



# **Red Hat Virtualization 4.0 Technical Reference**

---

The Technical Architecture of Red Hat Virtualization Environments

Red Hat Virtualization Documentation Team



## The Technical Architecture of Red Hat Virtualization Environments

Red Hat Virtualization Documentation Team  
Red Hat Customer Content Services  
[rhev-docs@redhat.com](mailto:rhev-docs@redhat.com)

## Legal Notice

Copyright © 2017 Red Hat.

This document is licensed by Red Hat under the [Creative Commons Attribution-ShareAlike 3.0 Unported License](#). If you distribute this document, or a modified version of it, you must provide attribution to Red Hat, Inc. and provide a link to the original. If the document is modified, all Red Hat trademarks must be removed.

Red Hat, as the licensor of this document, waives the right to enforce, and agrees not to assert, Section 4d of CC-BY-SA to the fullest extent permitted by applicable law.

Red Hat, Red Hat Enterprise Linux, the Shadowman logo, JBoss, OpenShift, Fedora, the Infinity logo, and RHCE are trademarks of Red Hat, Inc., registered in the United States and other countries.

Linux ® is the registered trademark of Linus Torvalds in the United States and other countries.

Java ® is a registered trademark of Oracle and/or its affiliates.

XFS ® is a trademark of Silicon Graphics International Corp. or its subsidiaries in the United States and/or other countries.

MySQL ® is a registered trademark of MySQL AB in the United States, the European Union and other countries.

Node.js ® is an official trademark of Joyent. Red Hat Software Collections is not formally related to or endorsed by the official Joyent Node.js open source or commercial project.

The OpenStack ® Word Mark and OpenStack logo are either registered trademarks/service marks or trademarks/service marks of the OpenStack Foundation, in the United States and other countries and are used with the OpenStack Foundation's permission. We are not affiliated with, endorsed or sponsored by the OpenStack Foundation, or the OpenStack community.

All other trademarks are the property of their respective owners.

## Abstract

This reference documents the concepts, components, and technologies used in a Red Hat Virtualization environment.

# Table of Contents

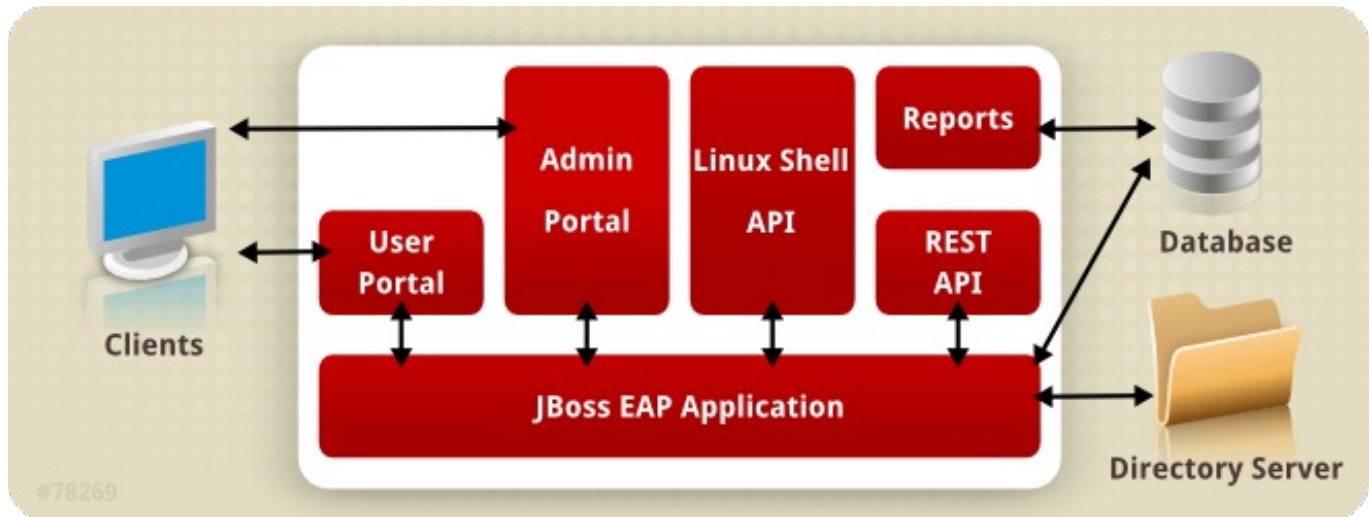
<b>Chapter 1. Introduction</b>	<b>3</b>
1.1. Red Hat Virtualization Manager	3
1.2. Red Hat Virtualization Host	3
1.3. Interfaces for Accessing the Manager	5
1.4. Components that Support the Manager	6
1.5. Storage	7
1.6. Network	8
1.7. Data Centers	11
<b>Chapter 2. Storage</b>	<b>12</b>
2.1. Storage Domains Overview	12
2.2. Types of Storage Backing Storage Domains	12
2.3. Storage Domain Types	13
2.4. Storage Formats for Virtual Disk Images	13
2.5. Virtual Disk Image Storage Allocation Policies	13
2.6. Storage Metadata Versions in Red Hat Virtualization	14
2.7. Storage Domain Autorecovery in Red Hat Virtualization	15
2.8. The Storage Pool Manager	15
2.9. Storage Pool Manager Selection Process	16
2.10. Exclusive Resources and Sanlock in Red Hat Virtualization	17
2.11. Thin Provisioning and Storage Over-Commitment	18
2.12. Logical Volume Extension	18
<b>Chapter 3. Network</b>	<b>20</b>
3.1. Network Architecture	20
3.2. Introduction: Basic Networking Terms	20
3.3. Network Interface Controller	20
3.4. Bridge	20
3.5. Bonds	21
3.6. Switch Configuration for Bonding	22
3.7. Virtual Network Interface Cards	22
3.8. Virtual LAN (VLAN)	24
3.9. Network Labels	24
3.10. Cluster Networking	25
3.11. Logical Networks	26
3.12. Required Networks, Optional Networks, and Virtual Machine Networks	28
3.13. Virtual Machine Connectivity	28
3.14. Port Mirroring	29
3.15. Host Networking Configurations	29
3.16. Bridge Configuration	29
3.17. VLAN Configuration	30
3.18. Bridge and Bond Configuration	31
3.19. Multiple Bridge, Multiple VLAN, and NIC Configuration	31
3.20. Multiple Bridge, Multiple VLAN, and Bond Configuration	32
<b>Chapter 4. Power Management</b>	<b>33</b>
4.1. Introduction to Power Management and Fencing	33
4.2. Power Management by Proxy in Red Hat Virtualization	33
4.3. Power Management	33
4.4. Fencing	34
4.5. Soft-Fencing Hosts	35
4.6. Using Multiple Power Management Fencing Agents	36

<b>Chapter 5. Load Balancing, Scheduling, and Migration</b> .....	<b>37</b>
5.1. Load Balancing, Scheduling, and Migration	37
5.2. Load Balancing Policy	37
5.3. Load Balancing Policy: VM_Evenly_Distributed	37
5.4. Load Balancing Policy: Evenly_Distributed	38
5.5. Load Balancing Policy: Power_Saving	38
5.6. Load Balancing Policy: None	39
5.7. Load Balancing Policy: InClusterUpgrade	39
5.8. Highly Available Virtual Machine Reservation	39
5.9. Scheduling	40
5.10. Migration	40
<b>Chapter 6. Directory Services</b> .....	<b>41</b>
6.1. Directory Services	41
6.2. Local Authentication: Internal Domain	41
6.3. Remote Authentication Using GSSAPI	41
<b>Chapter 7. Templates and Pools</b> .....	<b>43</b>
7.1. Templates and Pools	43
7.2. Templates	43
7.3. Pools	44
<b>Chapter 8. Virtual Machine Snapshots</b> .....	<b>45</b>
8.1. Snapshots	45
8.2. Live Snapshots in Red Hat Virtualization	45
8.3. Snapshot Creation	46
8.4. Snapshot Previews	47
8.5. Snapshot Deletion	48
<b>Chapter 9. Hardware Drivers and Devices</b> .....	<b>50</b>
9.1. Virtualized Hardware	50
9.2. Stable Device Addresses in Red Hat Virtualization	50
9.3. Central Processing Unit (CPU)	50
9.4. System Devices	51
9.5. Network Devices	51
9.6. Graphics Devices	51
9.7. Storage Devices	51
9.8. Sound Devices	52
9.9. Serial Driver	52
9.10. Balloon Driver	52
<b>Chapter 10. Minimum Requirements and Technical Limitations</b> .....	<b>53</b>
10.1. Minimum Requirements and Supported Limits	53
10.2. Resource Limitations	53
10.3. Cluster Limitations	53
10.4. Storage Domain Limitations	53
10.5. Red Hat Virtualization Manager Limitations	54
10.6. Hypervisor Requirements	55
10.7. Guest Requirements and Support Limits	57
10.8. SPICE Limitations	58
10.9. Additional References	58

## Chapter 1. Introduction

### 1.1. Red Hat Virtualization Manager

The Red Hat Virtualization Manager provides centralized management for a virtualized environment. A number of different interfaces can be used to access the Red Hat Virtualization Manager. Each interface facilitates access to the virtualized environment in a different manner.



**Figure 1.1. Red Hat Virtualization Manager Architecture**

The Red Hat Virtualization Manager provides graphical interfaces and an *Application Programming Interface* (API). Each interface connects to the Manager, an application delivered by an embedded instance of the *Red Hat JBoss Enterprise Application Platform*. There are a number of other components which support the Red Hat Virtualization Manager in addition to Red Hat JBoss Enterprise Application Platform.

### 1.2. Red Hat Virtualization Host

A Red Hat Virtualization environment has one or more hosts attached to it. A host is a server that provides the physical hardware that virtual machines make use of.

Red Hat Virtualization Host (RHVH) runs an optimized operating system installed using a special, customized installation media specifically for creating virtualization hosts.

Red Hat Enterprise Linux hosts are servers running a standard Red Hat Enterprise Linux operating system that has been configured after installation to permit use as a host.

Both methods of host installation result in hosts that interact with the rest of the virtualized environment in the same way, and so, will both be referred to as *hosts*.

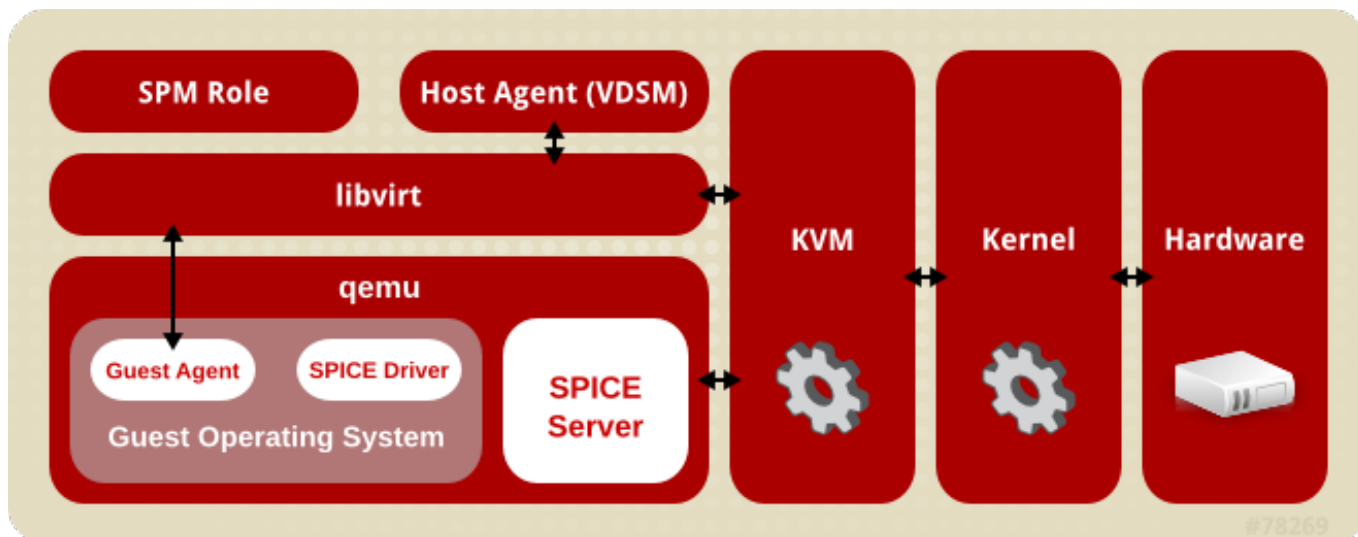


Figure 1.2. Host Architecture

### Kernel-based Virtual Machine (KVM)

The Kernel-based Virtual Machine (KVM) is a loadable kernel module that provides full virtualization through the use of the Intel VT or AMD-V hardware extensions. Though KVM itself runs in kernel space, the guests running upon it run as individual *QEMU* processes in user space. KVM allows a host to make its physical hardware available to virtual machines.

### QEMU

QEMU is a multi-platform emulator used to provide full system emulation. QEMU emulates a full system, for example a PC, including one or more processors, and peripherals. QEMU can be used to launch different operating systems or to debug system code. QEMU, working in conjunction with KVM and a processor with appropriate virtualization extensions, provides full hardware assisted virtualization.

### Red Hat Virtualization Manager Host Agent, VDSM

In Red Hat Virtualization, **VDSM** initiates actions on virtual machines and storage. It also facilitates inter-host communication. VDSM monitors host resources such as memory, storage, and networking. Additionally, VDSM manages tasks such as virtual machine creation, statistics accumulation, and log collection. A VDSM instance runs on each host and receives management commands from the Red Hat Virtualization Manager using the re-configurable port **54321**.

#### VDSM-REG

**VDSM** uses **VDSM-REG** to register each host with the Red Hat Virtualization Manager. **VDSM-REG** supplies information about itself and its host using port **80** or port **443**.

### libvirt

Libvirt facilitates the management of virtual machines and their associated virtual devices. When Red Hat Virtualization Manager initiates virtual machine life-cycle commands (start, stop, reboot), VDSM invokes libvirt on the relevant host machines to execute them.

### Storage Pool Manager, SPM

The Storage Pool Manager (SPM) is a role assigned to one host in a data center. The SPM host has sole authority to make all storage domain structure metadata changes for the data center. This includes creation, deletion, and manipulation of virtual disk images, snapshots, and templates. It



also includes allocation of storage for sparse block devices on a *Storage Area Network*(SAN). The role of SPM can be migrated to any host in a data center. As a result, all hosts in a data center must have access to all the storage domains defined in the data center.

Red Hat Virtualization Manager ensures that the SPM is always available. In case of storage connectivity errors, the Manager re-assigns the SPM role to another host.

## Guest Operating System

Guest operating systems can be installed without modification on virtual machines in a Red Hat Virtualization environment. The guest operating system, and any applications on the guest, are unaware of the virtualized environment and run normally.

Red Hat provides enhanced device drivers that allow faster and more efficient access to virtualized devices. You can also install the Red Hat Virtualization Guest Agent on guests, which provides enhanced guest information to the management console.

## 1.3. Interfaces for Accessing the Manager

### User Portal

Desktop virtualization provides users with a desktop environment that is similar a personal computer's desktop environment. The User Portal is for delivering *Virtual Desktop Infrastructure* to users. Users access the User Portal through a web browser to display and access their assigned virtual desktops. The actions available to a user in the User Portal are set by a system administrator. Standard users can start, stop, and use desktops that are assigned to them by the system administrator. Power users can perform some administrative actions. Both types of user access the User Portal from the same URL, and are presented with options appropriate to their permission level on login.

#### ➤ Standard User Access

Standard users are able to power their virtual desktops on and off and connect to them through the User Portal. Direct connection to virtual machines is facilitated with *Simple Protocol for Independent Computing Environments (SPICE)* or *Virtual Network Computing (VNC)* clients. Both protocols provide the user with an environment similar to a locally installed desktop environment. The administrator specifies the protocol used to connect to a virtual machine at the time of the virtual machine's creation.

More information on the actions available from the User Portal as well as supported browsers and clients can be found in the [Introduction to the User Portal](#).

#### ➤ Power User Access

The Red Hat Virtualization User Portal provides power users with a graphical user interface to create, use, and monitor virtual resources. System administrators can delegate some administration tasks by granting users power user access. In addition to the tasks that can be performed by standard users, power users can:

- Create, edit, and remove virtual machines.
- Manage virtual disks and network interfaces.
- Assign user permissions to virtual machines.
- Create and use templates to rapidly deploy virtual machines.
- Monitor resource usage and high-severity events.

- Create and use snapshots to restore virtual machines to previous states.

Power users can perform virtual machine administration tasks to be delegated. Data center and cluster level administration tasks are saved for the environment administrator.

## Administration Portal

The Administration Portal is the graphical administration interface of the Red Hat Virtualization Manager server. Using it administrators can monitor, create, and maintain all elements of the virtualized environment using from web browsers. Tasks which can be performed from the Administration Portal include:

- Creation and management of virtual infrastructure (networks, storage domains).
- Installation and management of hosts.
- Creation and management of logical entities (data centers, clusters).
- Creation and management of virtual machines.
- Red Hat Virtualization user and permission management.

The Administration Portal is displayed using the JavaScript.

Administration Portal functions are discussed in further detail in the [Red Hat Virtualization Administration Guide](#). Information on the browsers and platforms that are supported by the Administration Portal can be found in the [Red Hat Virtualization Installation Guide](#).

## Representational State Transfer (REST) API

The Red Hat Virtualization REST API provides a software interface for the interrogation and control of the Red Hat Virtualization environment. The REST API can be used by any programming language that supports HTTP actions.

Using the REST API developers and administrators can:

- Integrate with enterprise IT systems.
- Integrate with third party virtualization software.
- Perform automated maintenance and error checking tasks.
- Use scripts to automate repetitive tasks in a Red Hat Virtualization environment.

See the [Red Hat Virtualization REST API Guide](#) for the API specification and usage examples.

## 1.4. Components that Support the Manager

### Red Hat JBoss Enterprise Application Platform

Red Hat JBoss Enterprise Application Platform is a Java application server. It provides a framework to support efficient development and delivery of cross-platform Java applications. The Red Hat Virtualization Manager is delivered using Red Hat JBoss Enterprise Application Platform.



### Important

The version of the Red Hat JBoss Enterprise Application Platform bundled with Red Hat Virtualization Manager is **not** to be used to serve other applications. It has been customized for the specific purpose of serving the Red Hat Virtualization Manager. Using the Red Hat JBoss Enterprise Application Platform that is included with the Manager for additional purposes adversely affects its ability to service the Red Hat Virtualization environment.

## Gathering Reports and Historical Data

The Red Hat Virtualization Manager includes a data warehouse that collects monitoring data about hosts, virtual machines, and storage. A number of pre-defined reports are available. Customers can analyze their environments and create reports using any query tools that support SQL.

The Red Hat Virtualization Manager installation process creates two databases. These databases are created on a Postgres instance which is selected during installation.

- The engine database is the primary data store used by the Red Hat Virtualization Manager. Information about the virtualization environment like its state, configuration, and performance are stored in this database.
- The `ovirt_engine_history` database contains configuration information and statistical metrics which are collated over time from the engine operational database. The configuration data in the engine database is examined every minute, and changes are replicated to the `ovirt_engine_history` database. Tracking the changes to the database provides information on the objects in the database. This enables you to analyze and enhance the performance of your Red Hat Virtualization environment and resolve difficulties.

For more information on generating reports based on the `ovirt_engine_history` database see the [History Database](#) in the *Red Hat Virtualization Data Warehouse Guide*.



### Important

The replication of data in the `ovirt_engine_history` database is performed by the **RHEVM History Service**, `ovirt-engine-dwhd`.

## Directory services

Directory services provide centralized network-based storage of user and organizational information. Types of information stored include application settings, user profiles, group data, policies, and access control. The Red Hat Virtualization Manager supports Active Directory, Identity Management (IdM), OpenLDAP, and Red Hat Directory Server 9. There is also a local, internal domain for administration purposes only. This internal domain has only one user: the admin user.

## 1.5. Storage

Red Hat Virtualization uses a centralized storage system for virtual disk images, templates, snapshots, and ISO files. Storage is logically grouped into storage pools, which are comprised of storage domains. A storage domain is a combination of storage capacity and metadata that describes the internal structure of the storage. There are three types of storage domain; data, export, and ISO.

The data storage domain is the only one required by each data center. A data storage domain is exclusive to a single data center. Export and ISO domains are optional. Storage domains are shared resources, and must be accessible to all hosts in a data center.

Storage networking can be implemented using Network File System (NFS), Internet Small Computer System Interface (iSCSI), GlusterFS, Fibre Channel Protocol (FCP), or any POSIX compliant networked filesystem.

On NFS (and other POSIX compliant filesystems) domains, all virtual disks, templates, and snapshots are simple files.

On SAN (iSCSI/FCP) domains, block devices are aggregated by Logical Volume Manager (LVM) into a Volume Group (VG). Each virtual disk, template and snapshot is a Logical Volume (LV) on the VG. See the [Red Hat Enterprise Linux Logical Volume Manager Administration Guide](#) for more information on LVM.

### **Data storage domain**

Data domains hold the virtual hard disk images of all the virtual machines running in the environment. Templates and snapshots of the virtual machines are also stored in the data domain. A data domain cannot be shared across data centers.

### **Export storage domain**

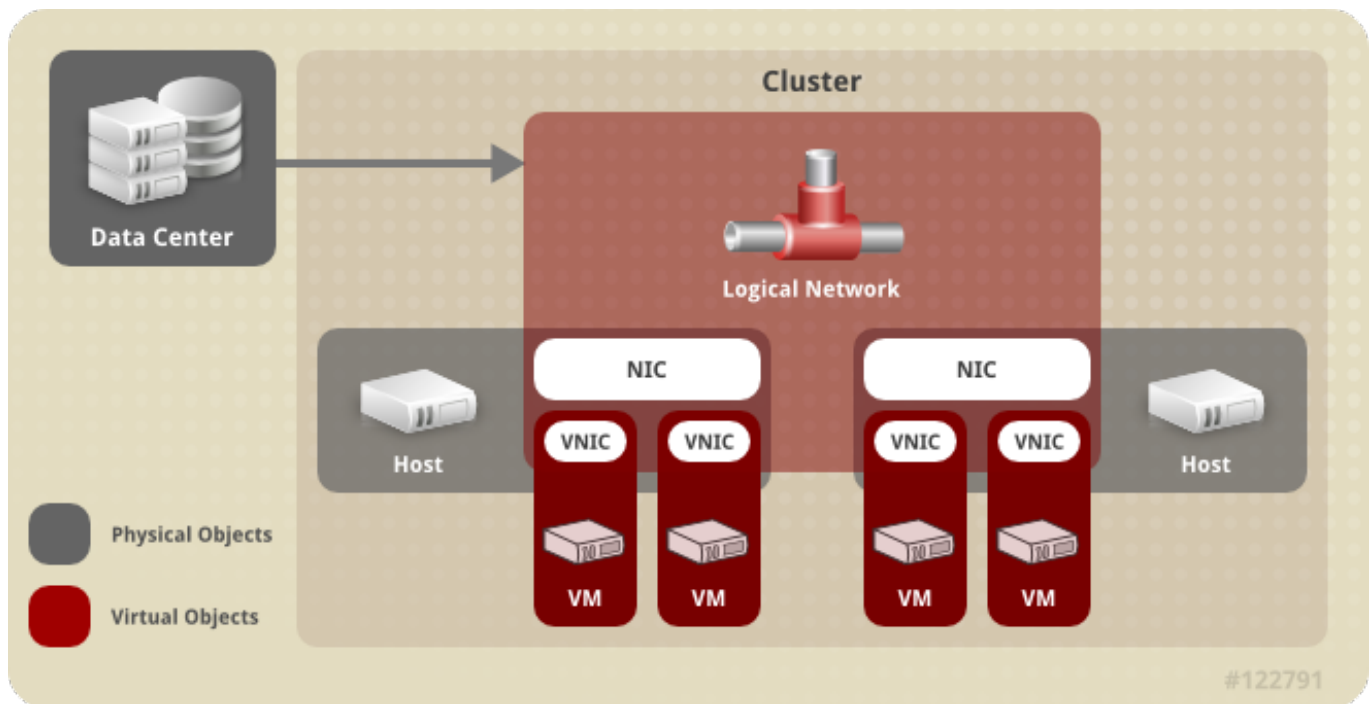
An export domain is a temporary storage repository that is used to copy and move images between data centers and Red Hat Virtualization environments. The export domain can be used to back up virtual machines and templates. An export domain can be moved between data centers, but can only be active in one data center at a time.

### **ISO storage domain**

ISO domains store ISO files, which are logical CD-ROMs used to install operating systems and applications for the virtual machines. As a logical entity that replaces a library of physical CD-ROMs or DVDs, an ISO domain removes the data center's need for physical media. An ISO domain can be shared across different data centers.

## **1.6. Network**

The Red Hat Virtualization network architecture facilitates connectivity between the different elements of the Red Hat Virtualization environment. The network architecture not only supports network connectivity, it also allows for network segregation.



**Figure 1.3. Network Architecture**

Networking is defined in Red Hat Virtualization in several layers. The underlying physical networking infrastructure must be in place and configured to allow connectivity between the hardware and the logical components of the Red Hat Virtualization environment.

### Networking Infrastructure Layer

The Red Hat Virtualization network architecture relies on some common hardware and software devices:

- *Network Interface Controllers (NICs)* are physical network interface devices that connect a host to the network.
- *Virtual NICs (VNICs)* are logical NICs that operate using the host's physical NICs. They provide network connectivity to virtual machines.
- *Bonds* bind multiple NICs into a single interface.
- *Bridges* are a packet-forwarding technique for packet-switching networks. They form the basis of *virtual machine logical networks*.

### Logical Networks

Logical networks allow segregation of network traffic based on environment requirements. The types of logical network are:

- logical networks that carry virtual machine network traffic,
- logical networks that do not carry virtual machine network traffic,
- optional logical networks,
- and required networks.

All logical networks can either be required or optional.

A logical network that carries virtual machine network traffic is implemented at the host level as a software bridge device. By default, one logical network is defined during the installation of the Red Hat Virtualization Manager: the **ovirtmgmt** management network.

Other logical networks that can be added by an administrator are: a dedicated storage logical network, and a dedicated display logical network. Logical networks that do not carry virtual machine traffic do not have an associated bridge device on hosts. They are associated with host network interfaces directly.

Red Hat Virtualization segregates management-related network traffic from migration-related network traffic. This makes it possible to use a dedicated network (without routing) for live migration, and ensures that the management network (ovirtmgmt) does not lose its connection to hypervisors during migrations.

## Explanation of logical networks on different layers

Logical networks have different implications for each layer of the virtualization environment.

### Data Center Layer

Logical networks are defined at the data center level. Each data center has the **ovirtmgmt** management network by default. Further logical networks are optional but recommended. Designation as a **VM Network** and a custom MTU can be set at the data center level. A logical network that is defined for a data center must also be added to the clusters that use the logical network.

### Cluster Layer

Logical networks are made available from a data center, and must be added to the clusters that will use them. Each cluster is connected to the management network by default. You can optionally add to a cluster logical networks that have been defined for the cluster's parent data center. When a required logical network has been added to a cluster, it must be implemented for each host in the cluster. Optional logical networks can be added to hosts as needed.

### Host Layer

Virtual machine logical networks are implemented for each host in a cluster as a software bridge device associated with a given network interface. Non-virtual machine logical networks do not have associated bridges, and are associated with host network interfaces directly. Each host has the management network implemented as a bridge using one of its network devices as a result of being included in a Red Hat Virtualization environment. Further required logical networks that have been added to a cluster must be associated with network interfaces on each host to become operational for the cluster.

### Virtual Machine Layer

Logical networks can be made available to virtual machines in the same way that a network can be made available to a physical machine. A virtual machine can have its virtual NIC connected to any virtual machine logical network that has been implemented on the host that runs it. The virtual machine then gains connectivity to any other devices or destinations that are available on the logical network it is connected to.

---

#### Example 1.1. Management Network

The management logical network, named **ovirtmgmt**, is created automatically when the Red Hat Virtualization Manager is installed. The **ovirtmgmt** network is dedicated to management traffic between the Red Hat Virtualization Manager and hosts. If no other specifically-purposed bridges are set up, **ovirtmgmt** is the default bridge for all traffic.

## 1.7. Data Centers

A data center is the highest level of abstraction in Red Hat Virtualization. A data center is a container that is comprised of three types of sub-containers:

- ✦ The *storage container* holds information about storage types and storage domains, including connectivity information for storage domains. Storage is defined for a data center, and available to all clusters in the data center. All host clusters within a data center have access to the same storage domains.
- ✦ The *network container* holds information about the data center's logical networks. This includes details such as network addresses, VLAN tags and STP support. Logical networks are defined for a data center, and are optionally implemented at the cluster level.
- ✦ The *cluster container* holds clusters. Clusters are groups of hosts with compatible processor cores, either AMD or Intel processors. Clusters are migration domains; virtual machines can be live-migrated to any host within a cluster, and not to other clusters. One data center can hold multiple clusters, and each cluster can contain multiple hosts.

## Chapter 2. Storage

### 2.1. Storage Domains Overview

A storage domain is a collection of images that have a common storage interface. A storage domain contains complete images of templates and virtual machines (including snapshots), ISO files, and metadata about themselves. A storage domain can be made of either block devices (SAN - iSCSI or FCP) or a file system (NAS - NFS, GlusterFS, or other POSIX compliant file systems).

On NAS, all virtual disks, templates, and snapshots are files.

On SAN (iSCSI/FCP), each virtual disk, template or snapshot is a logical volume. Block devices are aggregated into a logical entity called a volume group, and then divided by LVM (Logical Volume Manager) into logical volumes for use as virtual hard disks. See the [Red Hat Enterprise Linux Logical Volume Manager Administration Guide](#) for more information on LVM.

Virtual disks can have one of two formats, either QCOW2 or RAW. The type of storage can be either Sparse or Preallocated. Snapshots are always sparse but can be taken for disks created either as RAW or sparse.

Virtual machines that share the same storage domain can be migrated between hosts that belong to the same cluster.

### 2.2. Types of Storage Backing Storage Domains

Storage domains can be implemented using block based and file based storage.

#### File Based Storage

The file based storage types supported by Red Hat Virtualization are NFS, GlusterFS, other POSIX compliant file systems, and storage local to hosts.

File based storage is managed externally to the Red Hat Virtualization environment.

NFS storage is managed by a Red Hat Enterprise Linux NFS server, or other third party network attached storage server.

Hosts can manage their own local storage file systems.

#### Block Based Storage

Block storage uses unformatted block devices. Block devices are aggregated into *volume groups* by the Logical Volume Manager (LVM). An instance of LVM runs on all hosts, unaware of the instances running on other hosts. VDSM adds clustering logic on top of LVM by scanning volume groups for changes. When changes are detected, VDSM updates individual hosts by telling them to refresh their volume group information. The hosts divide the volume group into logical volumes, writing logical volume metadata to disk. If more storage capacity is added to an existing storage domain, the Red Hat Virtualization Manager causes VDSM on each host to refresh volume group information.

A Logical Unit Number (LUN) is an individual block device. One of the supported block storage protocols, iSCSI, FCoE, or SAS, is used to connect to a LUN. The Red Hat Virtualization Manager manages software iSCSI connections to the LUNs. All other block storage connections are managed externally to the Red Hat Virtualization environment. Any changes in a block based



storage environment, such as the creation of logical volumes, extension or deletion of logical volumes and the addition of a new LUN are handled by LVM on a specially selected host called the Storage Pool Manager. Changes are then synced by VDSM which storage metadata refreshes across all hosts in the cluster.

## 2.3. Storage Domain Types

Red Hat Virtualization supports three types of storage domains, including the storage types that each of the storage domains support:

- ✦ The *Data Storage Domain* stores the hard disk images of all virtual machines in the Red Hat Virtualization environment. Disk images may contain an installed operating system or data stored or generated by a virtual machine. Data storage domains support NFS, iSCSI, FCP, GlusterFS and POSIX compliant storage. A data domain cannot be shared between multiple data centers.
- ✦ The *Export Storage Domain* provides transitory storage for hard disk images and virtual machine templates being transferred between data centers. Additionally, export storage domains store backed up copies of virtual machines. Export storage domains support NFS storage. Multiple data centers can access a single export storage domain but only one data center can use it at a time.
- ✦ The *ISO Storage Domain* stores ISO files, also called images. ISO files are representations of physical CDs or DVDs. In the Red Hat Virtualization environment the common types of ISO files are operating system installation disks, application installation disks, and guest agent installation disks. These images can be attached to virtual machines and booted in the same way that physical disks are inserted into a disk drive and booted. ISO storage domains allow all hosts within the data center to share ISOs, eliminating the need for physical optical media.

## 2.4. Storage Formats for Virtual Disk Images

### QCOW2 Formatted Virtual Machine Storage

QCOW2 is a storage format for virtual disk images. QCOW stands for *QEMU copy on write*. The QCOW2 format decouples the physical storage layer from the virtual layer by adding a mapping between logical and physical blocks. Each logical block is mapped to its physical offset, which enables storage over-commitment and virtual machine snapshots, where each QCOW volume only represents changes made to an underlying disk image.

The initial mapping points all logical blocks to the offsets in the backing file or volume. When a virtual machine writes data to a QCOW2 volume after a snapshot, the relevant block is read from the backing volume, modified with the new information and written into a new snapshot QCOW2 volume. Then the map is updated to point to the new place.

### RAW

The RAW storage format has a performance advantage over QCOW2 in that no formatting is applied to virtual disk images stored in the RAW format. Virtual machine data operations on disk images stored in RAW format require no additional work from hosts. When a virtual machine writes data to a given offset in its virtual disk, the I/O is written to the same offset on the backing file or logical volume.

Raw format requires that the entire space of the defined image be preallocated unless using externally managed thin provisioned LUNs from a storage array.

## 2.5. Virtual Disk Image Storage Allocation Policies

## Preallocated Storage

All of the storage required for a virtual disk image is allocated prior to virtual machine creation. If a 20 GB disk image is created for a virtual machine, the disk image uses 20 GB of storage domain capacity. Preallocated disk images cannot be enlarged. Preallocating storage can mean faster write times because no storage allocation takes place during runtime, at the cost of flexibility. Allocating storage this way reduces the capacity of the Red Hat Virtualization Manager to overcommit storage. Preallocated storage is recommended for virtual machines used for high intensity I/O tasks with less tolerance for latency in storage. Generally, server virtual machines fit this description.



### Note

If thin provisioning functionality provided by your storage back-end is being used, preallocated storage should still be selected from the Administration Portal when provisioning storage for virtual machines.

## Sparsely Allocated Storage

The upper size limit for a virtual disk image is set at virtual machine creation time. Initially, the disk image does not use any storage domain capacity. Usage grows as the virtual machine writes data to disk, until the upper limit is reached. Capacity is not returned to the storage domain when data in the disk image is removed. Sparsely allocated storage is appropriate for virtual machines with low or medium intensity I/O tasks with some tolerance for latency in storage. Generally, desktop virtual machines fit this description.



### Note

If thin provisioning functionality is provided by your storage back-end, it should be used as the preferred implementation of thin provisioning. Storage should be provisioned from the graphical user interface as preallocated, leaving thin provisioning to the back-end solution.

## 2.6. Storage Metadata Versions in Red Hat Virtualization

Red Hat Virtualization stores information about storage domains as metadata on the storage domains themselves. Each major release of Red Hat Virtualization has seen improved implementations of storage metadata.

### ✱ V1 metadata (Red Hat Virtualization 2.x series)

Each storage domain contains metadata describing its own structure, and all of the names of physical volumes that are used to back virtual disk images.

Master domains additionally contain metadata for all the domains and physical volume names in the storage pool. The total size of this metadata is limited to 2 KB, limiting the number of storage domains that can be in a pool.

Template and virtual machine base images are read only.

V1 metadata is applicable to NFS, iSCSI, and FC storage domains.

### ✱ V2 metadata (Red Hat Enterprise Virtualization 3.0)

All storage domain and pool metadata is stored as logical volume tags rather than written to a logical volume. Metadata about virtual disk volumes is still stored in a logical volume on the domains.

Physical volume names are no longer included in the metadata.

Template and virtual machine base images are read only.

V2 metadata is applicable to iSCSI, and FC storage domains.

#### ✱ *V3 metadata (Red Hat Enterprise Virtualization 3.1+)*

All storage domain and pool metadata is stored as logical volume tags rather than written to a logical volume. Metadata about virtual disk volumes is still stored in a logical volume on the domains.

Virtual machine and template base images are no longer read only. This change enables live snapshots, live storage migration, and clone from snapshot.

Support for unicode metadata is added, for non-English volume names.

V3 metadata is applicable to NFS, GlusterFS, POSIX, iSCSI, and FC storage domains.

## 2.7. Storage Domain Autorecovery in Red Hat Virtualization

Hosts in a Red Hat Virtualization environment monitor storage domains in their data centers by reading metadata from each domain. A storage domain becomes inactive when all hosts in a data center report that they cannot access the storage domain.

Rather than disconnecting an inactive storage domain, the Manager assumes that the storage domain has become inactive temporarily, because of a temporary network outage for example. Once every 5 minutes, the Manager attempts to re-activate any inactive storage domains.

Administrator intervention may be required to remedy the cause of the storage connectivity interruption, but the Manager handles re-activating storage domains as connectivity is restored.

## 2.8. The Storage Pool Manager

Red Hat Virtualization uses metadata to describe the internal structure of storage domains. Structural metadata is written to a segment of each storage domain. Hosts work with the storage domain metadata based on a single writer, and multiple readers configuration. Storage domain structural metadata tracks image and snapshot creation and deletion, and volume and domain extension.

The host that can make changes to the structure of the data domain is known as the Storage Pool Manager (SPM). The SPM coordinates all metadata changes in the data center, such as creating and deleting disk images, creating and merging snapshots, copying images between storage domains, creating templates and storage allocation for block devices. There is one SPM for every data center. All other hosts can only read storage domain structural metadata.

A host can be manually selected as the SPM, or it can be assigned by the Red Hat Virtualization Manager. The Manager assigns the SPM role by causing a potential SPM host to attempt to assume a *storage-centric lease*. The lease allows the SPM host to write storage metadata. It is storage-centric because it is written to the storage domain rather than being tracked by the Manager or hosts. Storage-centric leases are written to a special logical volume in the master storage domain called **leases**. Metadata about the structure of the data domain is written to a special logical volume called **metadata**. Changes to the **metadata** logical volume are protected against by the **leases** logical volume.

The Manager uses VDSM to issue the **spmStart** command to a host, causing VDSM on that host to attempt to assume the storage-centric lease. If the host is successful it becomes the SPM and retains the storage-centric lease until the Red Hat Virtualization Manager requests that a new host assume the role of SPM.

The Manager moves the SPM role to another host if:

- the SPM host can not access all storage domains, but can access the master storage domain.
- the SPM host is unable to renew the lease because of a loss of storage connectivity or the lease volume is full and no write operation can be performed.
- the SPM host crashes.

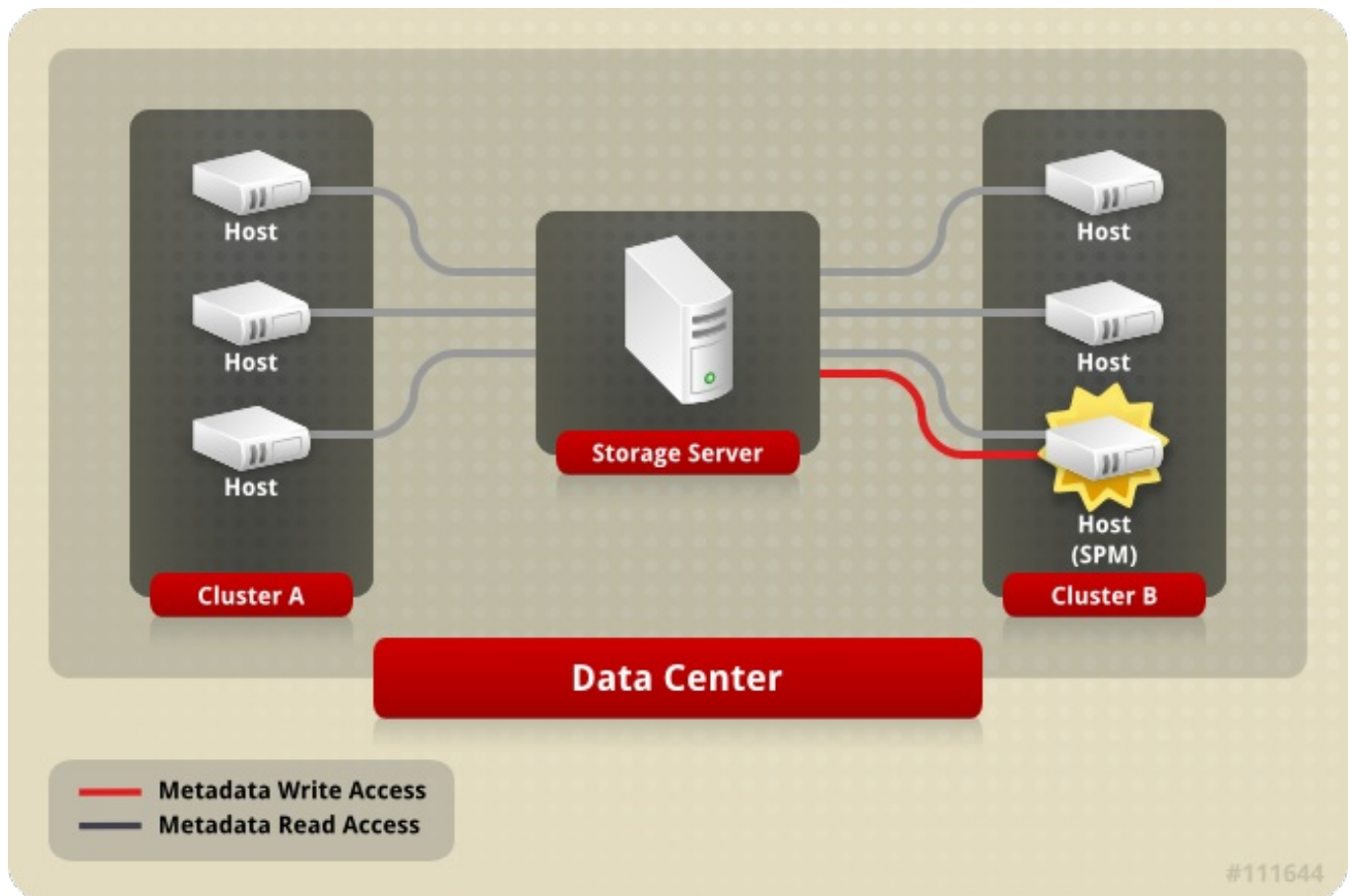


Figure 2.1. The Storage Pool Manager Exclusively Writes Structural Metadata.

## 2.9. Storage Pool Manager Selection Process

If a host has not been manually assigned the Storage Pool Manager (SPM) role, the SPM selection process is initiated and managed by the Red Hat Virtualization Manager.

First, the Red Hat Virtualization Manager requests that VDSM confirm which host has the storage-centric lease.

The Red Hat Virtualization Manager tracks the history of SPM assignment from the initial creation of a storage domain onward. The availability of the SPM role is confirmed in three ways:

- The "getSPMstatus" command: the Manager uses VDSM to check with the host that had SPM status last and receives one of "SPM", "Contending", or "Free".
- The metadata volume for a storage domain contains the last host with SPM status.

- ✱ The metadata volume for a storage domain contains the version of the last host with SPM status.

If an operational, responsive host retains the storage-centric lease, the Red Hat Virtualization Manager marks that host SPM in the administrator portal. No further action is taken.

If the SPM host does not respond, it is considered unreachable. If power management has been configured for the host, it is automatically fenced. If not, it requires manual fencing. The Storage Pool Manager role cannot be assigned to a new host until the previous Storage Pool Manager is fenced.

When the SPM role and storage-centric lease are free, the Red Hat Virtualization Manager assigns them to a randomly selected operational host in the data center.

If the SPM role assignment fails on a new host, the Red Hat Virtualization Manager adds the host to a list containing hosts the operation has failed on, marking these hosts as ineligible for the SPM role. This list is cleared at the beginning of the next SPM selection process so that all hosts are again eligible.

The Red Hat Virtualization Manager continues request that the Storage Pool Manager role and storage-centric lease be assumed by a randomly selected host that is not on the list of failed hosts until the SPM selection succeeds.

Each time the current SPM is unresponsive or unable to fulfill its responsibilities, the Red Hat Virtualization Manager initiates the Storage Pool Manager selection process.

## 2.10. Exclusive Resources and Sanlock in Red Hat Virtualization

Certain resources in the Red Hat Virtualization environment must be accessed exclusively.

The SPM role is one such resource. If more than one host were to become the SPM, there would be a risk of data corruption as the same data could be changed from two places at once.

Prior to Red Hat Enterprise Virtualization 3.1, SPM exclusivity was maintained and tracked using a VDSM feature called *safelease*. The lease was written to a special area on all of the storage domains in a data center. All of the hosts in an environment could track SPM status in a network-independent way. The VDSM's safe lease only maintained exclusivity of one resource: the SPM role.

Sanlock provides the same functionality, but treats the SPM role as one of the resources that can be locked. Sanlock is more flexible because it allows additional resources to be locked.

Applications that require resource locking can register with Sanlock. Registered applications can then request that Sanlock lock a resource on their behalf, so that no other application can access it. For example, instead of VDSM locking the SPM status, VDSM now requests that Sanlock do so.

Locks are tracked on disk in a *lockspace*. There is one lockspace for every storage domain. In the case of the lock on the SPM resource, each host's liveness is tracked in the lockspace by the host's ability to renew the *hostid* it received from the Manager when it connected to storage, and to write a timestamp to the lockspace at a regular interval. The *ids* logical volume tracks the unique identifiers of each host, and is updated every time a host renews its *hostid*. The SPM resource can only be held by a live host.

Resources are tracked on disk in the *leases* logical volume. A resource is said to be *taken* when its representation on disk has been updated with the unique identifier of the process that has taken it. In the case of the SPM role, the SPM resource is updated with the *hostid* that has taken it.

The Sanlock process on each host only needs to check the resources once to see that they are taken. After an initial check, Sanlock can monitor the lockspaces until timestamp of the host with a locked resource becomes stale.

Sanlock monitors the applications that use resources. For example, VDSM is monitored for SPM status and `hostid`. If the host is unable to renew its `hostid` from the Manager, it loses exclusivity on all resources in the lockspace. Sanlock updates the resource to show that it is no longer taken.

If the SPM host is unable to write a timestamp to the lockspace on the storage domain for a given amount of time, the host's instance of Sanlock requests that the VDSM process release its resources. If the VDSM process responds, its resources are released, and the SPM resource in the lockspace can be taken by another host.

If VDSM on the SPM host does not respond to requests to release resources, Sanlock on the host kills the VDSM process. If the kill command is unsuccessful, Sanlock escalates by attempting to kill VDSM using `sigkill`. If the `sigkill` is unsuccessful, Sanlock depends on the *watchdog daemon* to reboot the host.

Every time VDSM on the host renews its `hostid` and writes a timestamp to the lockspace, the watchdog daemon receives a *pet*. When VDSM is unable to do so, the watchdog daemon is no longer being petted. After the watchdog daemon has not received a pet for a given amount of time, it reboots the host. This final level of escalation, if reached, guarantees that the SPM resource is released, and can be taken by another host.

## 2.11. Thin Provisioning and Storage Over-Commitment

The Red Hat Virtualization Manager provides provisioning policies to optimize storage usage within the virtualization environment. A thin provisioning policy allows you to over-commit storage resources, provisioning storage based on the actual storage usage of your virtualization environment.

Storage over-commitment is the allocation of more storage to virtual machines than is physically available in the storage pool. Generally, virtual machines use less storage than what has been allocated to them. Thin provisioning allows a virtual machine to operate as if the storage defined for it has been completely allocated, when in fact only a fraction of the storage has been allocated.



### Note

While the Red Hat Virtualization Manager provides its own thin provisioning function, you should use the thin provisioning functionality of your storage back-end if it provides one.

To support storage over-commitment, a threshold is defined in VDSM which compares logical storage allocation with actual storage usage. This threshold is used to make sure that the data written to a disk image is smaller than the logical volume that backs it. QEMU identifies the highest offset written to in a logical volume, which indicates the point of greatest storage use. VDSM monitors the highest offset marked by QEMU to ensure that the usage does not cross the defined threshold. So long as VDSM continues to indicate that the highest offset remains below the threshold, the Red Hat Virtualization Manager knows that the logical volume in question has sufficient storage to continue operations.

When QEMU indicates that usage has risen to exceed the threshold limit, VDSM communicates to the Manager that the disk image will soon reach the size of its logical volume. The Red Hat Virtualization Manager requests that the SPM host extend the logical volume. This process can be repeated as long as the data storage domain for the data center has available space. When the data storage domain runs out of available free space, you must manually add storage capacity to expand it.

## 2.12. Logical Volume Extension

The Red Hat Virtualization Manager uses thin provisioning to overcommit the storage available in a storage pool, and allocates more storage than is physically available. Virtual machines write data as they operate. A



virtual machine with a thinly-provisioned disk image will eventually write more data than the logical volume backing its disk image can hold. When this happens, logical volume extension is used to provide additional storage and facilitate the continued operations for the virtual machine.

Red Hat Virtualization provides a thin provisioning mechanism over LVM. When using QCOW2 formatted storage, Red Hat Virtualization relies on the host system process *qemu-kvm* to map storage blocks on disk to logical blocks in a sequential manner. This allows, for example, the definition of a logical 100 GB disk backed by a 1 GB logical volume. When *qemu-kvm* crosses a usage threshold set by VDSM, the local VDSM instance makes a request to the SPM for the logical volume to be extended by another one gigabyte. VDSM on the host running a virtual machine in need of volume extension notifies the SPM VDSM that more space is required. The SPM extends the logical volume and the SPM VDSM instance causes the host VDSM to refresh volume group information and recognize that the extend operation is complete. The host can continue operations.

Logical Volume extension does not require that a host know which other host is the SPM; it could even be the SPM itself. The storage extension communication is done via a storage mailbox. The storage mailbox is a dedicated logical volume on the data storage domain. A host that needs the SPM to extend a logical volume writes a message in an area designated to that particular host in the storage mailbox. The SPM periodically reads the incoming mail, performs requested logical volume extensions, and writes a reply in the outgoing mail. After sending the request, a host monitors its incoming mail for responses every two seconds. When the host receives a successful reply to its logical volume extension request, it refreshes the logical volume map in device mapper to recognize the newly allocated storage.

When the physical storage available to a storage pool is nearly exhausted, multiple images can run out of usable storage with no means to replenish their resources. A storage pool that exhausts its storage causes QEMU to return an **enospc error**, which indicates that the device no longer has any storage available. At this point, running virtual machines are automatically paused and manual intervention is required to add a new LUN to the volume group.

When a new LUN is added to the volume group, the Storage Pool Manager automatically distributes the additional storage to logical volumes that need it. The automatic allocation of additional resources allows the relevant virtual machines to automatically continue operations uninterrupted or resume operations if stopped.

## Chapter 3. Network

### 3.1. Network Architecture

Red Hat Virtualization networking can be discussed in terms of basic networking, networking within a cluster, and host networking configurations. Basic networking terms cover the basic hardware and software elements that facilitate networking. Networking within a cluster includes network interactions among cluster level objects such as hosts, logical networks and virtual machines. Host networking configurations covers supported configurations for networking within a host.

A well designed and built network ensures, for example, that high bandwidth tasks receive adequate bandwidth, that user interactions are not crippled by latency, and virtual machines can be successfully migrated within a migration domain. A poorly built network can cause, for example, unacceptable latency, and migration and cloning failures resulting from network flooding.

### 3.2. Introduction: Basic Networking Terms

Red Hat Virtualization provides networking functionality between virtual machines, virtualization hosts, and wider networks using:

- » A Network Interface Controller (NIC)
- » A Bridge
- » A Bond
- » A Virtual NIC
- » A Virtual LAN (VLAN)

NICs, bridges, and VNICs allow for network communication between hosts, virtual machines, local area networks, and the Internet. Bonds and VLANs are optionally implemented to enhance security, fault tolerance and network capacity.

### 3.3. Network Interface Controller

The *NIC (Network Interface Controller)* is a network adapter or LAN adapter that connects a computer to a computer network. The NIC operates on both the physical and data link layers of the machine and allows network connectivity. All virtualization hosts in a Red Hat Virtualization environment have at least one NIC, though it is more common for a host to have two or more NICs.

One physical NIC can have multiple Virtual NICs (VNICs) logically connected to it. A virtual NIC acts as a physical network interface for a virtual machine. To distinguish between a VNIC and the NIC that supports it, the Red Hat Virtualization Manager assigns each VNIC a unique MAC address.

### 3.4. Bridge

A *Bridge* is a software device that uses packet forwarding in a packet-switched network. Bridging allows multiple network interface devices to share the connectivity of one NIC and appear on a network as separate physical devices. The bridge examines a packet's source addresses to determine relevant target addresses. Once the target address is determined, the bridge adds the location to a table for future reference. This allows a host to redirect network traffic to virtual machine associated VNICs that are members of a bridge.

In Red Hat Virtualization a logical network is implemented using a bridge. It is the bridge rather than the



physical interface on a host that receives an IP address. The IP address associated with the bridge is not required to be within the same subnet as the virtual machines that use the bridge for connectivity. If the bridge is assigned an IP address on the same subnet as the virtual machines that use it, the host is addressable within the logical network by virtual machines. As a rule it is not recommended to run network exposed services on a virtualization host. Guests are connected to a logical network by their VNICs, and the host is connected to remote elements of the logical network using its NIC. Each guest can have the IP address of its VNIC set independently, by DHCP or statically. Bridges can connect to objects outside the host, but such a connection is not mandatory.

Custom properties can be defined for both the bridge and the Ethernet connection. VDSM passes the network definition and custom properties to the setup network hook script.

### 3.5. Bonds

A *bond* is an aggregation of multiple network interface cards into a single software-defined device. Because bonded network interfaces combine the transmission capability of the network interface cards included in the bond to act as a single network interface, they can provide greater transmission speed than that of a single network interface card. Also, because all network interface cards in the bond must fail for the bond itself to fail, bonding provides increased fault tolerance. However, one limitation is that the network interface cards that form a bonded network interface must be of the same make and model to ensure that all network interface cards in the bond support the same options and modes.

The packet dispersal algorithm for a bond is determined by the bonding mode used.



#### Important

Modes 1, 2, 3 and 4 support both virtual machine (bridged) and non-virtual machine (bridgeless) network types. Modes 0, 5 and 6 support non-virtual machine (bridgeless) networks only.

### Bonding Modes

Red Hat Virtualization uses Mode 4 by default, but supports the following common bonding modes:

#### ***Mode 0 (round-robin policy)***

Transmits packets through network interface cards in sequential order. Packets are transmitted in a loop that begins with the first available network interface card in the bond and end with the last available network interface card in the bond. All subsequent loops then start with the first available network interface card. Mode 0 offers fault tolerance and balances the load across all network interface cards in the bond. However, Mode 0 cannot be used in conjunction with bridges, and is therefore not compatible with virtual machine logical networks.

#### ***Mode 1 (active-backup policy)***

Sets all network interface cards to a backup state while one network interface card remains active. In the event of failure in the active network interface card, one of the backup network interface cards replaces that network interface card as the only active network interface card in the bond. The MAC address of the bond in Mode 1 is visible on only one port to prevent any confusion that might otherwise be caused if the MAC address of the bond changed to reflect that of the active network interface card. Mode 1 provides fault tolerance and is supported in Red Hat Virtualization.

#### ***Mode 2 (XOR policy)***

Selects the network interface card through which to transmit packets based on the result of an XOR

operation on the source and destination MAC addresses modulo network interface card slave count. This calculation ensures that the same network interface card is selected for each destination MAC address used. Mode 2 provides fault tolerance and load balancing and is supported in Red Hat Virtualization.

#### **Mode 3 (*broadcast policy*)**

Transmits all packets to all network interface cards. Mode 3 provides fault tolerance and is supported in Red Hat Virtualization.

#### **Mode 4 (*IEEE 802.3ad policy*)**

Creates aggregation groups in which the interfaces share the same speed and duplex settings. Mode 4 uses all network interface cards in the active aggregation group in accordance with the IEEE 802.3ad specification and is supported in Red Hat Virtualization.

#### **Mode 5 (*adaptive transmit load balancing policy*)**

Ensures the distribution of outgoing traffic accounts for the load on each network interface card in the bond and that the current network interface card receives all incoming traffic. If the network interface card assigned to receive traffic fails, another network interface card is assigned to the role of receiving incoming traffic. Mode 5 cannot be used in conjunction with bridges, therefore it is not compatible with virtual machine logical networks.

#### **Mode 6 (*adaptive load balancing policy*)**

Combines Mode 5 (adaptive transmit load balancing policy) with receive load balancing for IPv4 traffic without any special switch requirements. ARP negotiation is used for balancing the receive load. Mode 6 cannot be used in conjunction with bridges, therefore it is not compatible with virtual machine logical networks.

### 3.6. Switch Configuration for Bonding

Switch configurations vary per the requirements of your hardware. Refer to the deployment and networking configuration guides for your operating system.



#### **Important**

For every type of switch it is important to set up the switch bonding with the *Link Aggregation Control Protocol (LACP)* protocol and *not* the *Cisco Port Aggregation Protocol (PAgP)* protocol.

### 3.7. Virtual Network Interface Cards

Virtual network interface cards are virtual network interfaces that are based on the physical network interface cards of a host. Each host can have multiple network interface cards, and each network interface card can be a base for multiple virtual network interface cards.

When you attach a virtual network interface card to a virtual machine, the Red Hat Virtualization Manager creates several associations between the virtual machine to which the virtual network interface card is being attached, the virtual network interface card itself, and the physical host network interface card on which the virtual network interface card is based. Specifically, when a virtual network interface card is attached to a virtual machine, a new virtual network interface card and MAC address are created on the physical host

network interface card on which the virtual network interface card is based. Then, the first time the virtual machine starts after that virtual network interface card is attached, **libvirt** assigns the virtual network interface card a PCI address. The MAC address and PCI address are then used to obtain the name of the virtual network interface card (for example, **eth0**) in the virtual machine.

The process for assigning MAC addresses and associating those MAC addresses with PCI addresses is slightly different when creating virtual machines based on templates or snapshots. When PCI addresses have already been created for a template or snapshot, the virtual network interface cards on virtual machines created based on that template or snapshot are ordered in accordance with those PCI addresses and MAC addresses allocated in that order. If PCI addresses have not already been created for a template, the virtual network interface cards on virtual machines created based on that template are allocated in the order of the naming of the virtual network interface cards. If PCI addresses have not already been created for a snapshot, the Red Hat Virtualization Manager allocates new MAC addresses to the virtual network interface cards on virtual machines based on that snapshot.

Once created, virtual network interface cards are added to a network bridge device. The network bridge devices are how virtual machines are connected to virtual machine logical networks.

Running the **ip addr show** command on a virtualization host shows all of the virtual network interface cards that are associated with virtual machines on that host. Also visible are any network bridges that have been created to back logical networks, and any network interface cards used by the host.

```
[root@rhev-host-01 ~]# ip addr show
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 16436 qdisc noqueue state UNKNOWN
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
    inet6 ::1/128 scope host
        valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state
UP qlen 1000
    link/ether 00:21:86:a2:85:cd brd ff:ff:ff:ff:ff:ff
    inet6 fe80::221:86ff:fea2:85cd/64 scope link
        valid_lft forever preferred_lft forever
3: wlan0: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc mq state DOWN
qlen 1000
    link/ether 00:21:6b:cc:14:6c brd ff:ff:ff:ff:ff:ff
5: vdsmdummy;: <BROADCAST,MULTICAST> mtu 1500 qdisc noop state DOWN
    link/ether 4a:d5:52:c2:7f:4b brd ff:ff:ff:ff:ff:ff
6: bond0: <BROADCAST,MULTICAST,MASTER> mtu 1500 qdisc noop state DOWN
    link/ether 00:00:00:00:00:00 brd ff:ff:ff:ff:ff:ff
7: bond4: <BROADCAST,MULTICAST,MASTER> mtu 1500 qdisc noop state DOWN
    link/ether 00:00:00:00:00:00 brd ff:ff:ff:ff:ff:ff
8: bond1: <BROADCAST,MULTICAST,MASTER> mtu 1500 qdisc noop state DOWN
    link/ether 00:00:00:00:00:00 brd ff:ff:ff:ff:ff:ff
9: bond2: <BROADCAST,MULTICAST,MASTER> mtu 1500 qdisc noop state DOWN
    link/ether 00:00:00:00:00:00 brd ff:ff:ff:ff:ff:ff
10: bond3: <BROADCAST,MULTICAST,MASTER> mtu 1500 qdisc noop state DOWN
    link/ether 00:00:00:00:00:00 brd ff:ff:ff:ff:ff:ff
11: ovirtmgmt: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue
state UNKNOWN
    link/ether 00:21:86:a2:85:cd brd ff:ff:ff:ff:ff:ff
    inet 10.64.32.134/23 brd 10.64.33.255 scope global ovirtmgmt
    inet6 fe80::221:86ff:fea2:85cd/64 scope link
        valid_lft forever preferred_lft forever
```

The console output from the command shows several devices: one loop back device (**lo**), one Ethernet device (**eth0**), one wireless device (**wlan0**), one VDSM dummy device (**;vdsmdummy;**), five bond devices (**bond0**, **bond4**, **bond1**, **bond2**, **bond3**), and one network bridge (**ovirtmgmt**).

Virtual network interface cards are all members of a network bridge device and logical network. Bridge membership can be displayed using the **brctl show** command:

```
[root@rhev-host-01 ~]# brctl show
bridge name bridge id      STP enabled interfaces
ovirtmgmt  8000.e41f13b7fdd4 no    vnet002
          vnet001
          vnet000
          eth0
```

The console output from the **brctl show** command shows that the virtio virtual network interface cards are members of the **ovirtmgmt** bridge. All of the virtual machines that the virtual network interface cards are associated with are connected to the **ovirtmgmt** logical network. The **eth0** network interface card is also a member of the **ovirtmgmt** bridge. The **eth0** device is cabled to a switch that provides connectivity beyond the host.

### 3.8. Virtual LAN (VLAN)

A *VLAN (Virtual LAN)* is an attribute that can be applied to network packets. Network packets can be "tagged" into a numbered VLAN. A VLAN is a security feature used to completely isolate network traffic at the switch level. VLANs are completely separate and mutually exclusive. The Red Hat Virtualization Manager is VLAN aware and able to tag and redirect VLAN traffic, however VLAN implementation requires a switch that supports VLANs.

At the switch level, ports are assigned a VLAN designation. A switch applies a VLAN tag to traffic originating from a particular port, marking the traffic as part of a VLAN, and ensures that responses carry the same VLAN tag. A VLAN can extend across multiple switches. VLAN tagged network traffic on a switch is completely undetectable except by machines connected to a port designated with the correct VLAN. A given port can be tagged into multiple VLANs, which allows traffic from multiple VLANs to be sent to a single port, to be deciphered using software on the machine that receives the traffic.

### 3.9. Network Labels

Network labels can be used to greatly simplify several administrative tasks associated with creating and administering logical networks and associating those logical networks with physical host network interfaces and bonds.

A network label is a plain text, human readable label that can be attached to a logical network or a physical host network interface. There is no strict limit on the length of label, but you must use a combination of lowercase and uppercase letters, underscores and hyphens; no spaces or special characters are allowed.

Attaching a label to a logical network or physical host network interface creates an association with other logical networks or physical host network interfaces to which the same label has been attached, as follows:

#### Network Label Associations

- ✱ When you attach a label to a logical network, that logical network will be automatically associated with any physical host network interfaces with the given label.

- ✦ When you attach a label to a physical host network interface, any logical networks with the given label will be automatically associated with that physical host network interface.
- ✦ Changing the label attached to a logical network or physical host network interface acts in the same way as removing a label and adding a new label. The association between related logical networks or physical host network interfaces is updated.

### Network Labels and Clusters

- ✦ When a labeled logical network is added to a cluster and there is a physical host network interface in that cluster with the same label, the logical network is automatically added to that physical host network interface.
- ✦ When a labeled logical network is detached from a cluster and there is a physical host network interface in that cluster with the same label, the logical network is automatically detached from that physical host network interface.

### Network Labels and Logical Networks With Roles

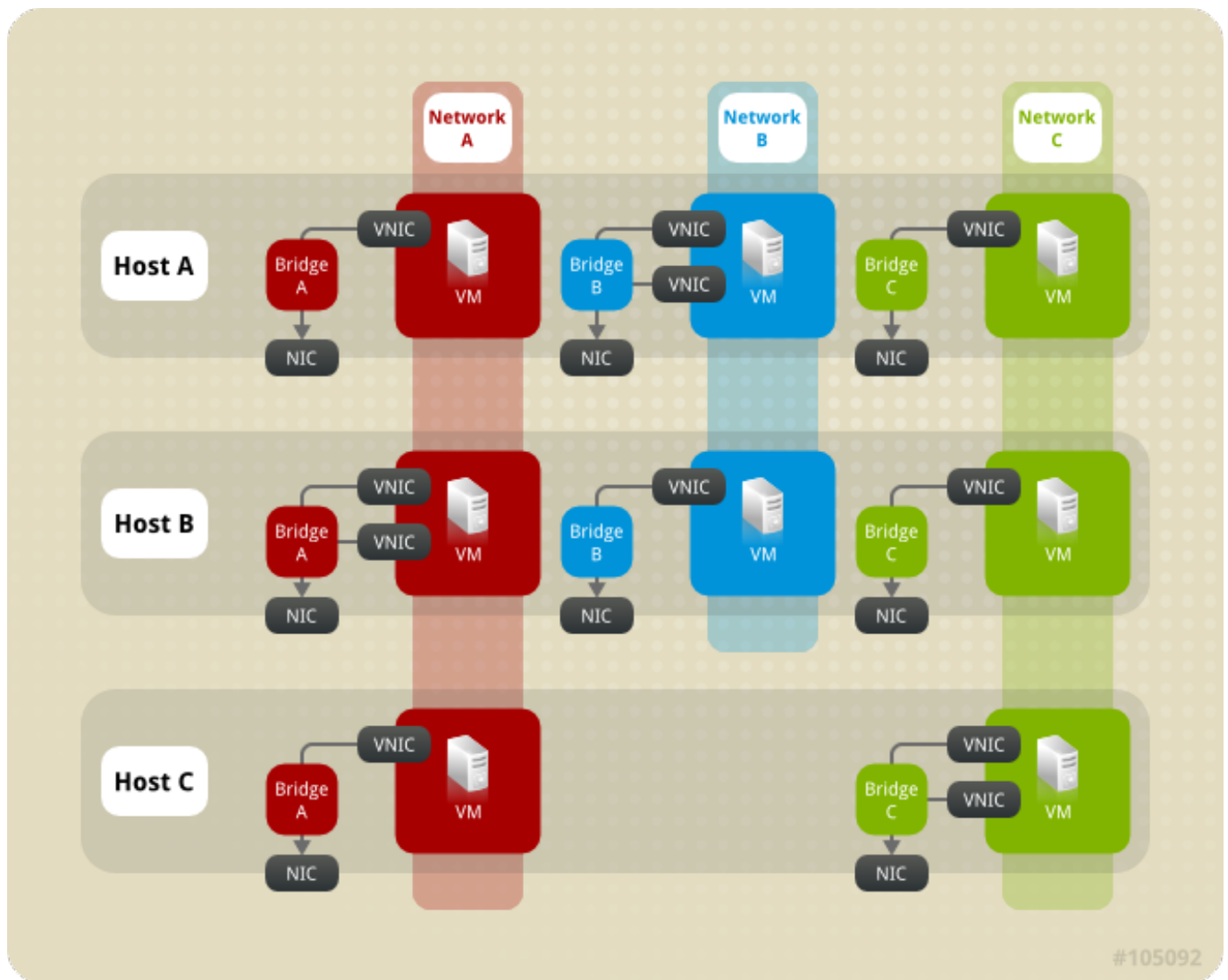
- ✦ When a labeled logical network is assigned to act as a display network or migration network, that logical network is then configured on the physical host network interface using DHCP so that the logical network can be assigned an IP address.

Setting a label on a role network (for instance, "a migration network" or "a display network") causes a mass deployment of that network on all hosts. Such mass additions of networks are achieved through the use of DHCP. This method of mass deployment was chosen over a method of typing in static addresses, because of the unscalable nature of the task of typing in many static IP addresses.

## 3.10. Cluster Networking

Cluster level networking objects include:

- ✦ Clusters
- ✦ Logical Networks



**Figure 3.1. Networking within a cluster**

A data center is a logical grouping of multiple clusters and each cluster is a logical group of multiple hosts. [Figure 3.1, “Networking within a cluster”](#) depicts the contents of a single cluster.

Hosts in a cluster all have access to the same storage domains. Hosts in a cluster also have logical networks applied at the cluster level. For a virtual machine logical network to become operational for use with virtual machines, the network must be defined and implemented for each host in the cluster using the Red Hat Virtualization Manager. Other logical network types can be implemented on only the hosts that use them.

Multi-host network configuration automatically applies any updated network settings to all of the hosts within the data center to which the network is assigned.

### 3.11. Logical Networks

Logical networking allows the Red Hat Virtualization environment to separate network traffic by type. For example, the `ovirtmgmt` network is created by default during the installation of the Red Hat Virtualization to be used for management communication between the Manager and hosts. A typical use for logical networks is to group network traffic with similar requirements and usage together. In many cases, a storage network and a display network are created by an administrator to isolate traffic of each respective type for optimization and troubleshooting.

The types of logical network are:

- ✧ logical networks that carry virtual machine network traffic,
- ✧ logical networks that do not carry virtual machine network traffic,
- ✧ optional logical networks,
- ✧ and required networks.

All logical networks can either be required or optional.

Logical networks are defined at the data center level, and added to a host. For a required logical network to be operational, it must be implemented for every host in a given cluster.

Each virtual machine logical network in a Red Hat Virtualization environment is backed by a network bridge device on a host. So when a new virtual machine logical network is defined for a cluster, a matching bridge device must be created on each host in the cluster before the logical network can become operational to be used by virtual machines. Red Hat Virtualization Manager automatically creates required bridges for virtual machine logical networks.

The bridge device created by the Red Hat Virtualization Manager to back a virtual machine logical network is associated with a host network interface. If the host network interface that is part of a bridge has network connectivity, then any network interfaces that are subsequently included in the bridge share the network connectivity of the bridge. When virtual machines are created and placed on a particular logical network, their virtual network cards are included in the bridge for that logical network. Those virtual machines can then communicate with each other and with other objects that are connected to the bridge.

Logical networks not used for virtual machine network traffic are associated with host network interfaces directly.

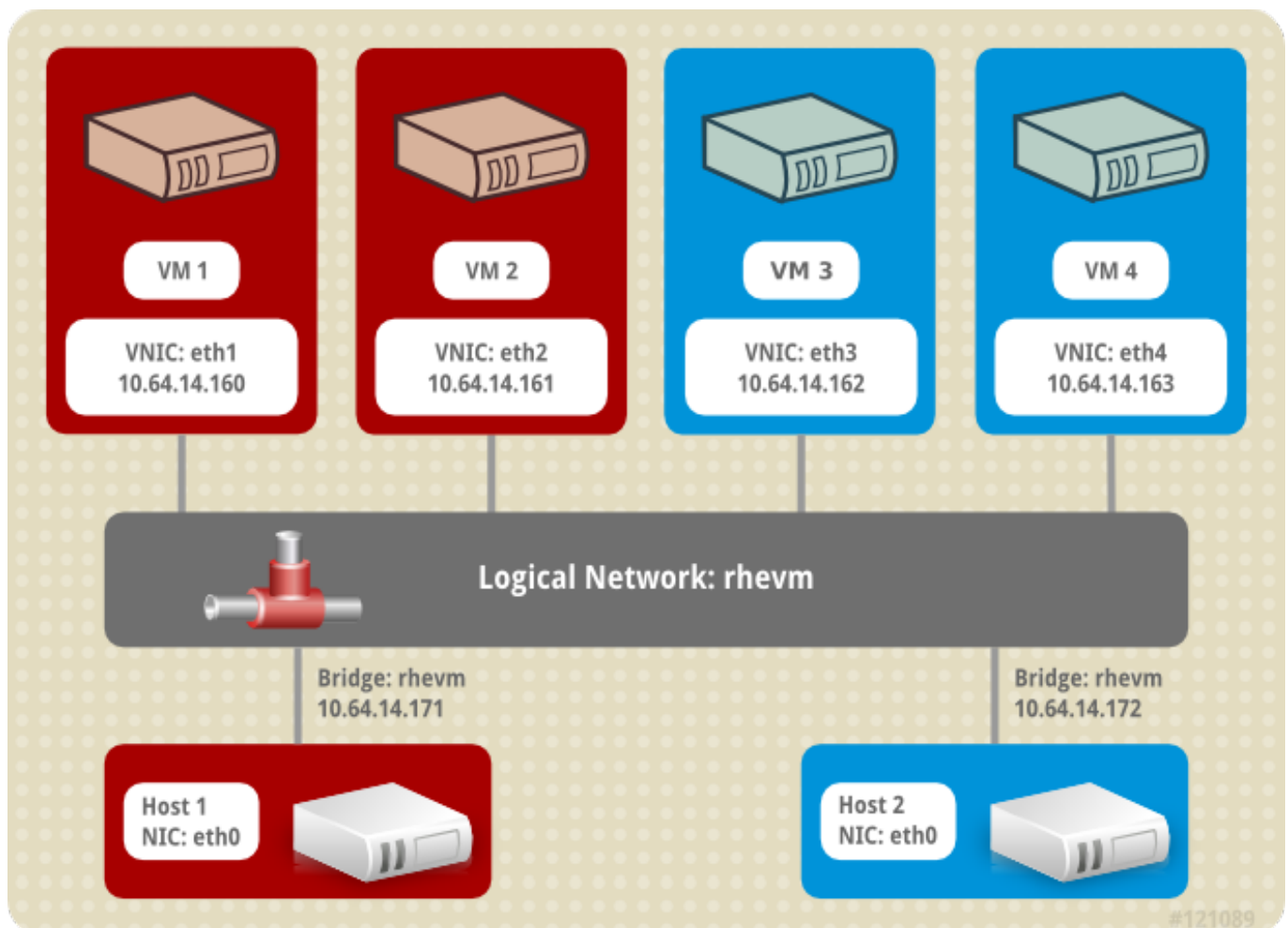


Figure 3.2. The ovirtmgmt logical network.



**Example 3.1. Example usage of a logical network.**

There are two hosts called Red and White in a cluster called Pink in a data center called Purple. Both Red and White have been using the default logical network, **ovirtmgmt** for all networking functions. The system administrator responsible for Pink decides to isolate network testing for a web server by placing the web server and some client virtual machines on a separate logical network. She decides to call the new logical network **network\_testing**.

First, she defines the logical network for the Purple data center. She then applies it to the Pink cluster. Logical networks must be implemented on a host in maintenance mode. So, the administrator first migrates all running virtual machines to Red, and puts White in maintenance mode. Then she edits the **Network** associated with the physical network interface that will be included in the bridge. The **Link Status** for the selected network interface will change from **Down** to **Non-Operational**. The non-operational status is because the corresponding bridge must be setup in all hosts in the cluster by adding a physical network interface on each host in the Pink cluster to the **network\_testing** network. Next she activates White, migrates all of the running virtual machines off of Red, and repeats the process for Red.

When both White and Red both have the **network\_testing** logical network bridged to a physical network interface, the **network\_testing** logical network becomes **Operational** and is ready to be used by virtual machines.

## 3.12. Required Networks, Optional Networks, and Virtual Machine Networks

A required network is a logical network that must be available to all hosts in a cluster. When a host's required network becomes non-operational, virtual machines running on that host are migrated to another host; the extent of this migration is dependent upon the chosen scheduling policy. This is beneficial if you have virtual machines running mission critical workloads.

An optional network is a logical network that has not been explicitly declared as **Required**. Optional networks can be implemented on only the hosts that use them. The presence or absence of optional networks does not affect the **Operational** status of a host. When a non-required network becomes non-operational, the virtual machines running on the network are not migrated to another host. This prevents unnecessary I/O overload caused by mass migrations. Note that when a logical network is created and added to clusters, the **Required** box is checked by default.

To change a network's **Required** designation, from the Administration Portal, select a network, click the **Cluster** tab, and click the **Manage Networks** button.

Virtual machine networks (called a **VM network** in the user interface) are logical networks designated to carry only virtual machine network traffic. Virtual machine networks can be required or optional. Virtual machines that uses an optional virtual machine network will only start on hosts with that network.

## 3.13. Virtual Machine Connectivity

In Red Hat Virtualization, a virtual machine has its NIC put on a logical network at the time that the virtual machine is created. From that point, the virtual machine is able to communicate with any other destination on the same network.



From the host perspective, when a virtual machine is put on a logical network, the vNIC that backs the virtual machine's NIC is added as a member to the bridge device for the logical network. For example, if a virtual machine is on the `ovirtmgmt` logical network, its vNIC is added as a member of the `ovirtmgmt` bridge of the host on which that virtual machine runs.

### 3.14. Port Mirroring

Port mirroring copies layer 3 network traffic on a given logical network and host to a virtual interface on a virtual machine. This virtual machine can be used for network debugging and tuning, intrusion detection, and monitoring the behavior of other virtual machines on the same host and logical network.

The only traffic copied is internal to one logical network on one host. There is no increase on traffic on the network external to the host; however a virtual machine with port mirroring enabled uses more host CPU and RAM than other virtual machines.

Port mirroring is enabled or disabled in the vNIC profiles of logical networks, and has the following limitations:

- ✦ Hot plugging vNICs with a profile that has port mirroring enabled is not supported.
- ✦ Port mirroring cannot be altered when the vNIC profile is attached to a virtual machine.

Given the above limitations, it is recommended that you enable port mirroring on an additional, dedicated vNIC profile.



#### Important

Enabling port mirroring reduces the privacy of other network users.

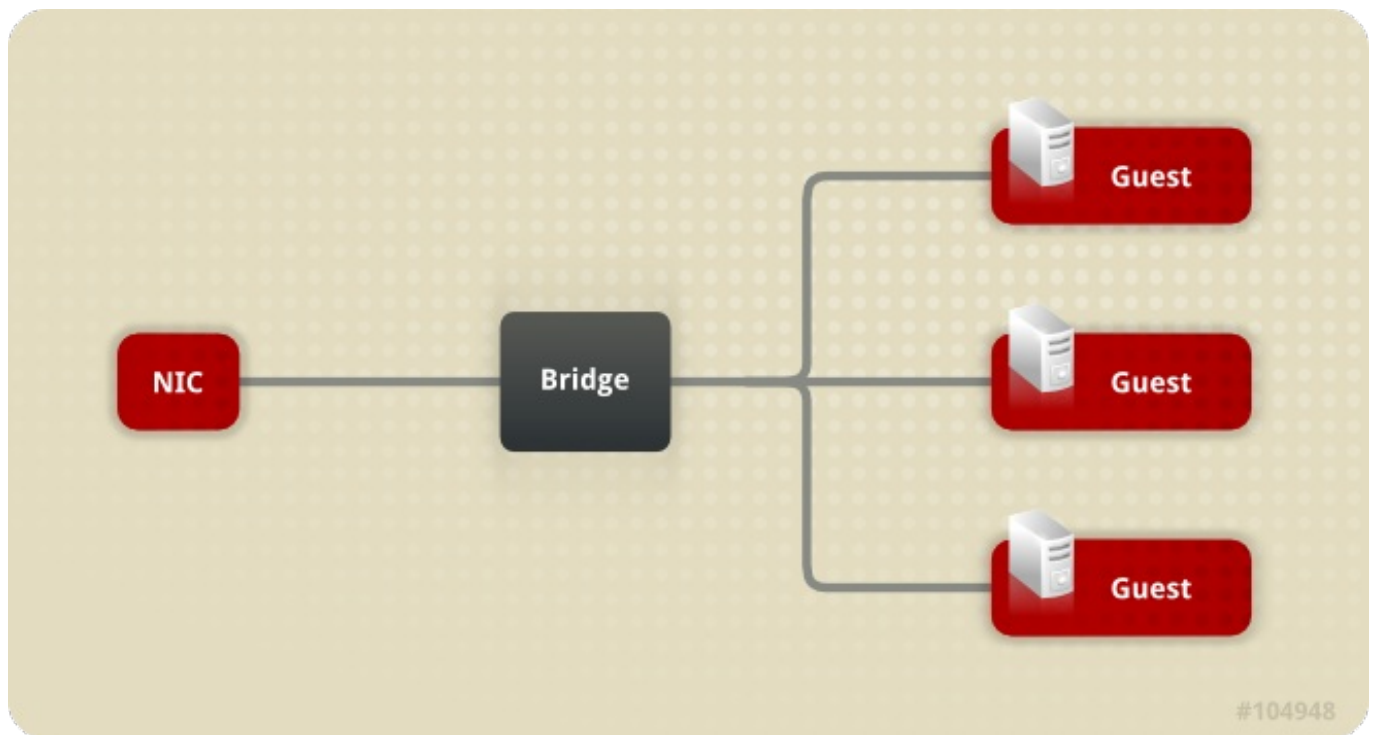
### 3.15. Host Networking Configurations

Common types of networking configurations for virtualization hosts include:

- ✦ Bridge and NIC configuration.
- ✦ Bridge, VLAN, and NIC configuration.
- ✦ Bridge, Bond, and VLAN configuration.
- ✦ Multiple Bridge, Multiple VLAN, and NIC configuration.

### 3.16. Bridge Configuration

The simplest host configuration in Red Hat Virtualization is the Bridge and NIC configuration. As [Figure 3.3, “Bridge and NIC configuration”](#) depicts, this configuration uses a bridge to connect one or more virtual machines (or guests) to the host's NIC.

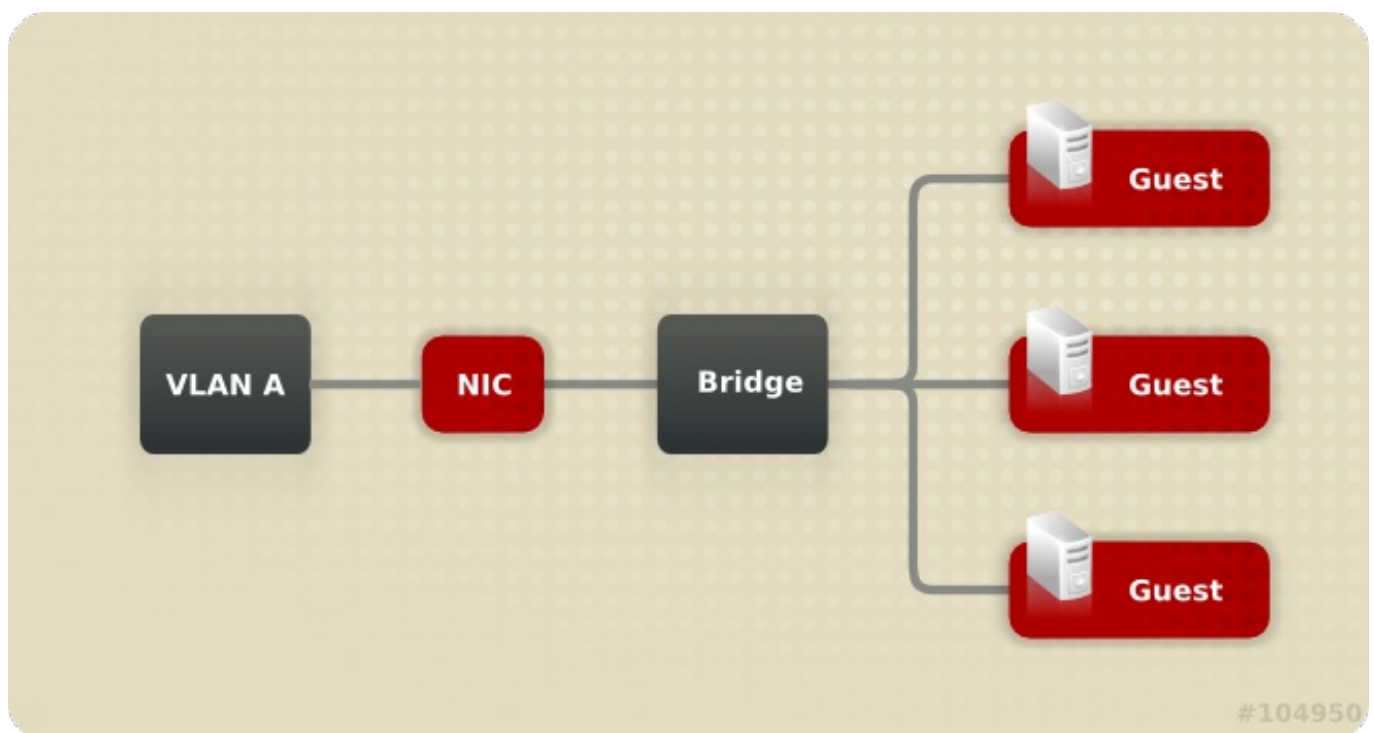


**Figure 3.3. Bridge and NIC configuration**

An example of this configuration is the automatic creation of the bridge **ovirtmgmt** when the Red Hat Virtualization Manager installs. On installation, the Red Hat Virtualization Manager installs **VDSM** on the host. The **VDSM** installation process creates the bridge **ovirtmgmt**. The **ovirtmgmt** bridge then obtains the **IP** address of the host to enable management communication for the host.

### 3.17. VLAN Configuration

[Figure 3.4, “Bridge, VLAN, and NIC configuration”](#) depicts an alternative configuration that includes a virtual LAN (VLAN) to connect the host NIC and bridge.

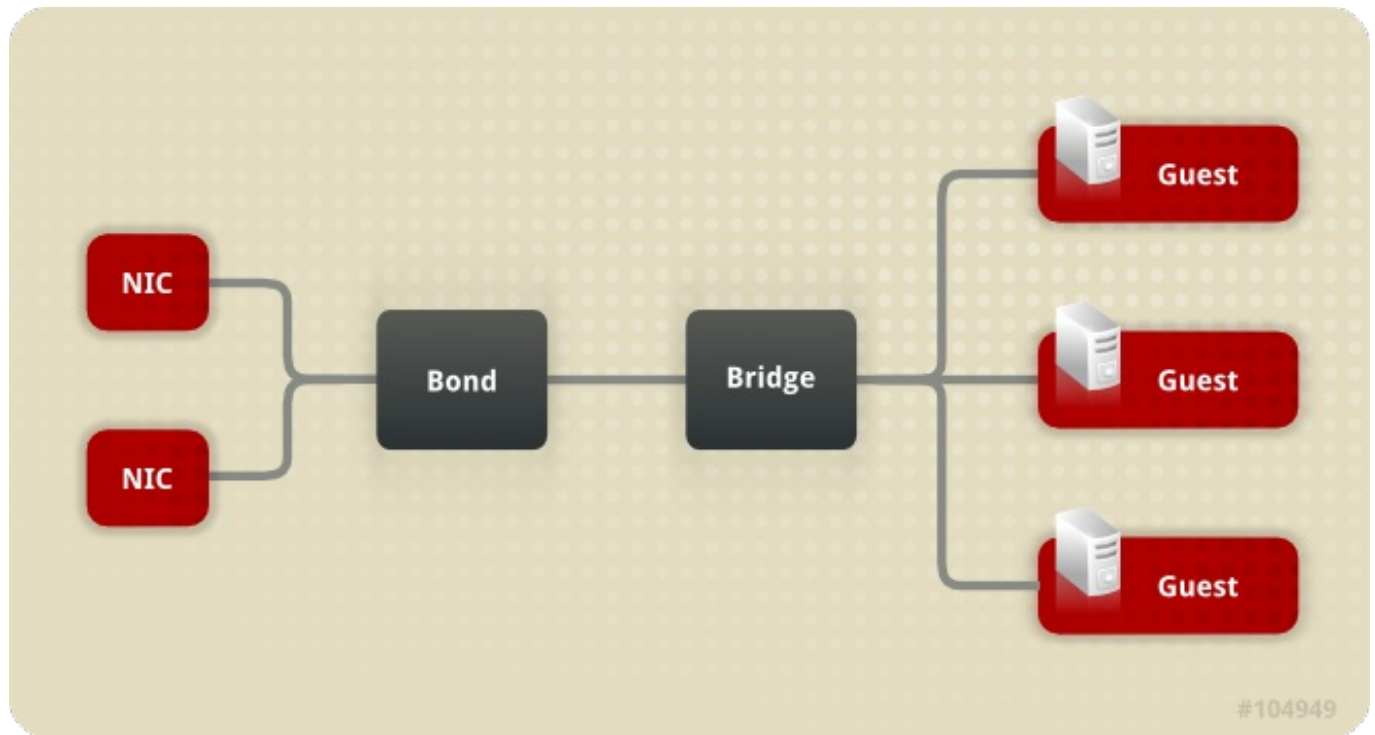


**Figure 3.4. Bridge, VLAN, and NIC configuration**

A VLAN is included to provide a secure channel for data transfer over this network and also to support the option to connect multiple bridges to a single NIC using multiple VLANs.

### 3.18. Bridge and Bond Configuration

[Figure 3.5, “Bridge, Bond, and NIC configuration”](#) displays a configuration that includes a bond to connect multiple host NICs to the same bridge and network.

**Figure 3.5. Bridge, Bond, and NIC configuration**

The included bond creates a logical link that combines the two (or more) physical Ethernet links. The resultant benefits include NIC fault tolerance and potential bandwidth extension, depending on the bonding mode.

### 3.19. Multiple Bridge, Multiple VLAN, and NIC Configuration

[Figure 3.6, “Multiple Bridge, Multiple VLAN, and NIC configuration”](#) depicts a configuration that connects a single NIC to two VLANs. This presumes that the network switch has been configured to pass network traffic that has been tagged into one of the two VLANs to one NIC on the host. The host uses two VNICs to separate VLAN traffic, one for each VLAN. Traffic tagged into either VLAN then connects to a separate bridge by having the appropriate VNIC as a bridge member. Each bridge is in turn connected to by multiple virtual machines.

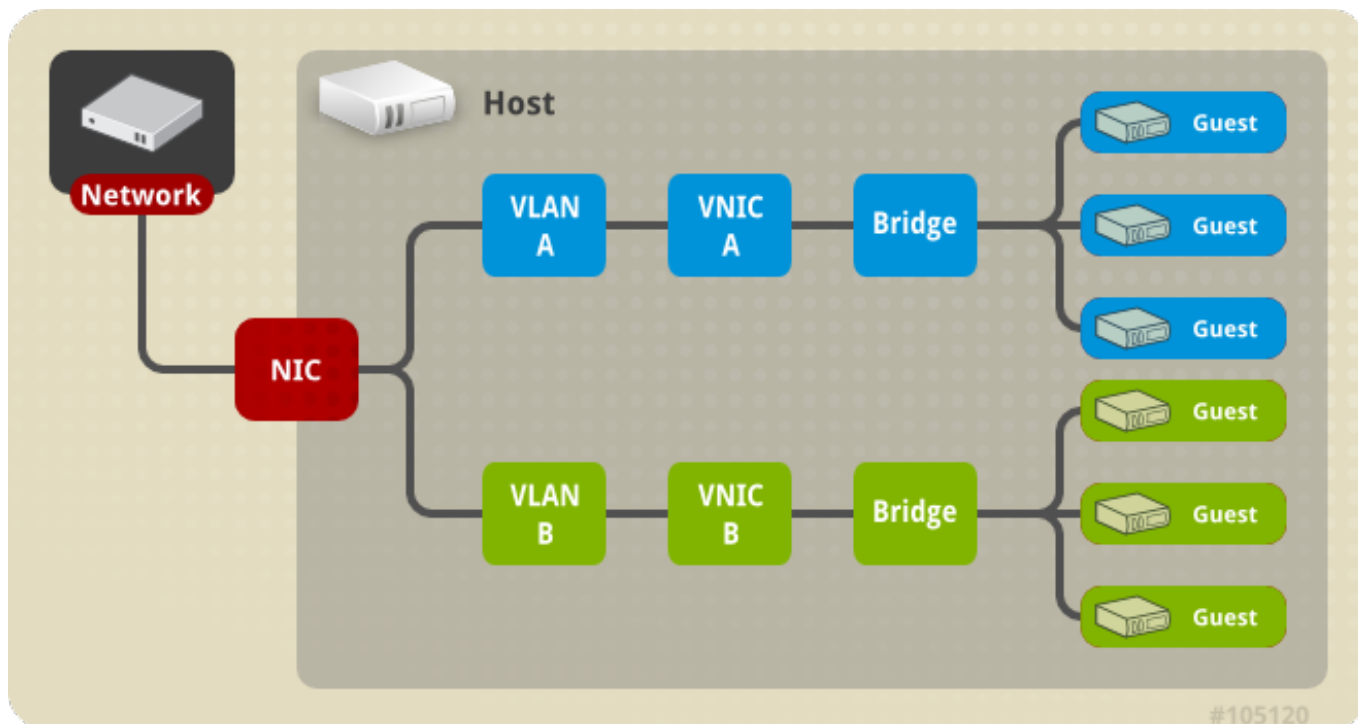


Figure 3.6. Multiple Bridge, Multiple VLAN, and NIC configuration

### 3.20. Multiple Bridge, Multiple VLAN, and Bond Configuration

Figure 3.7, “Multiple Bridge, Multiple VLAN, and Multiple NIC with Bond connection” displays a configuration that bonds multiple NICs to facilitate a connection with multiple VLANs.

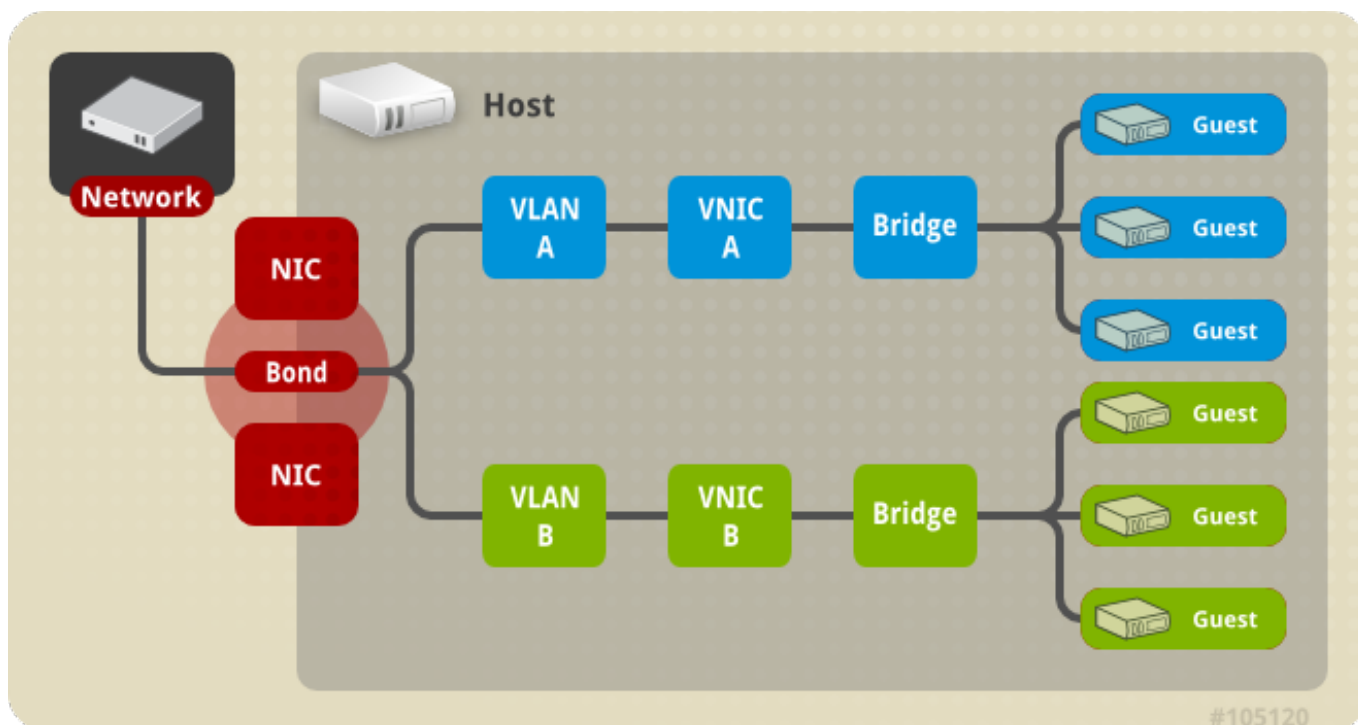


Figure 3.7. Multiple Bridge, Multiple VLAN, and Multiple NIC with Bond connection

Each VLAN in this configuration is defined over the bond connecting the NICs. Each VLAN connects to an individual bridge and each bridge connects to one or more guests.

## Chapter 4. Power Management

### 4.1. Introduction to Power Management and Fencing

The Red Hat Virtualization environment is most flexible and resilient when power management and fencing have been configured. Power management allows the Red Hat Virtualization Manager to control host power cycle operations, most importantly to reboot hosts on which problems have been detected. Fencing is used to isolate problem hosts from a functional Red Hat Virtualization environment by rebooting them, in order to prevent performance degradation. Fenced hosts can then be returned to responsive status through administrator action and be reintegrated into the environment.

Power management and fencing make use of special dedicated hardware in order to restart hosts independently of host operating systems. The Red Hat Virtualization Manager connects to a power management devices using a network IP address or hostname. In the context of Red Hat Virtualization, a power management device and a fencing device are the same thing.

### 4.2. Power Management by Proxy in Red Hat Virtualization

The Red Hat Virtualization Manager does not communicate directly with fence agents. Instead, the Manager uses a proxy to send power management commands to a host power management device. The Manager uses VDSM to execute power management device actions, so another host in the environment is used as a fencing proxy.

You can select between:

- ✧ Any host in the same cluster as the host requiring fencing.
- ✧ Any host in the same data center as the host requiring fencing.

A viable fencing proxy host has a status of either *UP* or *Maintenance*.

### 4.3. Power Management

The Red Hat Virtualization Manager 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. The Red Hat Virtualization environment supports the following power management devices:

- ✧ *American Power Conversion* (apc).
- ✧ *Bladecenter*.
- ✧ *Cisco Unified Computing System* (cisco\_ucs).
- ✧ *Dell Remote Access Card 5* (drac5).
- ✧ *Dell Remote Access Card 7* (drac7).
- ✧ *Electronic Power Switch* (eps).
- ✧ *HP BladeSystem* (hpblade).
- ✧ *Integrated Lights Out* (ilo, ilo2, ilo3, ilo4).
- ✧ *Intelligent Platform Management Interface* (ipmilan).

- ✧ *Remote Supervisor Adapter (rsa).*
- ✧ *rsb.*
- ✧ *Western Telematic, Inc (wti).*



## Note

APC 5.x power management devices are not supported by the **apc** fence agent. Use the **apc\_snmp** fence agent instead.

In order to communicate with the listed power management devices, the Red Hat Virtualization Manager makes use of *fence agents*. The Red Hat Virtualization Manager allows administrators to configure a fence agent for the power management device in their environment with parameters the device will accept and respond to. Basic configuration options can be configured using the graphical user interface. Special configuration options can also be entered, and are passed un-parsed to the fence device. Special configuration options are specific to a given fence device, while basic configuration options are for functionalities provided by all supported power management devices. The basic functionalities provided by all power management devices are:

- ✧ **Status:** check the status of the host.
- ✧ **Start:** power on the host.
- ✧ **Stop:** power down the host.
- ✧ **Restart:** restart the host. Actually implemented as stop, wait, status, start, wait, status.

Best practice is to test the power management configuration once when initially configuring it, and occasionally after that to ensure continued functionality.

Resilience is provided by properly configured power management devices in all of the hosts in an environment. Fencing agents allow the Red Hat Virtualization Manager to communicate with host power management devices to bypass the operating system on a problem host, and isolate the host from the rest of its environment by rebooting it. The Manager can then reassign the SPM role, if it was held by the problem host, and safely restart any highly available virtual machines on other hosts.

## 4.4. Fencing

In the context of the Red Hat Virtualization environment, fencing is a host reboot initiated by the Manager using a fence agent and performed by a power management device. Fencing allows a cluster to react to unexpected host failures as well as enforce power saving, load balancing, and virtual machine availability policies.

Fencing ensures that the role of Storage Pool Manager (SPM) is always assigned to a functional host. If the fenced host was the SPM, the SPM role is relinquished and reassigned to a responsive host. Because the host with the SPM role is the only host that is able to write data domain structure metadata, a non-responsive, un-fenced SPM host causes its environment to lose the ability to create and destroy virtual disks, take snapshots, extend logical volumes, and all other actions that require changes to data domain structure metadata.

When a host becomes non-responsive, all of the virtual machines that are currently running on that host can also become non-responsive. However, the non-responsive host retains the lock on the virtual machine hard disk images for virtual machines it is running. Attempting to start a virtual machine on a second host and assign the second host write privileges for the virtual machine hard disk image can cause data corruption.



Fencing allows the Red Hat Virtualization Manager to assume that the lock on a virtual machine hard disk image has been released; the Manager can use a fence agent to confirm that the problem host has been rebooted. When this confirmation is received, the Red Hat Virtualization Manager can start a virtual machine from the problem host on another host without risking data corruption. Fencing is the basis for highly-available virtual machines. A virtual machine that has been marked highly-available can not be safely started on an alternate host without the certainty that doing so will not cause data corruption.

When a host becomes non-responsive, the Red Hat Virtualization Manager allows a grace period of thirty (30) seconds to pass before any action is taken, to allow the host to recover from any temporary errors. If the host has not become responsive by the time the grace period has passed, the Manager automatically begins to mitigate any negative impact from the non-responsive host. The Manager uses the fencing agent for the power management card on the host to stop the host, confirm it has stopped, start the host, and confirm that the host has been started. When the host finishes booting, it attempts to rejoin the cluster that it was a part of before it was fenced. If the issue that caused the host to become non-responsive has been resolved by the reboot, then the host is automatically set to **Up** status and is once again capable of starting and hosting virtual machines.

## 4.5. Soft-Fencing Hosts

Hosts can sometimes become non-responsive due to an unexpected problem, and though VDSM is unable to respond to requests, the virtual machines that depend upon VDSM remain alive and accessible. In these situations, restarting VDSM returns VDSM to a responsive state and resolves this issue.

"SSH Soft Fencing" is a process where the Manager attempts to restart VDSM via SSH on non-responsive hosts. If the Manager fails to restart VDSM via SSH, the responsibility for fencing falls to the external fencing agent if an external fencing agent has been configured.

Soft-fencing over SSH works as follows. Fencing must be configured and enabled on the host, and a valid proxy host (a second host, in an UP state, in the data center) must exist. When the connection between the Manager and the host times out, the following happens:

1. On the first network failure, the status of the host changes to "connecting".
2. The Manager then makes three attempts to ask VDSM for its status, or it waits for an interval determined by the load on the host. The formula for determining the length of the interval is configured by the configuration values `TimeoutToResetVdsInSeconds` (the default is 60 seconds) + `[DelayResetPerVmInSeconds (the default is 0.5 seconds)]*(the count of running virtual machines on host) + [DelayResetForSpmlnSeconds (the default is 20 seconds)] * 1` (if host runs as SPM) or 0 (if the host does not run as SPM). To give VDSM the maximum amount of time to respond, the Manager chooses the longer of the two options mentioned above (three attempts to retrieve the status of VDSM or the interval determined by the above formula).
3. If the host does not respond when that interval has elapsed, **vdsd restart** is executed via SSH.
4. If **vdsd restart** does not succeed in re-establishing the connection between the host and the Manager, the status of the host changes to **Non Responsive** and, if power management is configured, fencing is handed off to the external fencing agent.



### Note

Soft-fencing over SSH can be executed on hosts that have no power management configured. This is distinct from "fencing": fencing can be executed only on hosts that have power management configured.

## 4.6. Using Multiple Power Management Fencing Agents

Single agents are treated as primary agents. The secondary agent is valid when there are two fencing agents, for example for dual-power hosts in which each power switch has two agents connected to the same power switch. Agents can be of the same or different types.

Having multiple fencing agents on a host increases the reliability of the fencing procedure. For example, when the sole fencing agent on a host fails, the host will remain in a non-operational state until it is manually rebooted. The virtual machines previously running on the host will be suspended, and only fail over to another host in the cluster after the original host is manually fenced. With multiple agents, if the first agent fails, the next agent can be called.

When two fencing agents are defined on a host, they can be configured to use a *concurrent* or *sequential* flow:

- ✦ **Concurrent:** Both primary and secondary agents have to respond to the Stop command for the host to be stopped. If one agent responds to the Start command, the host will go up.
- ✦ **Sequential:** To stop or start a host, the primary agent is used first, and if it fails, the secondary agent is used.



## Chapter 5. Load Balancing, Scheduling, and Migration

### 5.1. Load Balancing, Scheduling, and Migration

Individual hosts have finite hardware resources, and are susceptible to failure. To mitigate against failure and resource exhaustion, hosts are grouped into clusters, which are essentially a grouping of shared resources. A Red Hat Virtualization environment responds to changes in demand for host resources using load balancing policy, scheduling, and migration. The Manager is able to ensure that no single host in a cluster is responsible for all of the virtual machines in that cluster. Conversely, the Manager is able to recognize an underutilized host, and migrate all virtual machines off of it, allowing an administrator to shut down that host to save power.

Available resources are checked as a result of three events:

- Virtual machine start - Resources are checked to determine on which host a virtual machine will start.
- Virtual machine migration - Resources are checked in order to determine an appropriate target host.
- Time elapses - Resources are checked at a regular interval to determine whether individual host load is in compliance with cluster load balancing policy.

The Manager responds to changes in available resources by using the load balancing policy for a cluster to schedule the migration of virtual machines from one host in a cluster to another. The relationship between load balancing policy, scheduling, and virtual machine migration are discussed in the following sections.

### 5.2. Load Balancing Policy

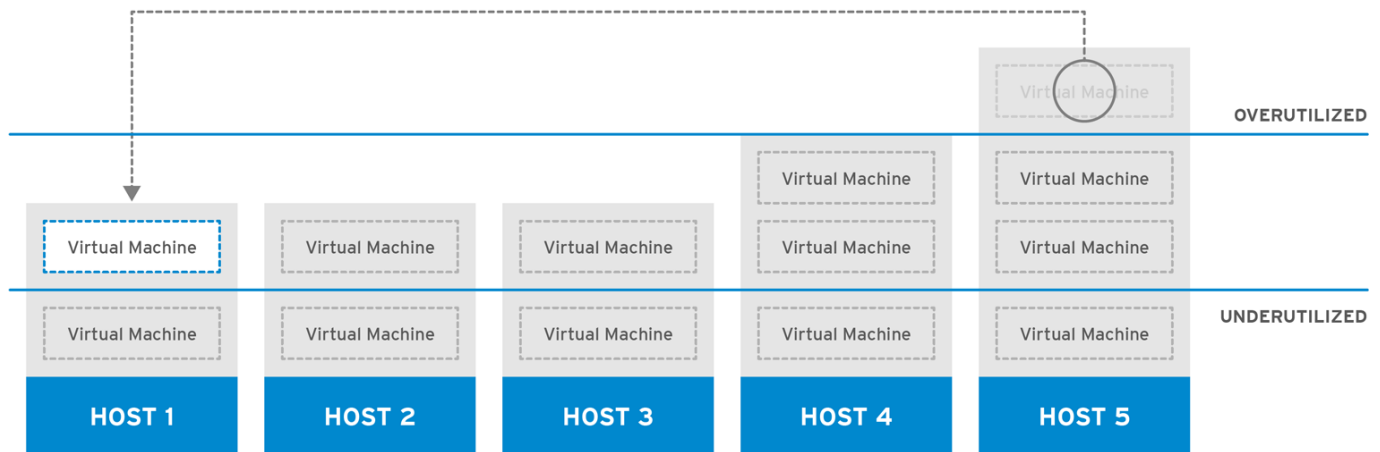
Load balancing policy is set for a cluster, which includes one or more hosts that may each have different hardware parameters and available memory. The Red Hat Virtualization Manager uses a load balancing policy to determine which host in a cluster to start a virtual machine on. Load balancing policy also allows the Manager determine when to move virtual machines from over-utilized hosts to under-utilized hosts.

The load balancing process runs once every minute for each cluster in a data center. It determines which hosts are over-utilized, which are hosts under-utilized, and which are valid targets for virtual machine migration. The determination is made based on the load balancing policy set by an administrator for a given cluster. The options for load balancing policies are **VM\_Evenly\_Distributed**, **Evenly\_Distributed**, **Power\_Saving**, and **None**.

### 5.3. Load Balancing Policy: VM\_Evenly\_Distributed

A virtual machine evenly distributed load balancing policy distributes virtual machines evenly between hosts based on a count of the virtual machines. The high virtual machine count is the maximum number of virtual machines that can run on each host, beyond which qualifies as overloading the host. The **VM\_Evenly\_Distributed** policy allows an administrator to set a high virtual machine count for hosts. The maximum inclusive difference in virtual machine count between the most highly-utilized host and the least-utilized host is also set by an administrator. The cluster is balanced when every host in the cluster has a virtual machine count that falls inside this migration threshold. The administrator also sets the number of slots for virtual machines to be reserved on SPM hosts. The SPM host will have a lower load than other hosts, so this variable defines how many fewer virtual machines than other hosts it can run. If any host is running more virtual machines than the high virtual machine count and at least one host has a virtual machine count that falls outside of the migration threshold, virtual machines are migrated one by one to the host in the cluster that has the lowest CPU utilization. One virtual machine is migrated at a time until every host in the cluster has a virtual machine count that falls within the migration threshold.

## 5.4. Load Balancing Policy: Evenly\_Distributed

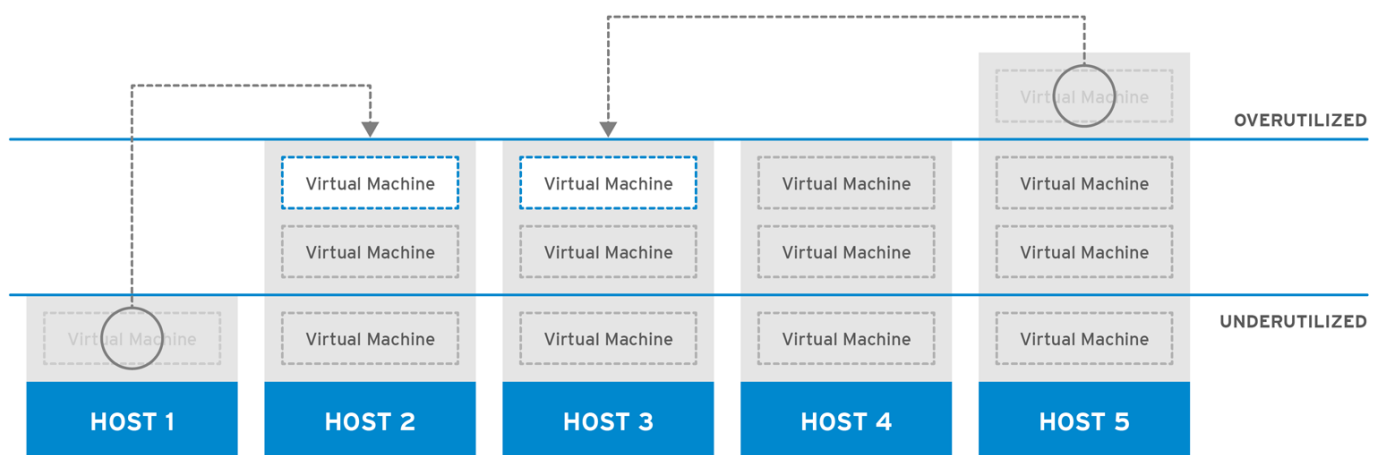


RHV\_444396\_0417

**Figure 5.1. Evenly Distributed Scheduling Policy**

An evenly distributed load balancing policy selects the host for a new virtual machine according to lowest CPU load or highest available memory. The maximum CPU load and minimum available memory that is allowed for hosts in a cluster for a set amount of time are defined by the evenly distributed scheduling policy's parameters. Beyond these limits the environment's performance will degrade. The evenly distributed policy allows an administrator to set these levels for running virtual machines. If a host has reached the defined maximum CPU load or minimum available memory and the host stays there for more than the set time, virtual machines on that host are migrated one by one to the host in the cluster that has the lowest CPU or highest available memory depending on which parameter is being utilized. Host resources are checked once per minute, and one virtual machine is migrated at a time until the host CPU load is below the defined limit or the host available memory is above the defined limit.

## 5.5. Load Balancing Policy: Power\_Saving



RHV\_444396\_0417

**Figure 5.2. Power Saving Scheduling Policy**

A power saving load balancing policy selects the host for a new virtual machine according to lowest CPU or highest available memory. The maximum CPU load and minimum available memory that is allowed for hosts in a cluster for a set amount of time is defined by the power saving scheduling policy's parameters. Beyond

these limits the environment's performance will degrade. The power saving parameters also define the minimum CPU load and maximum available memory allowed for hosts in a cluster for a set amount of time before the continued operation of a host is considered an inefficient use of electricity. If a host has reached the maximum CPU load or minimum available memory and stays there for more than the set time, the virtual machines on that host are migrated one by one to the host that has the lowest CPU or highest available memory depending on which parameter is being utilized. Host resources are checked once per minute, and one virtual machine is migrated at a time until the host CPU load is below the defined limit or the host available memory is above the defined limit. If the host's CPU load falls below the defined minimum level or the host's available memory rises above the defined maximum level the virtual machines on that host are migrated to other hosts in the cluster as long as the other host's in the cluster remain above maximum CPU load and below minimum available memory. When an under-utilized host is cleared of its remaining virtual machines, the Manager will automatically power down the host machine, and restart it again when load balancing requires or there are not enough free hosts in the cluster.

## 5.6. Load Balancing Policy: None

If no load balancing policy is selected, virtual machines are started on the host within a cluster with the lowest CPU utilization and available memory. To determine CPU utilization a combined metric is used that takes into account the virtual CPU count and the CPU usage percent. This approach is the least dynamic, as the only host selection point is when a new virtual machine is started. Virtual machines are not automatically migrated to reflect increased demand on a host.

An administrator must decide which host is an appropriate migration target for a given virtual machine. Virtual machines can also be associated with a particular host using *pinning*. Pinning prevents a virtual machine from being automatically migrated to other hosts. For environments where resources are highly consumed, manual migration is the best approach.

## 5.7. Load Balancing Policy: InClusterUpgrade

The InClusterUpgrade scheduling policy distributes virtual machines based on host operating system version. Hosts with a newer operating system than the virtual machine currently runs on are given priority over hosts with the same operating system. Virtual machines that migrate to a host with a newer operating system will not migrate back to an older operating system. A virtual machine can restart on any host in the cluster. The policy allows hosts in the cluster to be upgraded by allowing the cluster to have mixed operating system versions. Preconditions must be met before the policy can be enabled. See [Upgrading Hosts in a Cluster from Red Hat Enterprise Linux 6 to Red Hat Enterprise Linux 7](#) in the *Red Hat Enterprise Virtualization 3.6 Upgrade Guide*.



### Important

The InClusterUpgrade scheduling policy is used for upgrades between major versions only. For example, upgrading from Red Hat Enterprise Linux 6 to Red Hat Enterprise Linux 7.

## 5.8. Highly Available Virtual Machine Reservation

A highly available (HA) virtual machine reservation policy enables the Red Hat Virtualization Manager to monitor cluster capacity for highly available virtual machines. The Manager has the capability to flag individual virtual machines for High Availability, meaning that in the event of a host failure, these virtual machines will be rebooted on an alternative host. This policy balances highly available virtual machines

across the hosts in a cluster. If any host in the cluster fails, the remaining hosts can support the migrating load of highly available virtual machines without affecting cluster performance. When highly available virtual machine reservation is enabled, the Manager ensures that appropriate capacity exists within a cluster for HA virtual machines to migrate in the event that their existing host fails unexpectedly.

## 5.9. Scheduling

In Red Hat Virtualization, scheduling refers to the way the Red Hat Virtualization Manager selects a host in a cluster as the target for a new or migrated virtual machine.

For a host to be eligible to start a virtual machine or accept a migrated virtual machine from another host, it must have enough free memory and CPUs to support the requirements of the virtual machine being started on or migrated to it. A virtual machine will not start on a host with an overloaded CPU. By default, a host's CPU is considered overloaded if it has a load of more than 80% for 5 minutes, but these values can be changed using scheduling policies. If multiple hosts are eligible targets, one will be selected based on the load balancing policy for the cluster. For example, if the `Evenly_Distributed` policy is in effect, the Manager chooses the host with the lowest CPU utilization. If the `Power_Saving` policy is in effect, the host with the lowest CPU utilization between the maximum and minimum service levels will be selected. The Storage Pool Manager (SPM) status of a given host also affects eligibility as a target for starting virtual machines or virtual machine migration. A non-SPM host is a preferred target host, for instance, the first virtual machine started in a cluster will not run on the SPM host if the SPM role is held by a host in that cluster.

## 5.10. Migration

The Red Hat Virtualization Manager uses migration to enforce load balancing policies for a cluster. Virtual machine migration takes place according to the load balancing policy for a cluster and current demands on hosts within a cluster. Migration can also be configured to automatically occur when a host is fenced or moved to maintenance mode. The Red Hat Virtualization Manager first migrates virtual machines with the lowest CPU utilization. This is calculated as a percentage, and does not take into account RAM usage or I/O operations, except as I/O operations affect CPU utilization. If there are more than one virtual machines with the same CPU usage, the one that will be migrated first is the first virtual machine returned by the database query run by the Red Hat Virtualization Manager to determine virtual machine CPU usage.

Virtual machine migration has the following limitations by default:

- ✧ A bandwidth limit of 52 MiBps (megabytes per second) is imposed on each virtual machine migration.
- ✧ A migration will time out after 64 seconds per GB of virtual machine memory.
- ✧ A migration will abort if progress is stalled for 240 seconds.
- ✧ Concurrent outgoing migrations are limited to one per CPU core per host, or 2, whichever is smaller.

See <https://access.redhat.com/solutions/744423> for more details about tuning migration settings.

## Chapter 6. Directory Services

### 6.1. Directory Services

The Red Hat Virtualization platform relies on directory services for user authentication and authorization. Interactions with all Manager interfaces, including the User Portal, Administration Portal, and REST API are limited to authenticated, authorized users. Virtual machines within the Red Hat Virtualization environment can use the same directory services to provide authentication and authorization, however they must be configured to do so. The currently supported providers of directory services for use with the Red Hat Virtualization Manager are *Identity Management (IdM)*, *Red Hat Directory Server 9 (RHDS)*, *Active Directory (AD)*, and *OpenLDAP*. The Red Hat Virtualization Manager interfaces with the directory server for:

- Portal logins (User, Power User, Administrator, REST API).
- Queries to display user information.
- Adding the Manager to a domain.

Authentication is the verification and identification of a party who generated some data, and of the integrity of the generated data. A principal is the party whose identity is verified. The verifier is the party who demands assurance of the principal's identity. In the case of Red Hat Virtualization, the Manager is the verifier and a user is a principal. Data integrity is the assurance that the data received is the same as the data generated by the principal.

Confidentiality and authorization are closely related to authentication. Confidentiality protects data from disclosure to those not intended to receive it. Strong authentication methods can optionally provide confidentiality. Authorization determines whether a principal is allowed to perform an operation. Red Hat Virtualization uses directory services to associate users with roles and provide authorization accordingly. Authorization is usually performed after the principal has been authenticated, and may be based on information local or remote to the verifier.

During installation, a local, internal domain is automatically configured for administration of the Red Hat Virtualization environment. After the installation is complete, more domains can be added.

### 6.2. Local Authentication: Internal Domain

The Red Hat Virtualization Manager creates a limited, internal administration domain during installation. This domain is not the same as an AD or IdM domain, because it exists based on a key in the Red Hat Virtualization PostgreSQL database rather than as a directory service user on a directory server. The internal domain is also different from an external domain because the internal domain will only have one user: the **admin@internal** user. Taking this approach to initial authentication allows Red Hat Virtualization to be evaluated without requiring a complete, functional directory server, and ensures an administrative account is available to troubleshoot any issues with external directory services.

The **admin@internal** user is for the initial configuration of an environment. This includes installing and accepting hosts, adding external AD or IdM authentication domains, and delegating permissions to users from external domains.

### 6.3. Remote Authentication Using GSSAPI

In the context of Red Hat Virtualization, remote authentication refers to authentication that is handled remotely from the Red Hat Virtualization Manager. Remote authentication is used for user or API connections coming to the Manager from within an AD, IdM, or RHDS domain. The Red Hat Virtualization Manager must be configured by an administrator using the **engine-manage-domains** tool to be a part of an RHDS, AD, or

IdM domain. This requires that the Manager be provided with credentials for an account from the RHDS, AD, or IdM directory server for the domain with sufficient privileges to join a system to the domain. After domains have been added, domain users can be authenticated by the Red Hat Virtualization Manager against the directory server using a password. The Manager uses a framework called the *Simple Authentication and Security Layer* (SASL) which in turn uses the *Generic Security Services Application Program Interface* (GSSAPI) to securely verify the identity of a user, and ascertain the authorization level available to the user.

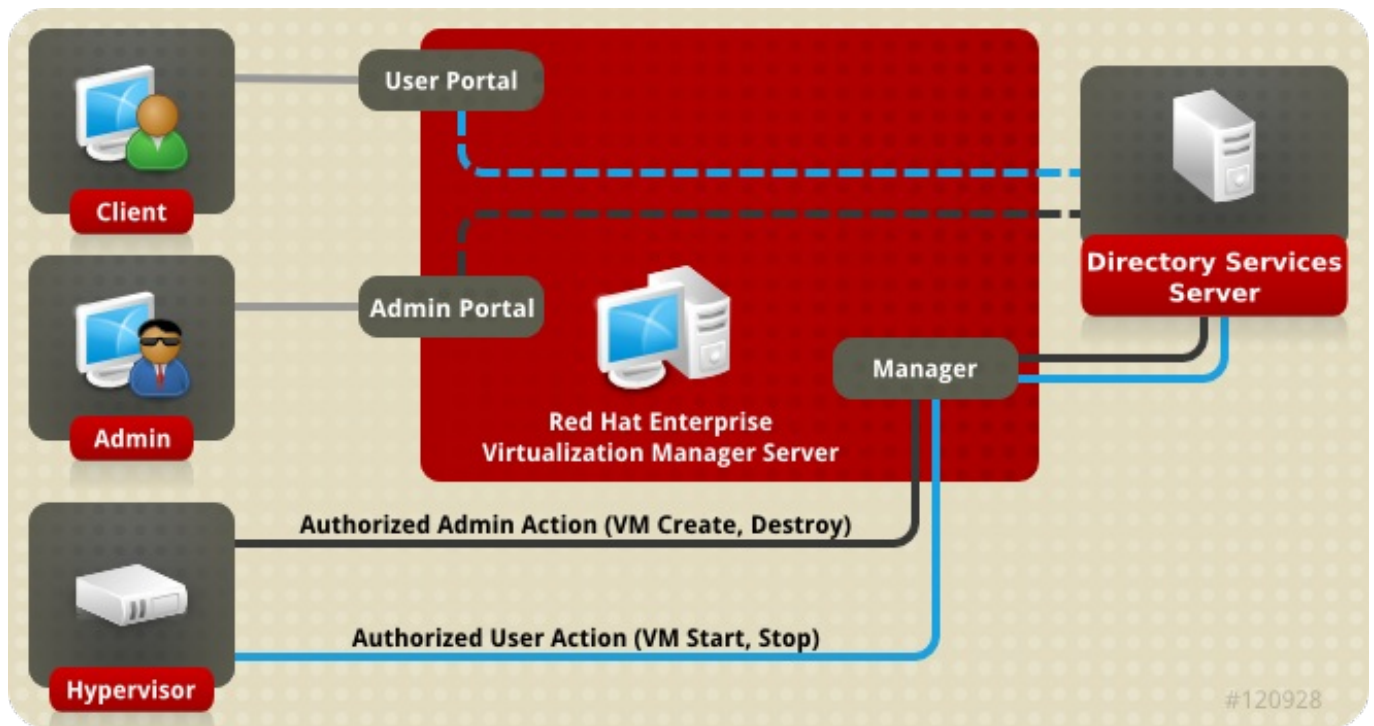


Figure 6.1. GSSAPI Authentication



## Chapter 7. Templates and Pools

### 7.1. Templates and Pools

The Red Hat Virtualization environment provides administrators with tools to simplify the provisioning of virtual machines to users. These are *templates* and *pools*. A template is a shortcut that allows an administrator to quickly create a new virtual machine based on an existing, pre-configured virtual machine, bypassing operating system installation and configuration. This is especially helpful for virtual machines that will be used like appliances, for example web server virtual machines. If an organization uses many instances of a particular web server, an administrator can create a virtual machine that will be used as a template, installing an operating system, the web server, any supporting packages, and applying unique configuration changes. The administrator can then create a template based on the working virtual machine that will be used to create new, identical virtual machines as they are required.

Virtual machine pools are groups of virtual machines based on a given template that can be rapidly provisioned to users. Permission to use virtual machines in a pool is granted at the pool level; a user who is granted permission to use the pool will be assigned any virtual machine from the pool. Inherent in a virtual machine pool is the transitory nature of the virtual machines within it. Because users are assigned virtual machines without regard for which virtual machine in the pool they have used in the past, pools are not suited for purposes which require data persistence. Virtual machine pools are best suited for scenarios where either user data is stored in a central location and the virtual machine is a means to accessing and using that data, or data persistence is not important. The creation of a pool results in the creation of the virtual machines that populate the pool, in a stopped state. These are then started on user request.

### 7.2. Templates

To create a template, an administrator creates and customizes a virtual machine. Desired packages are installed, customized configurations are applied, the virtual machine is prepared for its intended purpose in order to minimize the changes that must be made to it after deployment. An optional but recommended step before creating a template from a virtual machine is *generalization*. Generalization is used to remove details like system user names, passwords, and timezone information that will change upon deployment. Generalization does not affect customized configurations. Generalization of Windows and Linux guests in the Red Hat Virtualization environment is discussed further in [Templates](#) in the *Virtual Machine Management Guide*. Red Hat Enterprise Linux guests are generalized using **sys-unconfig**. Windows guests are generalized using **sys-prep**.

When the virtual machine that provides the basis for a template is satisfactorily configured, generalized if desired, and stopped, an administrator can create a template from the virtual machine. Creating a template from a virtual machine causes a read only copy of the specially configured virtual disk image to be created. The read only image will form the backing image for all subsequently created virtual machines that are based on that template. In other words, a template is essentially a customized read only disk image with an associated virtual hardware configuration. The hardware can be changed in virtual machines created from a template, for instance provisioning two gigabytes of RAM for a virtual machine created from a template that has one gigabyte of RAM. The template disk image, however, can not be changed as doing so would result in changes for all virtual machines based on the template.

When a template has been created, it can be used as the basis for multiple virtual machines. Virtual machines are created from a given template using a *Thin* provisioning method or a *Clone* provisioning method. Virtual machines that are cloned from templates take a complete writable copy of the template base image, sacrificing the space savings of a the thin creation method in exchange for no longer depending on the presence of the template. Virtual machines that are created from a template using the thin method use the read only image from the template as a base image, requiring that the template and all virtual machines created from it be stored on the same storage domain. Changes to data and newly generated data are stored

in a copy on write image. Each virtual machine based on a template uses the same base read only image, as well as a copy on write image that is unique to the virtual machine. This provides storage savings by limiting the number of times identical data is kept in storage. Furthermore, frequent use of the read only backing image can cause the data being accessed to be cached, resulting in a net performance increase.

## 7.3. Pools

Virtual machine pools allow for rapid provisioning of numerous identical virtual machines to users as desktops. Users who have been granted permission to access and use virtual machines from a pool receive an available virtual machine based on their position in a queue of requests. Virtual machines in a pool do not allow data persistence; each time a virtual machine is assigned from a pool, it is allocated in its base state. This is ideally suited to be used in situations where user data is stored centrally.

Virtual machine pools are created from a template. Each virtual machine in a pool uses the same backing read only image, and uses a temporary copy on write image to hold changed and newly generated data. Virtual machines in a pool are different from other virtual machines in that the copy on write layer that holds user generated and changed data is lost at shutdown. The implication of this is that a virtual machine pool requires no more storage than the template that backs it, plus some space for data generated or changed during use. Virtual machine pools are an efficient way to provide computing power to users for some tasks without the storage cost of providing each user with a dedicated virtual desktop.

### **Example 7.1. Example Pool Usage**

A technical support company employs 10 help desk staff. However, only five are working at any given time. Instead of creating ten virtual machines, one for each help desk employee, a pool of five virtual machines can be created. Help desk employees allocate themselves a virtual machine at the beginning of their shift and return it to the pool at the end.



## Chapter 8. Virtual Machine Snapshots

### 8.1. Snapshots

Snapshots are a storage function that allows an administrator to create a restore point of a virtual machine's operating system, applications, and data at a certain point in time. Snapshots save the data currently present in a virtual machine hard disk image as a COW volume and allow for a recovery to the data as it existed at the time the snapshot was taken. A snapshot causes a new COW layer to be created over the current layer. All write actions performed after a snapshot is taken are written to the new COW layer.

It is important to understand that a virtual machine hard disk image is a chain of one or more volumes. From the perspective of a virtual machine, these volumes appear as a single disk image. A virtual machine is oblivious to the fact that its disk is comprised of multiple volumes.

The term COW volume and COW layer are used interchangeably, however, layer more clearly recognizes the temporal nature of snapshots. Each snapshot is created to allow an administrator to discard unsatisfactory changes made to data *after* the snapshot is taken. Snapshots provide similar functionality to the **Undo** function present in many word processors.



#### Note

Snapshots of virtual machine hard disks marked **shareable** and those that are based on **Direct LUN** connections are not supported, live or otherwise.

The three primary snapshot operations are:

- ✦ Creation, which involves the first snapshot created for a virtual machine.
- ✦ Previews, which involves previewing a snapshot to determine whether or not to restore the system data to the point in time that the snapshot was taken.
- ✦ Deletion, which involves deleting a restoration point that is no longer required.

For task based information about snapshot operations, see [Snapshots](#) in the *Red Hat Virtualization Virtual Machine Management Guide*.

### 8.2. Live Snapshots in Red Hat Virtualization

Snapshots of virtual machine hard disks marked **shareable** and those that are based on **Direct LUN** connections are not supported, live or otherwise.

Any other virtual machine that is not being cloned or migrated can have a snapshot taken when running, paused, or stopped.

When a live snapshot of a virtual machine is initiated, the Manager requests that the SPM host create a new volume for the virtual machine to use. When the new volume is ready, the Manager uses VDSM to communicate with libvirt and qemu on the host running the virtual machine that it should begin using the new volume for virtual machine write operations. If the virtual machine is able to write to the new volume, the snapshot operation is considered a success and the virtual machine stops writing to the previous volume. If the virtual machine is unable to write to the new volume, the snapshot operation is considered a failure, and the new volume is deleted.

The virtual machine requires access to both its current volume and the new one from the time when a live snapshot is initiated until after the new volume is ready, so both volumes are opened with read-write access.

Virtual machines with an installed guest agent that supports quiescing can ensure filesystem consistency across snapshots. Registered Red Hat Enterprise Linux guests can install the **qemu-guest-agent** to enable quiescing before snapshots.

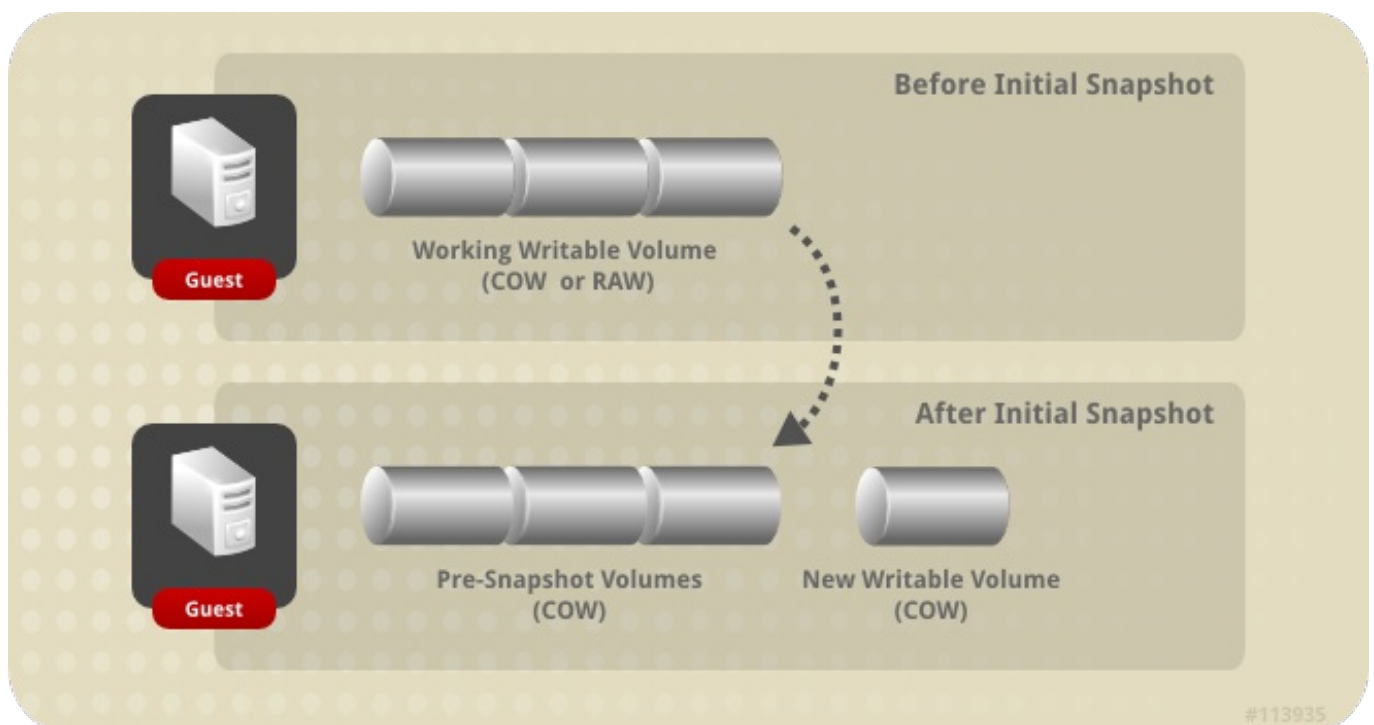
If a quiescing compatible guest agent is present on a virtual machine when it a snapshot is taken, VDSM uses libvirt to communicate with the agent to prepare for a snapshot. Outstanding write actions are completed, and then filesystems are frozen before a snapshot is taken. When the snapshot is complete, and libvirt has switched the virtual machine to the new volume for disk write actions, the filesystem is thawed, and writes to disk resume.

All live snapshots attempted with quiescing enabled. If the snapshot command fails because there is no compatible guest agent present, the live snapshot is re-initiated without the use-quiescing flag. When a virtual machine is reverted to its pre-snapshot state with quiesced filesystems, it boots cleanly with no filesystem check required. Reverting the previous snapshot using an un-quiesced filesystem requires a filesystem check on boot.

### 8.3. Snapshot Creation

In Red Hat Virtualization the initial snapshot for a virtual machine is different from subsequent snapshots in that the initial snapshot retains its format, either QCOW2 or RAW. The first snapshot for a virtual machine designates existing volumes as a base image. Additional snapshots are additional COW layers tracking the changes made to the data stored in the image since the previous snapshot.

In Red Hat Virtualization, a guest virtual machine usually interacts with a RAW disk image unless the image is created as a thinly provisioned image or the user specifically asked for it to be QCOW2. As depicted in [Figure 8.1, “Initial Snapshot Creation”](#), the creation of a snapshot causes the volumes that comprise a virtual disk image to serve as the base image for all subsequent snapshots.



**Figure 8.1. Initial Snapshot Creation**

Snapshots taken after the initial snapshot result in the creation of new COW volumes in which data that is created or changed after the snapshot is taken will be stored. Each new COW layer begins containing only

COW metadata. Data that is created through virtual machine use and operation after a snapshot is written to a new COW layer. When a virtual machine is used to modify data that exists in a previous COW layer, the data is read from the previous layer, and written into the newest layer. Virtual machines locate data by checking each COW layer from most recent to oldest, transparently to the virtual machine.

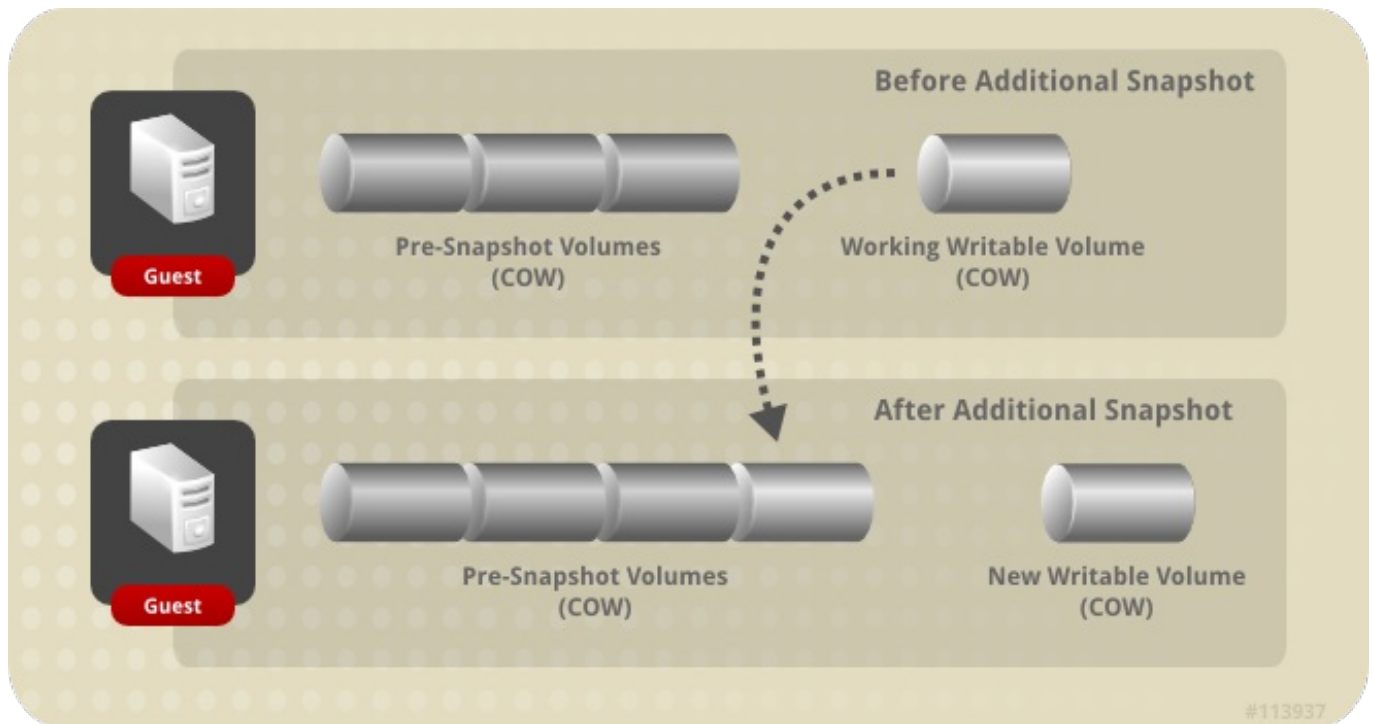


Figure 8.2. Additional Snapshot Creation

## 8.4. Snapshot Previews

To select which snapshot a virtual disk image will be reverted to, the administrator can preview all previously created snapshots.

From the available snapshots per guest, the administrator can select a snapshot volume to preview its contents. As depicted in [Figure 8.3, “Preview Snapshot”](#), each snapshot is saved as a COW volume, and when it is previewed, a new preview layer is copied from the snapshot being previewed. The guest interacts with the preview instead of the actual snapshot volume.

After the administrator previews the selected snapshot, the preview can be committed to restore the guest data to the state captured in the snapshot. If the administrator commits the preview, the guest is attached to the preview layer.

After a snapshot is previewed, the administrator can select **Undo** to discard the preview layer of the viewed snapshot. The layer that contains the snapshot itself is preserved despite the preview layer being discarded.

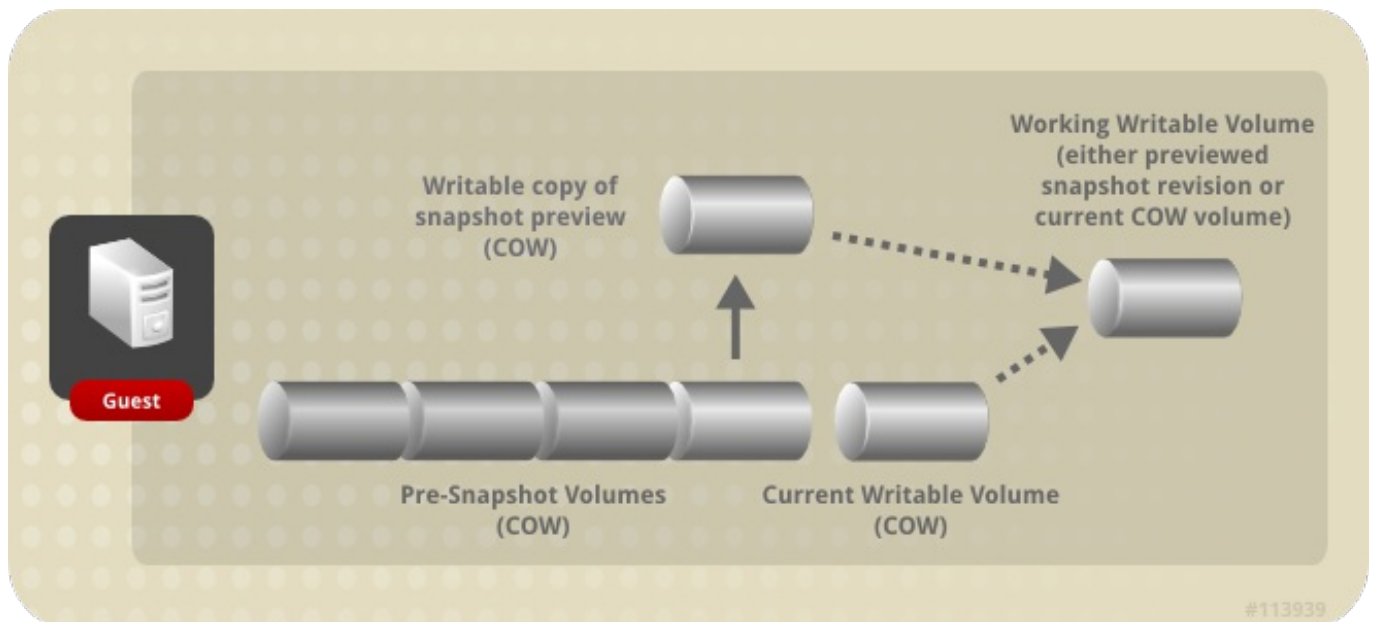


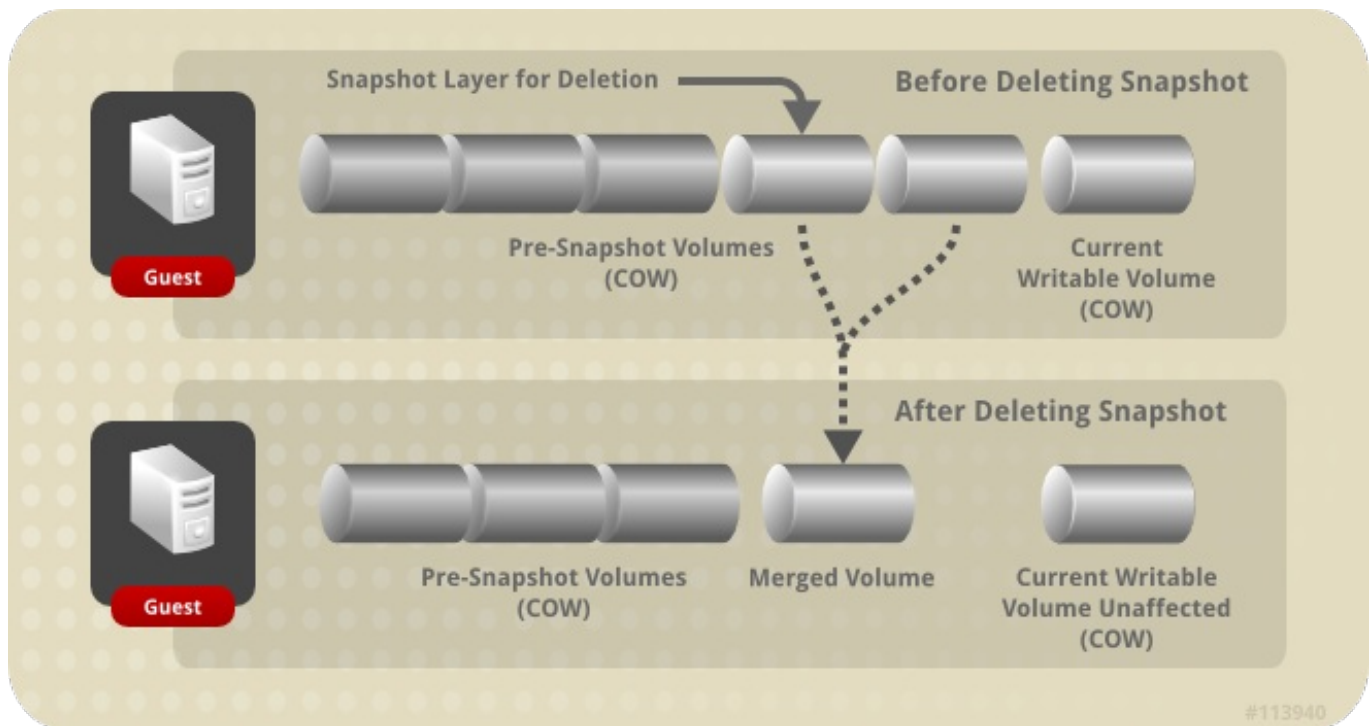
Figure 8.3. Preview Snapshot

## 8.5. Snapshot Deletion

You can delete individual snapshots or a series of snapshots that are no longer required. Deleting a snapshot removes the ability to restore a virtual disk image to that particular restoration point. It does not necessarily reclaim the disk space consumed by the snapshot, nor does it delete the data. The disk space will only be reclaimed if a subsequent snapshot has overwritten the data of the deleted snapshot. For example, if the third snapshot out of five snapshots is deleted, the unchanged data in the third snapshot must be preserved on the disk for the fourth and fifth snapshots to be usable; however, if the fourth or fifth snapshot has overwritten the data of the third, then the third snapshot has been made redundant and the disk space can be reclaimed. Aside from potential disk space reclamation, deleting a snapshot may also improve the performance of the virtual machine.

When a snapshot is selected for deletion, QEMU creates a new logical volume of the same size to merge the snapshot being deleted with the subsequent snapshot. This new logical volume is resized to accommodate all differences between the two snapshots. The new logical volume can potentially be the total combined size of the two snapshots. Once the two snapshots are merged, the subsequent snapshot is renamed and flagged for deletion and is replaced by the new logical volume, which takes its name. Both the snapshot originally flagged for deletion and its subsequent snapshot are deleted, and in their place is the single merged snapshot.

For example, snapshot **Delete\_snapshot** is 200 GB, and the subsequent snapshot, **Next\_snapshot**, is 100 GB. **Delete\_snapshot** is deleted and a new logical volume is created, temporarily named **Snapshot\_merge**, with a size of 200 GB. **Snapshot\_merge** ultimately resizes to 300 GB to accommodate the total merged contents of both **Delete\_snapshot** and **Next\_snapshot**. **Next\_snapshot** is then renamed **Delete\_me\_too\_snapshot** so that **Snapshot\_merge** can be renamed **Next\_snapshot**. Finally, **Delete\_snapshot** and **Delete\_me\_too\_snapshot** are deleted.



**Figure 8.4. Snapshot Deletion**

The logic used to delete snapshots from running virtual machines is slightly different to that for virtual machines that are shut down. Live snapshot deletion is handled as an asynchronous block job in which VDSM maintains a record of the operation in the recovery file for the virtual machine so that the job can be tracked even if VDSM is restarted or the virtual machine is shut down during the operation. Once the operation begins, the snapshot being deleted cannot be previewed or used as a restoration point, even if the operation fails or is interrupted. In operations in which the active layer is to be merged with its parent, the operation is split into a two-stage process during which data is copied from the active layer to the parent layer, and disk writes are mirrored to both the active layer and the parent. Finally, the job is considered complete once the data in the snapshot being deleted has been merged with its parent snapshot and VDSM synchronizes the changes throughout the image chain.

## Chapter 9. Hardware Drivers and Devices

### 9.1. Virtualized Hardware

Red Hat Virtualization presents three distinct types of system devices to virtualized guests. These hardware devices all appear as physically attached hardware devices to the virtualized guest but the device drivers work in different ways.

#### Emulated devices

Emulated devices, sometimes referred to as *virtual devices*, exist entirely in software. *Emulated device drivers* are a translation layer between the operating system running on the host (which manages the source device) and the operating systems running on the guests. The device level instructions directed to and from the emulated device are intercepted and translated by the hypervisor. Any device of the same type as that being emulated and recognized by the Linux kernel is able to be used as the backing source device for the emulated drivers.

#### Para-virtualized Devices

Para-virtualized devices require the installation of device drivers on the guest operating system providing it with an interface to communicate with the hypervisor on the host machine. This interface is used to allow traditionally intensive tasks such as disk I/O to be performed outside of the virtualized environment. Lowering the overhead inherent in virtualization in this manner is intended to allow guest operating system performance closer to that expected when running directly on physical hardware.

#### Physically shared devices

Certain hardware platforms allow virtualized guests to directly access various hardware devices and components. This process in virtualization is known as *passthrough* or *device assignment*. Passthrough allows devices to appear and behave as if they were physically attached to the guest operating system.

### 9.2. Stable Device Addresses in Red Hat Virtualization

Virtual hardware PCI address allocations are persisted in the ovirt-engine database.

PCI addresses are allocated by **QEMU** at virtual machine creation time, and reported to **VDSM** by **libvirt**. **VDSM** reports them back to the Manager, where they are stored in the ovirt-engine database.

When a virtual machine is started, the Manager sends **VDSM** the device address out of the database. **VDSM** passes them to **libvirt** which starts the virtual machine using the PCI device addresses that were allocated when the virtual machine was run for the first time.

When a device is removed from a virtual machine, all references to it, including the stable PCI address, are also removed. If a device is added to replace the removed device, it is allocated a PCI address by **QEMU**, which is unlikely to be the same as the device it replaced.

### 9.3. Central Processing Unit (CPU)

Each host within a cluster has a number of *virtual CPUs* (vCPUs). The virtual CPUs are in turn exposed to guests running on the hosts. All virtual CPUs exposed by hosts within a cluster are of the type selected when the cluster was initially created via Red Hat Virtualization Manager. Mixing of virtual CPU types within a cluster is not possible.



Each available virtual CPU type has characteristics based on physical CPUs of the same name. The virtual CPU is indistinguishable from the physical CPU to the guest operating system.



### Note

Support for x2APIC:

All virtual CPU models provided by Red Hat Enterprise Linux 7 hosts include support for x2APIC. This provides an *Advanced Programmable Interrupt Controller (APIC)* to better handle hardware interrupts.

## 9.4. System Devices

System devices are critical for the guest to run and cannot be removed. Each system device attached to a guest also takes up an available PCI slot. The default system devices are:

- » the host bridge,
- » the ISA bridge and USB bridge (The USB and ISA bridges are the same device),
- » the graphics card (using either the Cirrus or qxl driver), and
- » the memory balloon device.

## 9.5. Network Devices

Red Hat Virtualization is able to expose three different types of network interface controller to guests. The type of network interface controller to expose to a guest is chosen when the guest is created but is changeable from the Red Hat Virtualization Manager.

- » The **e1000** network interface controller exposes a virtualized Intel PRO/1000 (e1000) to guests.
- » The **virtio** network interface controller exposes a para-virtualized network device to guests.
- » The **rtl8139** network interface controller exposes a virtualized **Realtek Semiconductor Corp RTL8139** to guests.

Multiple network interface controllers are permitted per guest. Each controller added takes up an available PCI slot on the guest.

## 9.6. Graphics Devices

Two emulated graphics devices are provided. These devices can be connected to with the SPICE protocol or with VNC.

- » The **ac97** emulates a **Cirrus CLGD 5446 PCI VGA** card.
- » The **vga** emulates a dummy VGA card with **BochsVESA** extensions (hardware level, including all non-standard modes).

## 9.7. Storage Devices

Storage devices and storage pools can use the block device drivers to attach storage devices to virtualized guests. Note that the storage drivers are not storage devices. The drivers are used to attach a backing storage device, file or storage pool volume to a virtualized guest. The backing storage device can be any supported type of storage device, file, or storage pool volume.

- The **IDE** driver exposes an emulated block device to guests. The emulated **IDE** driver can be used to attach any combination of up to four virtualized **IDE** hard disks or virtualized **IDE** CD-ROM drives to each virtualized guest. The emulated **IDE** driver is also used to provide virtualized DVD-ROM drives.
- The **VirtIO** driver exposes a para-virtualized block device to guests. The para-virtualized block driver is a driver for all storage devices supported by the hypervisor attached to the virtualized guest (except for floppy disk drives, which must be emulated).

## 9.8. Sound Devices

Two emulated sound devices are available:

- The **ac97** emulates an **Intel 82801AA AC97 Audio** compatible sound card.
- The **es1370** emulates an **ENSONIQ AudioPCI ES1370** sound card.

## 9.9. Serial Driver

The para-virtualized serial driver (**virtio-serial**) is a bytestream-oriented, character stream driver. The para-virtualized serial driver provides a simple communication interface between the host's user space and the guest's user space where networking is not be available or unusable.

## 9.10. Balloon Driver

The balloon driver allows guests to express to the hypervisor how much memory they require. The balloon driver allows the host to efficiently allocate and memory to the guest and allow free memory to be allocated to other guests and processes.

Guests using the balloon driver can mark sections of the guest's RAM as not in use (balloon inflation). The hypervisor can free the memory and use the memory for other host processes or other guests on that host. When the guest requires the freed memory again, the hypervisor can reallocate RAM to the guest (balloon deflation).



## Chapter 10. Minimum Requirements and Technical Limitations

### 10.1. Minimum Requirements and Supported Limits

There are a number of physical and logical limitations which apply to Red Hat Virtualization environments. Environments with configurations outside of these limitations are currently not supported.

### 10.2. Resource Limitations

Certain limitations apply to resources such as storage domains and hosts.

**Table 10.1. Resource Limitations**

Item	Limitations
Storage Domains	<p>A minimum of 2 storage domains per data center is recommended:</p> <ul style="list-style-type: none"> <li>➤ A data storage domain is required.</li> <li>➤ An ISO storage domain is recommended.</li> </ul>
Hosts	Red Hat supports a maximum of 200 hosts per Red Hat Virtualization Manager.

### 10.3. Cluster Limitations

A cluster is a set of physical hosts that are treated as a resource pool for a set of virtual machines. Hosts in a cluster share the same network infrastructure and the same storage. The cluster is a migration domain within which virtual machines can be moved from host to host. To ensure stability a number of limitations apply to each cluster.

- All managed hypervisors must be in a cluster.
- All managed hypervisors within a cluster must have the same CPU type. Intel and AMD CPUs cannot co-exist within the same cluster.



#### Note

See [Clusters](#) in the *Administration Guide* for further information about clusters.

### 10.4. Storage Domain Limitations

Storage domains provide space for the storage of virtual disk images and ISO images as well as the import and export of virtual machines. While many storage domains may be created within a given data center, there are a number of limitations and recommendations that apply to each storage domain.

**Table 10.2. Storage Domain Limitations**

Item	Limitations
------	-------------

Item	Limitations
Storage Types	<p>Supported storage types are:</p> <ul style="list-style-type: none"> <li>✦ Fibre Channel Protocol (FCP)</li> <li>✦ Internet Small Computer System Interface (iSCSI)</li> <li>✦ Network File System (NFS)</li> <li>✦ POSIX Compliant File System (POSIX)</li> <li>✦ Red Hat Gluster Storage (GlusterFS)</li> </ul> <p>New ISO and export storage domains in Red Hat Virtualization 4.0 can be provided by any file-based storage (NFS, Posix or GlusterFS).</p>
Logical Unit Numbers (LUNs)	No more than 300 LUNs are permitted for each storage domain that is provided by iSCSI or FCP.
Logical Volumes (LVs)	<p>In Red Hat Virtualization, logical volumes represent virtual disks for virtual machines, templates, and virtual machine snapshots.</p> <p>No more than 350 logical volumes are recommended for each storage domain that is provided by iSCSI or FCP. If the number of logical volumes in a given storage domain exceeds this number, splitting available storage into separate storage domains with no more than 350 logical volumes each is recommended.</p> <p>The root cause of this limitation is the size of LVM metadata. As the number of logical volumes increases, the LVM metadata associated with those logical volumes also increases. When this metadata exceeds 1 MB in size, the performance of provisioning operations such as creating new disks or snapshots decreases, and lvextend operations for thinly provisioning a logical volume when running a QCOW disk take longer to run.</p> <p>Further detail about logical volumes is available in <a href="https://access.redhat.com/solutions/441203">https://access.redhat.com/solutions/441203</a>.</p>



### Note

See [Storage](#) in the *Administration Guide* for further information about storage domains.

## 10.5. Red Hat Virtualization Manager Limitations

Red Hat Virtualization Manager servers must run Red Hat Enterprise Linux 7. A number of additional hardware requirements must also be met.

**Table 10.3. Red Hat Virtualization Manager Limitations**

Item	Limitations
RAM	<ul style="list-style-type: none"> <li>➤ A minimum of 4 GB of RAM is required.</li> </ul>
PCI Devices	<ul style="list-style-type: none"> <li>➤ At least one network controller with a minimum bandwidth of 1 Gbps is recommended.</li> </ul>
Storage	<ul style="list-style-type: none"> <li>➤ A minimum of 25 GB of available local disk space is recommended.</li> </ul>



### Note

See the [Installation Guide](#) for further information about the Red Hat Virtualization Manager.

## 10.6. Hypervisor Requirements

Red Hat Virtualization Host (RHVH) has a number of hardware requirements and supported limits. The storage requirements for Red Hat Enterprise Linux hosts vary based on the amount of disk space used by their existing configuration but are expected to be greater than those of RHVH.

**Table 10.4. Red Hat Virtualization Host Requirements and Supported Limits**

Item	Support Limit
CPU	<p>A minimum of 1 physical CPU is required. Red Hat Virtualization supports the use of these CPU models in hosts:</p> <ul style="list-style-type: none"> <li>➤ AMD Opteron G1</li> <li>➤ AMD Opteron G2</li> <li>➤ AMD Opteron G3</li> <li>➤ AMD Opteron G4</li> <li>➤ AMD Opteron G5</li> <li>➤ Intel Conroe</li> <li>➤ Intel Penryn</li> <li>➤ Intel Nehalem</li> <li>➤ Intel Westmere</li> <li>➤ Intel Haswell</li> <li>➤ Intel SandyBridge Family</li> <li>➤ IBM POWER 8</li> </ul> <p>All CPUs must have support for the Intel® 64 or AMD64 CPU extensions, and the AMD-V™ or Intel VT® hardware virtualization extensions enabled. Support for the <b>No eXecute</b> flag (NX) is also required.</p>

Item	Support Limit
RAM	<p>The amount of RAM required for each virtual machine varies depending on:</p> <ul style="list-style-type: none"><li>✧ guest operating system requirements,</li><li>✧ guest application requirements, and</li><li>✧ memory activity and usage of virtual machines.</li></ul> <p>Additionally KVM is able to over-commit physical RAM for virtual machines. It does this by only allocating RAM for virtual machines as required and shifting underutilized virtual machines into swap.</p> <p>See <a href="https://access.redhat.com/articles/rhel-limits">https://access.redhat.com/articles/rhel-limits</a> for the maximum and minimum supported RAM.</p>
Storage	<p>The minimum supported internal storage for a host is the total of the following list:</p> <ul style="list-style-type: none"><li>✧ The root (/) partition requires at least 6 GB of storage.</li><li>✧ The /boot partition requires at least 1 GB of storage.</li><li>✧ The /var partition requires at least 15 GB of storage. For self-hosted engine deployment, this must be at least 60 GB.</li><li>✧ The swap partition requires at least 8 MB of storage. The recommended size of the swap partition varies depending on both the system the host is being installed upon and the anticipated level of overcommit for the environment. See <a href="https://access.redhat.com/solutions/15244">https://access.redhat.com/solutions/15244</a> for more information.</li></ul> <p>Please note that these are the <i>minimum</i> storage requirements for host installation. It is recommended to use the default allocations which use more storage space.</p>
PCI Devices	<p>At least one network controller is required with a recommended minimum bandwidth of 1 Gbps.</p>



## Important

When the Red Hat Virtualization Host boots a message may appear:

```
Virtualization hardware is unavailable.
(No virtualization hardware was detected on this system)
```

This warning indicates the virtualization extensions are either disabled or not present on your processor. Ensure that the CPU supports the listed extensions and they are enabled in the system BIOS.

To check that processor has virtualization extensions, and that they are enabled:

- ✧ At the host boot screen press any key and select the **Boot** or **Boot with serial console** entry from the list. Press **Tab** to edit the kernel parameters for the selected option. After the last kernel parameter listed ensure there is a **Space** and append the **rescue** parameter.
- ✧ Press **Enter** to boot into rescue mode.
- ✧ At the prompt which appears, determine that your processor has the virtualization extensions and that they are enabled by running this command:

```
# grep -E 'svm|vmx' /proc/cpuinfo
```

If any output is shown, the processor is hardware virtualization capable. If no output is shown it is still possible that your processor supports hardware virtualization. In some circumstances manufacturers disable the virtualization extensions in the BIOS. Where you believe this to be the case consult the system's BIOS and the motherboard manual provided by the manufacturer.

- ✧ As an additional check, verify that the **kvm** modules are loaded in the kernel:

```
# lsmod | grep kvm
```

If the output includes **kvm\_intel** or **kvm\_amd** then the **kvm** hardware virtualization modules are loaded and your system meets requirements.

## 10.7. Guest Requirements and Support Limits

The following requirements and support limits apply to guests that are run on Red Hat Virtualization Host (RHVH):

**Table 10.5. Virtualized Hardware**

Item	Limitations
CPU	A maximum of 240 virtualized CPUs per guest is supported in Red Hat Enterprise Linux 7.

Item	Limitations
RAM	<p>Different guests have different RAM requirements. The amount of RAM required for each guest varies based on the requirements of the guest operating system and the load under which the guest is operating.</p> <p>See <a href="https://access.redhat.com/articles/rhel-kvm-limits">https://access.redhat.com/articles/rhel-kvm-limits</a> for the maximum and minimum supported RAM for guest machines.</p>
PCI devices	<p>A maximum of 31 virtualized PCI devices per guest is supported. A number of system devices count against this limit, some of which are mandatory. Mandatory devices which count against the PCI devices limit include the PCI host bridge, ISA bridge, USB bridge, board bridge, graphics card, and the IDE or VirtIO block device.</p>
Storage	<p>A maximum of 28 virtualized storage devices per guest is supported, composed of a possible 3 IDE and 25 Virtio.</p>

## 10.8. SPICE Limitations

SPICE currently supports a maximum resolution of 2560x1600 pixels.

## 10.9. Additional References

These additional documentation resources do not form part of the Red Hat Virtualization documentation suite. They do, however, contain useful information for System Administrators managing Red Hat Virtualization environments and are available at <https://access.redhat.com/documentation/en/red-hat-enterprise-linux/>.

### ***Red Hat Enterprise Linux - System Administrator's Guide***

A guide to the deployment, configuration and administration of Red Hat Enterprise Linux.

### ***Red Hat Enterprise Linux - DM-Multipath Guide***

A guide to the use of Device-Mapper Multipathing on Red Hat Enterprise Linux.

### ***Red Hat Enterprise Linux - Installation Guide***

A guide to the installation of Red Hat Enterprise Linux.

### ***Red Hat Enterprise Linux - Storage Administration Guide***

A guide to the management of storage devices and file systems on Red Hat Enterprise Linux.

### ***Red Hat Enterprise Linux - Virtualization Deployment and Administration Guide***

A guide to the installation, configuration, administration and troubleshooting of virtualization technologies in Red Hat Enterprise Linux.