO device

Follow this process to manually clean up a VDO device that has caused a deployment failure.

> **WARNING**
>
> This is a destructive process. You will lose all data on the device that you clean up.

Procedure

- Clean the device using wipefs.

```
# wipefs -a /dev/sdX
```

Verify

- Confirm that the device does not have **TYPE="vdo"** set any more.

```
# blkid -p /dev/sdb
/dev/sdb: UUID="fee52367-c2ca-4fab-a6e9-58267895fe3f" TYPE="vdo" USAGE="other"
```

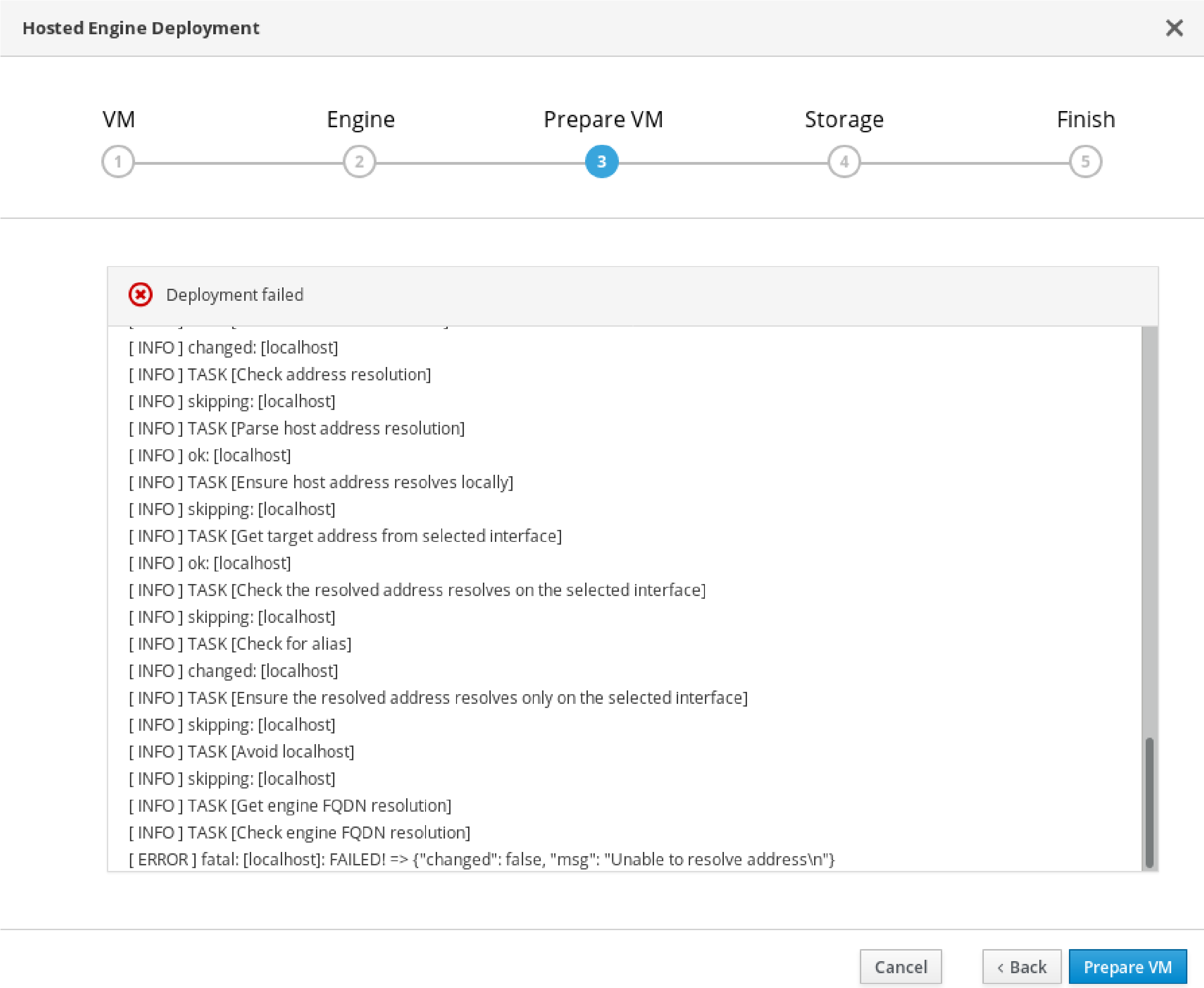
Next steps

- Return to storage deployment to try again.

17.3. FAILED TO PREPARE VIRTUAL MACHINE

If an error occurs while preparing the virtual machine in Hosted Engine deployment, deployment pauses, and you see a screen similar to the following:

Preparing virtual machine failed



- Review the Web Console output for error information.
- Click Back and correct any entered values that may have caused errors. Ensure proper values for network configurations are provided in VM tab. If you need help resolving errors, contact Red Hat Support with details.
- Ensure that the **rhvm-appliance** package is available on the first hyperconverged host.

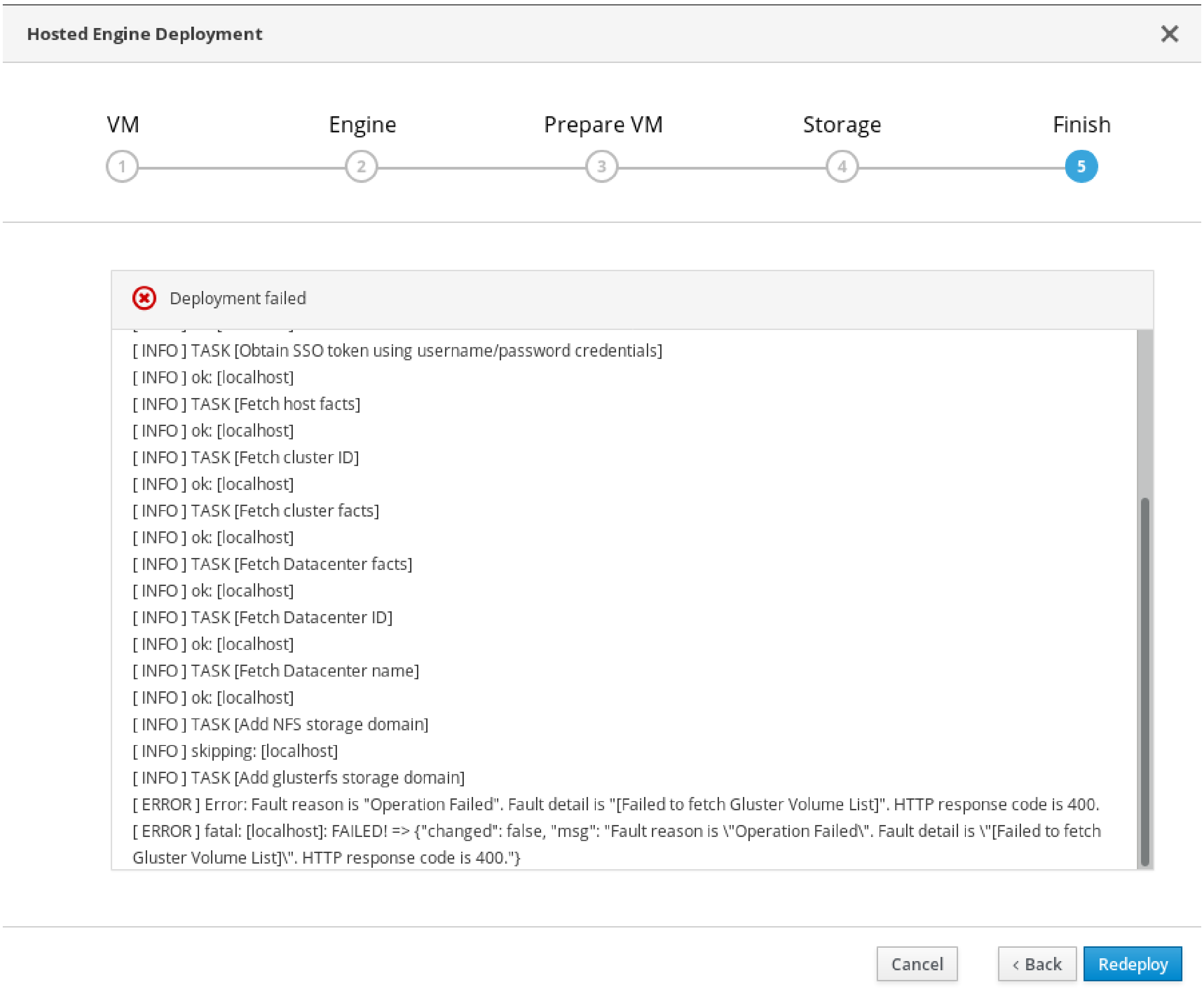
```
# yum install rhvm-appliance
```

- Return to Hosted Engine deployment to try again.
  If you closed the deployment wizard while you resolved errors, you can select Use existing configuration when you retry the deployment process.

17.4. FAILED TO DEPLOY HOSTED ENGINE

If an error occurs during hosted engine deployment, deployment pauses and **Deployment failed** is displayed.

Hosted engine deployment failed



1. **Review the Web Console output for error information.**
2. **Remove the contents of the engine volume.**
   a. **Mount the engine volume.**

   ```
   # mount -t glusterfs <server1>:/engine /mnt/test
   ```

   b. **Remove the contents of the volume.**

   ```
   # rm -rf /mnt/test/*
   ```

   c. **Unmount the engine volume.**

   ```
   # umount /mnt/test
   ```

3. **Click Redeploy and correct any entered values that may have caused errors.**
4. **If the deployment fails after performing the above steps a, b and c. Perform these steps again and this time clean the Hosted Engine:**

   ```
   # ovirt-hosted-engine-cleanup
   ```

5. Return to Hosted Engine deployment to try again.
   If you closed the deployment wizard while you resolved errors, you can select Use existing configuration when you retry the deployment process.

   If you need help resolving errors, contact Red Hat Support with details.

PART VI. 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 LUKS_TANG_INVENTORY.YML FILE

B.1. CONFIGURATION PARAMETERS FOR DISK ENCRYPTION

hc_nodes (required)

A list of hyperconverged hosts that uses the back-end FQDN of the host, and the configuration details of those hosts. Configuration that is specific to a host is defined under that host's back-end FQDN. Configuration that is common to all hosts is defined in the vars: section.

```
hc_nodes:
  hosts:
    host1backend.example.com:
      [configuration specific to this host]
    host2backend.example.com:
    host3backend.example.com:
    host4backend.example.com:
    host5backend.example.com:
    host6backend.example.com:
  vars:
    [configuration common to all hosts]
```

blacklist_mpath_devices (optional)

By default, Red Hat Virtualization Host enables multipath configuration, which provides unique multipath names and worldwide identifiers for all disks, even when disks do not have underlying multipath configuration. Include this section if you do not have multipath configuration so that the multipath device names are not used for listed devices. Disks that are not listed here are assumed to have multipath configuration available, and require the path format **/dev/mapper/<WWID>** instead of **/dev/sdx** when defined in subsequent sections of the inventory file.
On a server with four devices (sda, sdb, sdc and sdd), the following configuration blacklists only two devices. The path format /dev/mapper/<WWID> is expected for devices not in this list.

```
hc_nodes:
  hosts:
    host1backend.example.com:
      blacklist_mpath_devices:
        - sdb
        - sdc
```

gluster_infra_luks_devices (required)

A list of devices to encrypt and the encryption passphrase to use for each device.

```
hc_nodes:
  hosts:
    host1backend.example.com:
      gluster_infra_luks_devices:
        - devicename: /dev/sdb
          passphrase: Str0ngPa55#
```

devicename

The name of the device in the format **/dev/sdx**.

passphrase

The password to use for this device when configuring encryption. After disk encryption with Network-Bound Disk Encryption (NBDE) is configured, a new random key is generated, providing greater security.

rootpassphrase (required)

The password that you used when you selected Encrypt my data during operating system installation on this host.

```
hc_nodes:
  hosts:
    host1backend.example.com:
      rootpassphrase: h1-Str0ngPa55#
```

rootdevice (required)

The root device that was encrypted when you selected Encrypt my data during operating system installation on this host.

```
hc_nodes:
  hosts:
    host1backend.example.com:
      rootdevice: /dev/sda2
```

networkinterface (required)

The network interface this host uses to reach the NBDE key server.

```
hc_nodes:
  hosts:
    host1backend.example.com:
      networkinterface: ens3s0f0
```

ip_version (required)

Whether to use IPv4 or IPv6 networking. Valid values are **IPv4** and **IPv6**. There is no default value. Mixed networks are not supported.

```
hc_nodes:
  vars:
    ip_version: IPv4
```

ip_config_method (required)

Whether to use DHCP or static networking. Valid values are **dhcp** and **static**. There is no default value.

```
hc_nodes:
  vars:
    ip_config_method: dhcp
```

The other valid value for this option is **static**, which requires the following additional parameters and is defined individually for each host:

```
hc_nodes:
```

```
  hosts:
    host1backend.example.com:
      ip_config_method: static
      host_ip_addr: 192.168.1.101
      host_ip_prefix: 24
      host_net_gateway: 192.168.1.100
    host2backend.example.com:
      ip_config_method: static
      host_ip_addr: 192.168.1.102
      host_ip_prefix: 24
      host_net_gateway: 192.168.1.100
    host3backend.example.com:
      ip_config_method: static
      host_ip_addr: 192.168.1.102
      host_ip_prefix: 24
      host_net_gateway: 192.168.1.100
```

gluster_infra_tangservers

The address of your NBDE key server or servers, including **http://**. If your servers use a port other than the default (80), specify a port by appending **:_port_** to the end of the URL.

```
hc_nodes:
  vars:
    gluster_infra_tangservers:
      - url: http://key-server1.example.com
      - url: http://key-server2.example.com:80
```

B.2. EXAMPLE LUKS_TANG_INVENTORY.YML

Dynamically allocated IP addresses

```
hc_nodes:
  hosts:
    host1-backend.example.com:
      blacklist_mpath_devices:
        - sda
        - sdb
        - sdc
      gluster_infra_luks_devices:
        - devicename: /dev/sdb
          passphrase: dev-sdb-encrypt-passphrase
        - devicename: /dev/sdc
          passphrase: dev-sdc-encrypt-passphrase
      rootpassphrase: host1-root-passphrase
      rootdevice: /dev/sda2
      networkinterface: eth0
    host2-backend.example.com:
      blacklist_mpath_devices:
        - sda
        - sdb
        - sdc
      gluster_infra_luks_devices:
        - devicename: /dev/sdb
```

```
        passphrase: dev-sdb-encrypt-passphrase
      - devicename: /dev/sdc
        passphrase: dev-sdc-encrypt-passphrase
    rootpassphrase: host2-root-passphrase
    rootdevice: /dev/sda2
    networkinterface: eth0
  host3-backend.example.com:
    blacklist_mpath_devices:
      - sda
      - sdb
      - sdc
    gluster_infra_luks_devices:
      - devicename: /dev/sdb
        passphrase: dev-sdb-encrypt-passphrase
      - devicename: /dev/sdc
        passphrase: dev-sdc-encrypt-passphrase
    rootpassphrase: host3-root-passphrase
    rootdevice: /dev/sda2
    networkinterface: eth0
 vars:
  ip_version: IPv4
  ip_config_method: dhcp
  gluster_infra_tangservers:
    - url: http://key-server1.example.com:80
    - url: http://key-server2.example.com:80
```

Static IP addresses

```
hc_nodes:
 hosts:
  host1-backend.example.com:
    blacklist_mpath_devices:
      - sda
      - sdb
      - sdc
    gluster_infra_luks_devices:
      - devicename: /dev/sdb
        passphrase: dev-sdb-encrypt-passphrase
      - devicename: /dev/sdc
        passphrase: dev-sdc-encrypt-passphrase
    rootpassphrase: host1-root-passphrase
    rootdevice: /dev/sda2
    networkinterface: eth0
    host_ip_addr: host1-static-ip
    host_ip_prefix: network-prefix
    host_net_gateway: default-network-gateway
  host2-backend.example.com:
    blacklist_mpath_devices:
      - sda
      - sdb
      - sdc
    gluster_infra_luks_devices:
      - devicename: /dev/sdb
        passphrase: dev-sdb-encrypt-passphrase
      - devicename: /dev/sdc
        passphrase: dev-sdc-encrypt-passphrase
```

```
      rootpassphrase: host2-root-passphrase
      rootdevice: /dev/sda2
      networkinterface: eth0
      host_ip_addr: host1-static-ip
      host_ip_prefix: network-prefix
      host_net_gateway: default-network-gateway
    host3-backend.example.com:
      blacklist_mpath_devices:
        - sda
        - sdb
        - sdc
      gluster_infra_luks_devices:
        - devicename: /dev/sdb
          passphrase: dev-sdb-encrypt-passphrase
        - devicename: /dev/sdc
          passphrase: dev-sdc-encrypt-passphrase
      rootpassphrase: host3-root-passphrase
      rootdevice: /dev/sda2
      networkinterface: eth0
      host_ip_addr: host1-static-ip
      host_ip_prefix: network-prefix
      host_net_gateway: default-network-gateway
  vars:
    ip_version: IPv4
    ip_config_method: static
    gluster_infra_tangservers:
      - url: http://key-server1.example.com:80
      - url: http://key-server2.example.com:80
```

APPENDIX C. UNDERSTANDING THE GLUSTER_NETWORK_INVENTORY.YML FILE

C.1. CONFIGURATION PARAMETERS FOR CREATION OF GLUSTER NETWORK

- vars

  he_fqdn
  : FQDN of the hosted engine VM

  he_admin_password
  : Password for RHV Manager Administration Portal.

  datacenter_name
  : RHV datacenter name. Usually Red Hat Hyperconverged Infrastructure for Virtualization deployment adds all 3 hosts to **Default** cluster in **Default** datacenter.

  cluster_name
  : RHV cluster name.

  boot_protocol
  : Whether to use DHCP or static networking. .

  version (optional)
  : Whether to use IPv4 or IPv6 networking. v4 is the default, and is assumed if this parameter is omitted. The other valid value is v6. Mixed networks are not supported.

  mtu_value (optional)
  : Specifies the Maximum Transmission Unit for the network, the largest packet or frame size that can be sent in a single transaction. The default value is **1500**. Increasing this to **9000** on networks that support Jumbo frames greatly improves throughput.

- cluster_nodes

  host
  : Host's public network FQDN, which is mentioned in Red Hat Virtualization Administration Portal.

  interface
  : Network interface or the bond, corresponding to storage or backend network.

C.2. EXAMPLE GLUSTER_NETWORK_INVENTORY.YML

```
all:
 hosts:
  localhost:
 vars:
  he_fqdn: rhv-manager.example.com
  he_admin_password: xxxxxxxxxx
  datacenter_name: Default
  cluster_name: Default
  boot_protocol: dhcp
  version: v4
  mtu_value: 9000
```

```
# For dhcp boot_protocol
cluster_nodes:
- {host: host1-frontend.example.com, interface: eth1}
- {host: host2-frontend.example.com, interface: eth1}
- {host: host3-frontend.example.com, interface: eth1}
```

APPENDIX D. GLOSSARY OF TERMS

D.1. VIRTUALIZATION TERMS

Administration Portal

A web user interface provided by Red Hat Virtualization Manager, based on the oVirt engine web user interface. It allows administrators to manage and monitor cluster resources like networks, storage domains, and virtual machine templates.

Hosted Engine

The instance of Red Hat Virtualization Manager that manages RHHI for Virtualization.

Hosted Engine virtual machine

The virtual machine that acts as Red Hat Virtualization Manager. The Hosted Engine virtual machine runs on a virtualization host that is managed by the instance of Red Hat Virtualization Manager that is running on the Hosted Engine virtual machine.

Manager node

A virtualization host that runs Red Hat Virtualization Manager directly, rather than running it in a Hosted Engine virtual machine.

Red Hat Enterprise Linux host

A physical machine installed with Red Hat Enterprise Linux plus additional packages to provide the same capabilities as a Red Hat Virtualization host. This type of host is not supported for use with RHHI for Virtualization.

Red Hat Virtualization

An operating system and management interface for virtualizing resources, processes, and applications for Linux and Microsoft Windows workloads.

Red Hat Virtualization host

A physical machine installed with Red Hat Virtualization that provides the physical resources to support the virtualization of resources, processes, and applications for Linux and Microsoft Windows workloads. This is the only type of host supported with RHHI for Virtualization.

Red Hat Virtualization Manager

A server that runs the management and monitoring capabilities of Red Hat Virtualization.

Self-Hosted Engine node

A virtualization host that contains the Hosted Engine virtual machine. All hosts in a RHHI for Virtualization deployment are capable of becoming Self-Hosted Engine nodes, but there is only one Self-Hosted Engine node at a time.

storage domain

A named collection of images, templates, snapshots, and metadata. A storage domain can be comprised of block devices or file systems. Storage domains are attached to data centers in order to provide access to the collection of images, templates, and so on to hosts in the data center.

virtualization host

A physical machine with the ability to virtualize physical resources, processes, and applications for client access.

VM Portal

A web user interface provided by Red Hat Virtualization Manager. It allows users to manage and monitor virtual machines.

D.2. STORAGE TERMS

brick

An exported directory on a server in a trusted storage pool.

cache logical volume

A small, fast logical volume used to improve the performance of a large, slow logical volume.

geo-replication

One way asynchronous replication of data from a source Gluster volume to a target volume. Geo-replication works across local and wide area networks as well as the Internet. The target volume can be a Gluster volume in a different trusted storage pool, or another type of storage.

gluster volume

A logical group of bricks that can be configured to distribute, replicate, or disperse data according to workload requirements.

logical volume management (LVM)

A method of combining physical disks into larger virtual partitions. Physical volumes are placed in volume groups to form a pool of storage that can be divided into logical volumes as needed.

Red Hat Gluster Storage

An operating system based on Red Hat Enterprise Linux with additional packages that provide support for distributed, software-defined storage.

source volume

The Gluster volume that data is being copied from during geo-replication.

storage host

A physical machine that provides storage for client access.

target volume

The Gluster volume or other storage volume that data is being copied to during geo-replication.

thin provisioning

Provisioning storage such that only the space that is required is allocated at creation time, with further space being allocated dynamically according to need over time.

thick provisioning

Provisioning storage such that all space is allocated at creation time, regardless of whether that space is required immediately.

trusted storage pool

A group of Red Hat Gluster Storage servers that recognise each other as trusted peers.

D.3. HYPERCONVERGED INFRASTRUCTURE TERMS

Red Hat Hyperconverged Infrastructure (RHHI) for Virtualization

RHHI for Virtualization is a single product that provides both virtual compute and virtual storage resources. Red Hat Virtualization and Red Hat Gluster Storage are installed in a converged configuration, where the services of both products are available on each physical machine in a cluster.

hyperconverged host

A physical machine that provides physical storage, which is virtualized and consumed by virtualized processes and applications run on the same host. All hosts installed with RHHI for Virtualization are hyperconverged hosts.

Web Console

The web user interface for deploying, managing, and monitoring RHHI for Virtualization. The Web Console is provided by the Web Console service and plugins for Red Hat Virtualization Manager.