Red Hat Enterprise Virtualization 3.0

Technical Reference Guide

The Technical Architecture of Red Hat Enterprise Virtualization Environments

Tim Hildred

Stephen Gordon

David Jorm

Red Hat Enterprise Virtualization 3.0 Technical Reference Guide
The Technical Architecture of Red Hat Enterprise Virtualization Environments
Edition 1

Author	Tim Hildred	*thildred@redhat.com*
Author	Stephen Gordon	*sgordon@redhat.com*
Author	David Jorm	*djorm@redhat.com*

Copyright © 2011 Red Hat, Inc

This guide documents the concepts, components, and technologies used in a Red Hat Enterprise Virtualization environment. It is intended as a conceptual exploration rather than a task orientation guide. It should be read to gain a deeper understanding into how Red Hat Enterprise Virtualization works. The Technical Reference guide will cover the interactions the Red Hat Enterprise Virtualization manager and hypervisors have with existing infrastructure including directory services, storage devices and existing installations of Red Hat Enterprise Linux or previous versions of Red Hat Enterprise Virtualization.

D. Revision History   **85**

Preface

The Red Hat Enterprise Virtualization platform is a richly featured virtualization management solution providing fully integrated management across virtual machines. It is based on the leading open source virtualization platform and provides superior technical capabilities. The platform offers scalability in the management of large numbers of virtual machines.

1. About this Guide

This guide provides an in depth explanation of some of the important elements of the Red Hat Enterprise Virtualization environment. An exploratory approach is taken rather than a task oriented approach. This book should help those who are already familiar with Red Hat Enterprise Virtualization get a better understanding of the various elements that support it.

1.1. Documentation Suite

The Red Hat Enterprise Virtualization documentation suite provides information on installation, development of applications, configuration and usage of the Red Hat Enterprise Virtualization platform and its related products.

- *Red Hat Enterprise Virtualization — Administration Guide* describes how to setup, configure and manage Red Hat Enterprise Virtualization. It assumes that you have successfully installed the Red Hat Enterprise Virtualization Manager and hosts.

- *Red Hat Enterprise Virtualization — Evaluation Guide* enables prospective customers to evaluate the features of Red Hat Enterprise Virtualization. Use this guide if you have an evaluation license.

- *Red Hat Enterprise Virtualization — Installation Guide* describes the installation prerequisites and procedures. Read this if you need to install Red Hat Enterprise Virtualization. The installation of hosts, Manager and storage are covered in this guide. You will need to refer to the *Red Hat Enterprise Virtualization Administration Guide* to configure the system before you can start using the platform.

- *Red Hat Enterprise Virtualization — Manager Release Notes* contain release specific information for Red Hat Enterprise Virtualization Managers.

- *Red Hat Enterprise Virtualization — Power User Portal Guide* describes how power users can create and manage virtual machines from the Red Hat Enterprise Virtualization user portal.

- *Red Hat Enterprise Virtualization — Quick Start Guide* provides quick and simple instructions for first time users to set up a basic Red Hat Enterprise Virtualization environment.

- *Red Hat Enterprise Virtualization — REST API Guide* describes how to use the REST API to set up and manage virtualization tasks. Use this guide if you wish to develop systems which integrate with Red Hat Enterprise Virtualization, using an open and platform independent API.

- *Red Hat Enterprise Virtualization — Technical Reference Guide* (the book you are reading) describes the technical architecture of Red Hat Enterprise Virtualization and its interactions with existing infrastructure.

- *Red Hat Enterprise Virtualization — User Portal Guide* describes how users of the Red Hat Enterprise Virtualization system can access and use virtual desktops from the user portal.

- *Red Hat Enterprise Linux — Hypervisor Deployment Guide* describes how to deploy and install the Hypervisor. Read this guide if you need advanced information about installing and deploying

Hypervisors. The basic installation of Hypervisor hosts is also described in the *Red Hat Enterprise Virtualization Installation Guide*.

- *Red Hat Enterprise Linux — V2V Guide* describes importing virtual machines from KVM, Xen and VMware ESX to Red Hat Enterprise Virtualization and KVM managed by libvirt.

1.2. Audience

This documentation suite is intended for system administrators who have been administering a Red Hat Enterprise Virtualization environment and desire a deeper understanding, or solution architects looking for more information to help them plan an optimal Red Hat Enterprise Virtualization environment.

2. Document Conventions

This manual uses several conventions to highlight certain words and phrases and draw attention to specific pieces of information.

In PDF and paper editions, this manual uses typefaces drawn from the *Liberation Fonts*[1] set. The Liberation Fonts set is also used in HTML editions if the set is installed on your system. If not, alternative but equivalent typefaces are displayed. Note: Red Hat Enterprise Linux 5 and later includes the Liberation Fonts set by default.

2.1. Typographic Conventions

Four typographic conventions are used to call attention to specific words and phrases. These conventions, and the circumstances they apply to, are as follows.

`Mono-spaced Bold`

Used to highlight system input, including shell commands, file names and paths. Also used to highlight keycaps and key combinations. For example:

> To see the contents of the file `my_next_bestselling_novel` in your current working directory, enter the `cat my_next_bestselling_novel` command at the shell prompt and press `Enter` to execute the command.

The above includes a file name, a shell command and a keycap, all presented in mono-spaced bold and all distinguishable thanks to context.

Key combinations can be distinguished from keycaps by the hyphen connecting each part of a key combination. For example:

> Press `Enter` to execute the command.

> Press `Ctrl`+`Alt`+`F2` to switch to the first virtual terminal. Press `Ctrl`+`Alt`+`F1` to return to your X-Windows session.

The first paragraph highlights the particular keycap to press. The second highlights two key combinations (each a set of three keycaps with each set pressed simultaneously).

If source code is discussed, class names, methods, functions, variable names and returned values mentioned within a paragraph will be presented as above, in `mono-spaced bold`. For example:

---

[1] https://fedorahosted.org/liberation-fonts/

> File-related classes include **filesystem** for file systems, **file** for files, and **dir** for directories. Each class has its own associated set of permissions.

Proportional Bold

This denotes words or phrases encountered on a system, including application names; dialog box text; labeled buttons; check-box and radio button labels; menu titles and sub-menu titles. For example:

> Choose **System** → **Preferences** → **Mouse** from the main menu bar to launch **Mouse Preferences**. In the **Buttons** tab, click the **Left-handed mouse** check box and click **Close** to switch the primary mouse button from the left to the right (making the mouse suitable for use in the left hand).

> To insert a special character into a **gedit** file, choose **Applications** → **Accessories** → **Character Map** from the main menu bar. Next, choose **Search** → **Find…** from the **Character Map** menu bar, type the name of the character in the **Search** field and click **Next**. The character you sought will be highlighted in the **Character Table**. Double-click this highlighted character to place it in the **Text to copy** field and then click the **Copy** button. Now switch back to your document and choose **Edit** → **Paste** from the **gedit** menu bar.

The above text includes application names; system-wide menu names and items; application-specific menu names; and buttons and text found within a GUI interface, all presented in proportional bold and all distinguishable by context.

Mono-spaced Bold Italic or ***Proportional Bold Italic***

Whether mono-spaced bold or proportional bold, the addition of italics indicates replaceable or variable text. Italics denotes text you do not input literally or displayed text that changes depending on circumstance. For example:

> To connect to a remote machine using ssh, type **ssh *username@domain.name*** at a shell prompt. If the remote machine is **example.com** and your username on that machine is john, type **ssh john@example.com**.

> The **mount -o remount *file-system*** command remounts the named file system. For example, to remount the **/home** file system, the command is **mount -o remount /home**.

> To see the version of a currently installed package, use the **rpm -q *package*** command. It will return a result as follows: ***package-version-release***.

Note the words in bold italics above — username, domain.name, file-system, package, version and release. Each word is a placeholder, either for text you enter when issuing a command or for text displayed by the system.

Aside from standard usage for presenting the title of a work, italics denotes the first use of a new and important term. For example:

> Publican is a *DocBook* publishing system.

2.2. Pull-quote Conventions

Terminal output and source code listings are set off visually from the surrounding text.

Output sent to a terminal is set in **mono-spaced roman** and presented thus:

```
books        Desktop    documentation  drafts  mss    photos   stuff  svn
books_tests  Desktop1  downloads              images  notes  scripts  svgs
```

Source-code listings are also set in **mono-spaced roman** but add syntax highlighting as follows:

```java
package org.jboss.book.jca.ex1;

import javax.naming.InitialContext;

public class ExClient
{
   public static void main(String args[])
      throws Exception
   {
      InitialContext iniCtx = new InitialContext();
      Object         ref    = iniCtx.lookup("EchoBean");
      EchoHome       home   = (EchoHome) ref;
      Echo           echo   = home.create();

      System.out.println("Created Echo");

      System.out.println("Echo.echo('Hello') = " + echo.echo("Hello"));
   }
}
```

2.3. Notes and Warnings

Finally, we use three visual styles to draw attention to information that might otherwise be overlooked.

Note

Notes are tips, shortcuts or alternative approaches to the task at hand. Ignoring a note should have no negative consequences, but you might miss out on a trick that makes your life easier.

Important

Important boxes detail things that are easily missed: configuration changes that only apply to the current session, or services that need restarting before an update will apply. Ignoring a box labeled 'Important' will not cause data loss but may cause irritation and frustration.

Warning

Warnings should not be ignored. Ignoring warnings will most likely cause data loss.

3. We Need Feedback!

If you find a typographical error in this manual, or if you have thought of a way to make this manual better, we would love to hear from you! Please submit a report in Bugzilla: *http://bugzilla.redhat.com/* against the product **Red Hat Enterprise Virtualization Hypervisor**.

When submitting a bug report, be sure to mention the manual's identifier: *Technical_Reference_Guide*.

If you have a suggestion for improving the documentation, try to be as specific as possible when describing it. If you have found an error, include the section number and some of the surrounding text so we can find it easily.

Introducing Red Hat Enterprise Virtualization

Red Hat Enterprise Virtualization provides a full-featured virtualization platform and the tools required to efficiently manage it. This chapter introduces the various virtualization technologies, applications and features and explains how they work. The purpose of this chapter is to assist Red Hat Enterprise Virtualization users in understanding virtualization.

Red Hat Enterprise Virtualization Hypervisor

> The *Red Hat Enterprise Virtualization Hypervisor* is a compact, full-featured virtualization platform for quickly and easily deploying and managing virtualized guests. The Hypervisor is a minimal installation of Red Hat Enterprise Linux designed specifically to support virtualization workloads. You can manage hypervisors with the Red Hat Enterprise Virtualization Manager.
>
> Full virtualization is provided by using a loadable Linux kernel module called *Kernel-based Virtual Machine* (KVM). KVM can concurrently host multiple virtualized guests running either Windows or Linux operating systems. Virtualized guests run as individual Linux processes on the host machine and are managed remotely using the Red Hat Enterprise Virtualization Manager.

Red Hat Enterprise Virtualization Manager

> Red Hat Enterprise Virtualization Manager is a centralized management system that allows system administrators to view and manage virtual machines and images. The Red Hat Enterprise Virtualization Manager provides a comprehensive range of features including search capabilities, resource management, live migrations, and provisioning. The Red Hat Enterprise Virtualization Manager runs on Red Hat Enterprise Linux 6.
>
> The manager provides you with a graphical user interface which you can use to manager the physical and logical resources of the virtual environment. The graphical user interface (GUI) can be used to manage provisioning, connection protocols, user sessions, virtual machine pools, images, and virtual machine high availability. You can interact with the Red Hat Enterprise Virtualization Manager by using an Administration Portal, a User Portal, and an Application Programming Interface (API).
>
> * The Administration Portal is used to set up, configure and manage the Red Hat Enterprise Virtualization environment.
>
> * The User Portal is used to start, stop, reboot, and connect to virtual machines. Users can be granted "power user" access by the environment's administrators. Power users are allowed to create virtual machine templates and virtual machines from this interface.
>
> * The REST API provides an interface for automation of tasks normally accomplished manually by users. Scripts that make use of the REST API can be written in any language which supports accessing HTTP and HTTPS resources.

Architecture

A Red Hat Enterprise Virtualization environment consists of:

• The Red Hat Enterprise Virtualization Manager,

• one or more Red Hat Enterprise Virtualization Hypervisor hosts, or Red Hat Enterprise Linux virtualization hosts,

• the associated backing storage,

• and the networking that facilitates communication between all elements of the Red Hat Enterprise Virtualization environment.

Each of these entities consists of a number of components which are transparent to users and administrators. These components interact to enable the efficient virtualization of workloads using Red Hat Enterprise Virtualization.

2.1. Red Hat Enterprise Virtualization Manager

The Red Hat Enterprise Virtualization Manager provides centralized management for a virtualized environment. You can use a number of different interfaces to access the Red Hat Enterprise Virtualization Manager. Each interface facilitates access to the virtualized environment in a different manner.
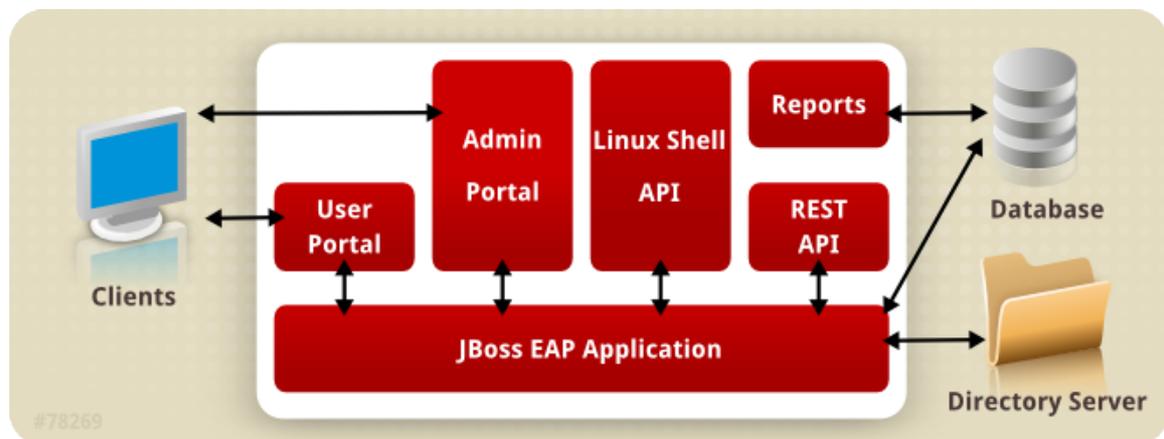
Figure 2.1. Red Hat Enterprise Virtualization Manager Architecture

The Red Hat Enterprise 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 *JBoss Enterprise Application Platform*. There are a number of other components which support the Red Hat Enterprise Virtualization Manager in addition to JBoss Enterprise Application Platform.

2.1.1. Interfaces for Accessing The Manager

User portal

   The user portal is the primary method of delivering *Virtual Desktop Infrastructure* to end users. Desktop virtualization provides users with a desktop environment that is similar a personal computer's desktop environment. You can access the user portal through a web browser which will allow you to display and access a user's 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 *User Portal Guide*.

- **Power User Access**

  The Red Hat Enterprise Virtualization User Portal provides power users with a graphical user interface that enables them to connect to, manage, and monitor virtual resources. Power users can connect to multiple virtual machines using any web browser. The Power User Portal allows system administrators to delegate some administration tasks. For example, power users can manage virtual resources that have been assigned to them. In addition to the tasks that can 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.

  The power user portal allows virtual machine administration tasks to be delegated. It saves tasks at the data center level for the environment administrator.

Administration portal

The Administration Portal is the graphical administration interface of the Red Hat Enterprise Virtualization Manager server. It allows administrators to monitor, create, and maintain all elements of the virtualized environment using their 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 Enterprise Virtualization user and permission management.

The administration portal is displayed using the *Windows Presentation Foundation (WPF)*. **Windows Presentation Foundation (WPF)** is a presentation layer currently only available on the Microsoft Windows platform. It uses vector graphics to render and manipulate user interfaces and the screen elements they contain. Because Red Hat Enterprise Virtualization relies on the use of

WPF, the Administration Portal can currently only be accessed from machines which run Microsoft Windows.

Administration Portal functions are discussed in further detail in the *Red Hat Enterprise Virtualization Administration Guide*. Information on the browsers and platforms that are supported by the Administration Portal can be found in the *Red Hat Enterprise Virtualization Installation Guide*.

Representational State Transfer (REST) API

The Red Hat Enterprise Virtualization REST API provides a software interface for the interrogation and control of the Red Hat Enterprise Virtualization environment. The REST API ensures that scripts that interact with the Red Hat Enterprise Virtualization Manager are not restricted to specific programming languages or platforms. The REST API can be used by any programming language that supports HTTP actions.

The REST API provides developers and administrators with the ability to:

- 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 Enterprise Virtualization environment.

See the *REST API Guide* for the API specification and usage examples.

2.1.2. Components that Support the Manager

JBoss Enterprise Application Platform

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

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

Gathering Reports and Historical Data

Red Hat Enterprise Virtualization Manager includes a data warehouse that collects monitoring data for 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. Refer to *Section 9.2, "Configuration History Views"* and *Section 9.1, "Statistics History Views"* for more information.

The Red Hat Enterprise Virtualization Manager installation program creates two databases. These databases are created on the Postgres instance selected during installation.

- The **rhevm** database is the primary data store used by Red Hat Enterprise Virtualization Manager. Information about the virtualization environment such as its state, configuration, and performance are stored in this database.

- The **rhevm_history** database contains configuration information and statistical metrics which are collated over time from the **rhevm** operational database. The configuration data in the **rhevm** database is examined every minute, and changes are replicated to the **rhevm_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 Enterprise Virtualization environment and resolve difficulties.

  For more information on generating reports based on the **rhevm_history** database see the *Red Hat Enterprise Virtualization Administration Guide*.

  > **RHEVM History Service**
  >
  > The replication of data in the rhevm_history database is performed by the **RHEVM History Service**. This service must be manually configured to start automatically in the service Manager before building reports.

Directory services
    Directory services provide a centralized network-based registry for the storage of information. Types of information stored include application settings, user profiles, group data, policies, and access control. Red Hat Enterprise Virtualization Manager has traditionally relied on directory services provided by Active Directory. Beginning in Red Hat Enterprise Virtualization 3.0, there are now several options choices of directory service provider. If you are already using Active Directory for authentication you can continue to do so. Alternatively you can use IPA for directory services including authentication of users as well as retrieving and maintaining their access controls. There is also a local, internal domain for administration purposes only. This internal domain has only one user: the admin user.

    See *Chapter 7, Directory Services* for more information on Directory Services.

2.2. Red Hat Virtualization Hypervisor

A Red Hat Enterprise 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 Enterprise Virtualization Hypervisor hosts run 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 customized 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 referred to as *hosts*.
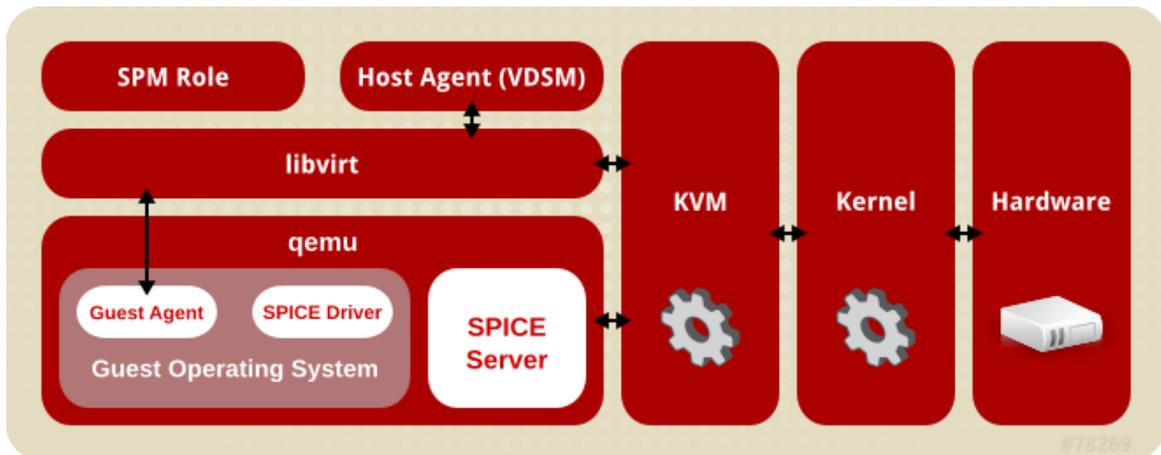
Figure 2.2. Host Architecture

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.
>
>   For information on the other devices available to guests by QEMU see *Appendix C, Virtualized Hardware*.

Red Hat Enterprise Virtualization Manager Host Agent, VDSM
>   In Red Hat Enterprise Virtualization, VDSM performs actions on virtual machines, storage. It also facilitates inter-host communication. VDSM monitors virtualized hosts' resources such as memory, storage, and networking. Additionally, VDSM manages host administration tasks such as virtual machine creation, statistics accumulation, and log collection. A VDSM instance runs on each host and receives management operation information from the Red Hat Enterprise Virtualization Manager. The Red Hat Enterprise Virtualization Manager uses VDSM to manage host administration tasks such as virtual machine creation, statistics accumulation, and log collection.
>
>   The Red Hat Enterprise Virtualization Manager uses VDSM as the host management module and establishes communication with it over the re-configurable port **54321**.

VDSM-REG
>   VDSM uses `VDSM-REG` to register each host with the Red Hat Enterprise 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 the infrastructure that supports them. When Red Hat Enterprise Virtualization Manager is used to initiate virtual machine life-cycle commands (start, stop, reboot), VDSM invokes libvirt on the relevant host machines to execute these host life-cycle commands.

Storage Pool Manager, SPM

> The Storage Pool Manager (SPM) is a role assigned to one host in a data center, which gives the SPM host 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 Enterprise 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.
>
> See *Section 3.2, "Role: The Storage Pool Manager"* for more information on the Storage Pool Manager.

Guest Operating System

> Guest operating systems can be installed in a Red Hat Enterprise Virtualization environment without modification. The guest operating system, and any applications on the guest, are not aware of the virtualized environment and run normally. However, device drivers that allow faster and more efficient access to virtualized devices are available and can be installed inside the guest. You can also install the Red Hat Enterprise Virtualization Agent on guests, which provides enhanced guest information to the management console.

2.3. Storage

Red Hat Enterprise Virtualization uses a centralized storage system for virtual machine disk images, templates, snapshots, and ISO files. Storage is logically grouped into storage pools, which are comprised of storage domains. A storage domain combines storage capacity that contains images with 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 most important and is the only one required by each data center. The 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), or Fibre Channel Protocol (FCP). A storage domain can consist of block devices (SAN - iSCSI or FCP) or files (NAS - NFS).

On NFS, all virtual disks, templates, and snapshots are simple files. On SAN (iSCSI/FCP), block devices are aggregated into a logical entity called a Volume Group (VG). This is done using the Logical Volume Manager (LVM). See *Red Hat Enterprise Linux Logical Volume Manager Administration Guide* for more information on LVM. Each virtual disk, template or snapshot is a Logical Volume (LV) on the VG.
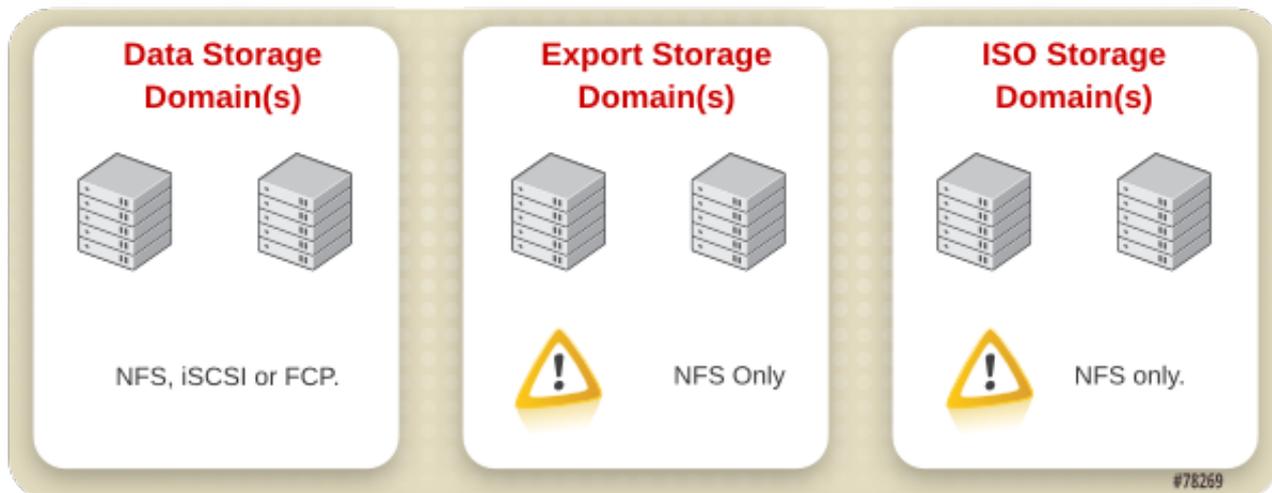
Figure 2.3. Storage Architecture

Data storage domain

> Data domains hold the disk images of all the virtual machines running in the environment, including installed operating system images, and data disks. Snapshots of the virtual machines are also stored in the data domain. A data domain cannot be shared across data centers, and the data domain must be of the same type as the data center. For example, a data center of a iSCSI type, must have an iSCSI data domain.

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 Enterprise 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.

> See *Chapter 3, Storage Architecture* for more information on the Red Hat Enterprise Virtualization storage architecture.

2.4. Network

The Red Hat Enterprise Virtualization network architecture facilitates connectivity between the different elements of the Red Hat Enterprise Virtualization environment. This includes communication between the Red Hat Enterprise Virtualization Manager and hosts, communication between individual hosts, and storage communication between hosts and network attached storage. The network architecture also facilitates connectivity among virtual machines, communication between virtual machines and network attached storage, and communication between virtual machine users or clients and their virtual machines. The Red Hat Enterprise Virtualization network architecture also allows optional connectivity to destinations and objects that are external to the Red Hat Enterprise Virtualization environment.

The Red Hat Enterprise Virtualization network architecture not only supports network connectivity, it also allows for network segregation. Hosts in separate clusters can be isolated from each other, as

can virtual machines hosted in separate clusters. Virtual machines used for specific purposes can connect to special purpose networks, and can be isolated from general purpose virtual machines. Network traffic can also be segregated by traffic type. Storage traffic and display traffic can be carried on separate networks, for example.
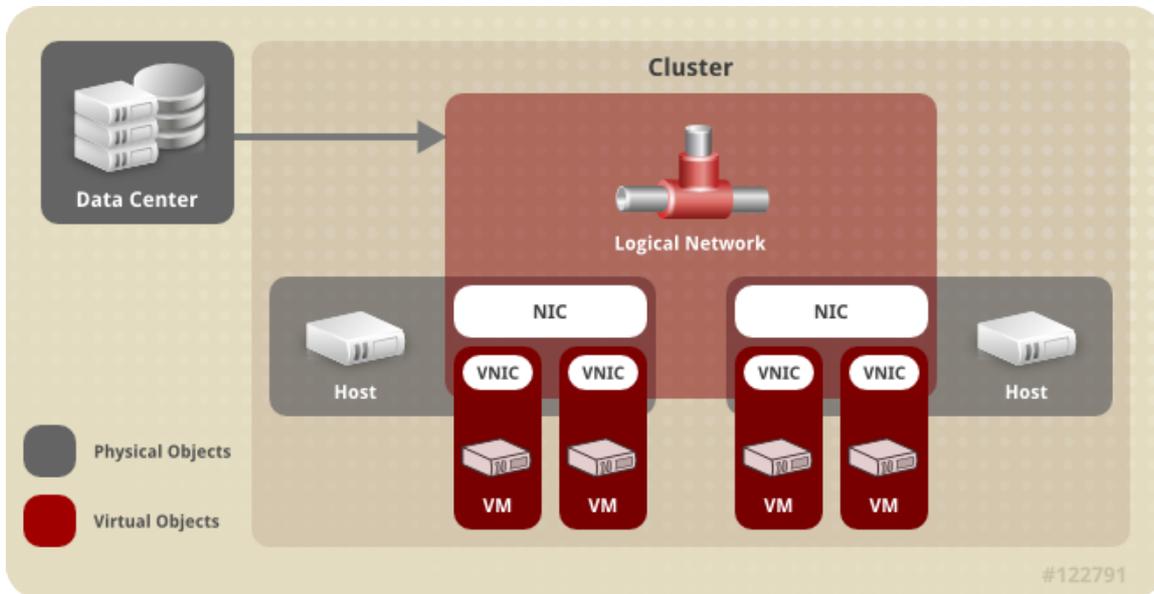
Figure 2.4. Network Architecture

In order to support all these networking possibilities, networking is defined in Red Hat Enterprise 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 Enterprise Virtualization environment.

Networking Infrastructure

The Red Hat Enterprise 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 or bridge.

- *Bridges* are a packet-forwarding technique for packet-switching networks. They form the basis of *logical networks*.

Logical Networks

Logical networks allow segregation of network traffic based on environment requirements. A logical network 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 Enterprise Virtualization Manager: the rhevm Management network. Other logical networks that can be added by an administrator are: a dedicated storage logical network, and a dedicated display logical network.

Data Center Layer

Logical networks are defined at the data center level. Each data center has a management network. Further logical networks are optional but recommended. IP address, gateway, subnet

mask, and VLAN tagging can be set at the data center level, but at this level the network is purely logical. 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 logical network has been added to a cluster, it must be implemented for each host in the cluster.

Host Layer

Logical networks are implemented for each host in a cluster as a software bridge device associated with a physical NIC. 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 Enterprise Virtualization environment. Further logical networks that have been added to a cluster must be associated with NICs 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 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 2.1. Management Network
>
> The management logical network, named **rhevm**, is created automatically when the Red Hat Enterprise Virtualization Manager is installed. The **rhevm** network is dedicated to management traffic between the Red Hat Enterprise Virtualization Manager and hosts. If no other specifically-purposed bridges are set up, **rhevm** is the default bridge for all traffic.

See *Chapter 4, Network Architecture* for more information on the Red Hat Enterprise Virtualization network architecture.

Storage Architecture

In this chapter, the Red Hat Enterprise Virtualization storage architecture is divided into components, roles, attributes, and functionalities. The storage components are the logical building blocks of the architecture. The storage pool manager is a role held by a host that empowers it to manage structural changes to a data domain. Storage attributes are applied to virtual machine image storage by system administrators to optimize their environment for their use case. Storage functionalities are storage related features or processes.

3.1. Storage Components

The storage components covered in this section; data centers and storage domains, are fundamental to the Red Hat Enterprise Virtualization storage architecture. It is the interactions between these components that provides users with a robust and flexible virtualization environment.

3.1.1. Data Centers

A data center is the highest level of abstraction in Red Hat Enterprise 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 supported 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 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.

3.1.2. Storage Domains

Red Hat Enterprise Virtualization supports two storage types:

- File based storage

- Block storage

Red Hat Enterprise Virtualization uses a centralized storage system that can be implemented with NFS or FCP. FCP includes storage accessed using iSCSI, FCoE, and SAS.

File Based Storage
    The file based storage types supported by Red Hat Enterprise Virtualization are NFS and file systems on the local storage of a Red Hat Enterprise Virtualization host. File based storage allows an entity external to the Red Hat Enterprise Virtualization environment to manage the file system. In the case of NFS storage, this could be a Red Hat Enterprise Linux NFS server, or other third party network attached storage server. Red Hat Enterprise Virtualization hosts are capable of managing their own local storage and file systems. Hosts interact with files, either local or networked, as if they were present in local storage.

Block Based Storage

> Block storage makes use of un-formatted block devices, for example un-partitioned, un-formatted hard drives. Block devices are aggregated into *volume groups* by the Logical Volume Manager (LVM). An instance of LVM runs on all hosts and each LVM instance is unaware of instances running on other hosts. VDSM adds clustering logic on top of LVM by scanning volume groups for changes, and updating individual hosts by refreshing volume group information. A volume group is presented to Red Hat Enterprise Virtualization hosts as logical volumes for use with virtual machines. If more storage capacity is added to an existing storage domain, the Red Hat Enterprise Virtualization Manager causes VDSM on each host to refresh volume group information.

LUNs

> A *Logical Unit Number* (LUN) is an individual block device. To connect to a LUN, one of the supported block storage protocols, iSCSI, FCoE, or SAS, can be used. Selecting iSCSI causes the Red Hat Enterprise Virtualization Manager to manage software iSCSI connections to storage to gain access to the LUNs. All other block storage connections are managed externally to the Red Hat Enterprise 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. Changes are then synced by VDSM which initiates LVM refreshes across all hosts in the cluster.

The Red Hat Enterprise Virtualization Manager selects one host as the Storage Pool Manager, designating it as the host responsible for writing metadata about the structure of the data storage domain. In order to get the SPM role, a host must acquire a storage-centric lease that acts as a *mutex* (or mutual exclusion). The mutex ensures that only one host can become SPM. Changing volume group information from multiple hosts can lead to data corruption, so the mutex function limits write access to data domain structure metadata to a single host at any given time. Mutex is essential in selection and operations of the storage pool manager (refer to *Section 3.2, "Role: The Storage Pool Manager"*.
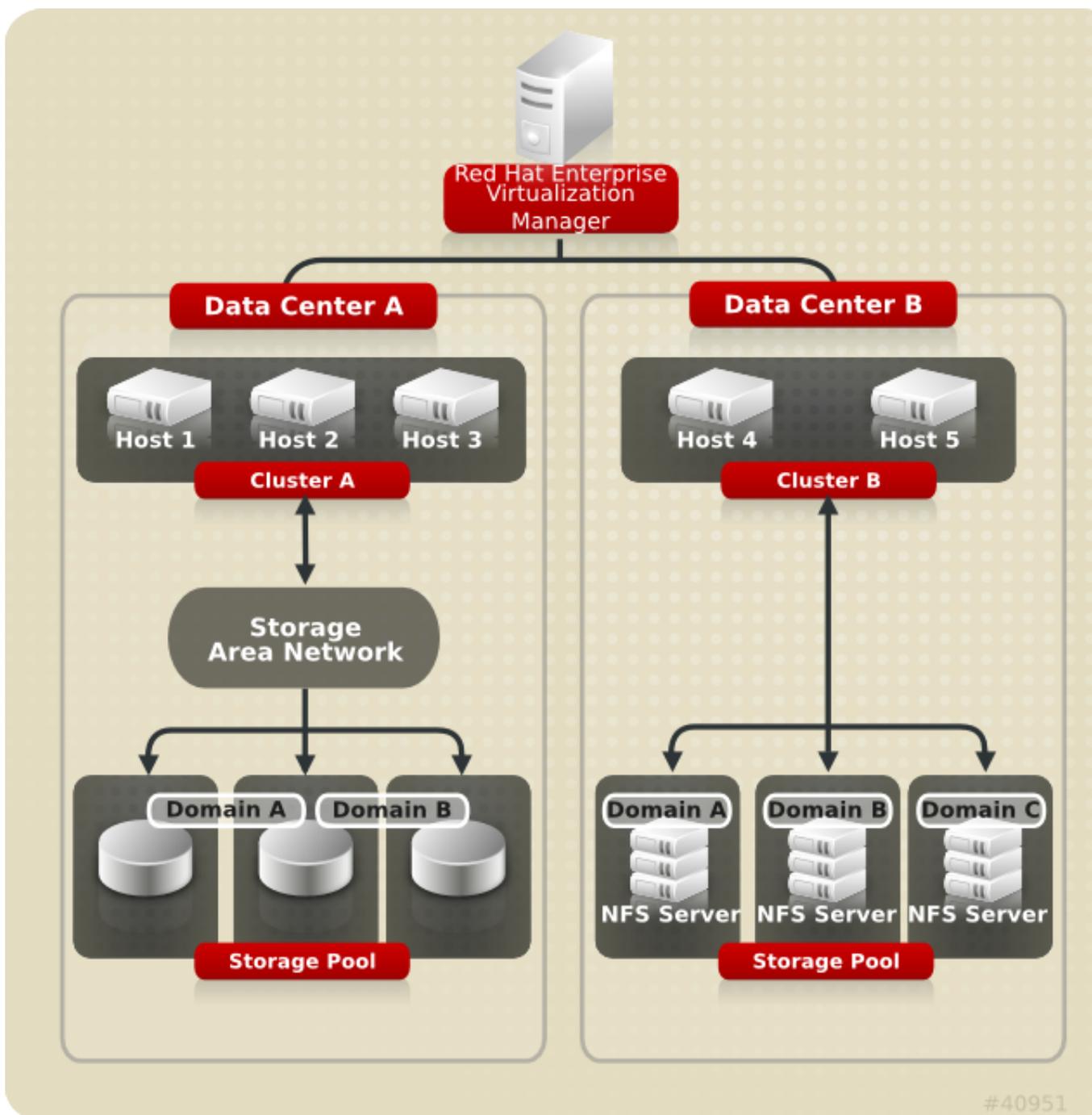
Figure 3.1. Red Hat Enterprise Virtualization Block Storage vs. File Storage

Figure 3.1, "Red Hat Enterprise Virtualization Block Storage vs. File Storage" displays the similarities between block storage (data center A) and file based storage (data center B) in Red Hat Enterprise Virtualization. A storage area network is much like a regular network, over FCP instead of Ethernet, with a special SAN type switch instead of an IP switch. The storage used by Data Center B is referred to as Network Attached Storage (NAS). There is no logical difference in reaching either storage type. The difference between file and block storage is that a file server runs a file system and provides hosts with file level access while a block storage server provides access to un-formatted, raw storage and leaves volume management to Red Hat Enterprise Virtualization hosts and file system creation to virtual machine operation systems.
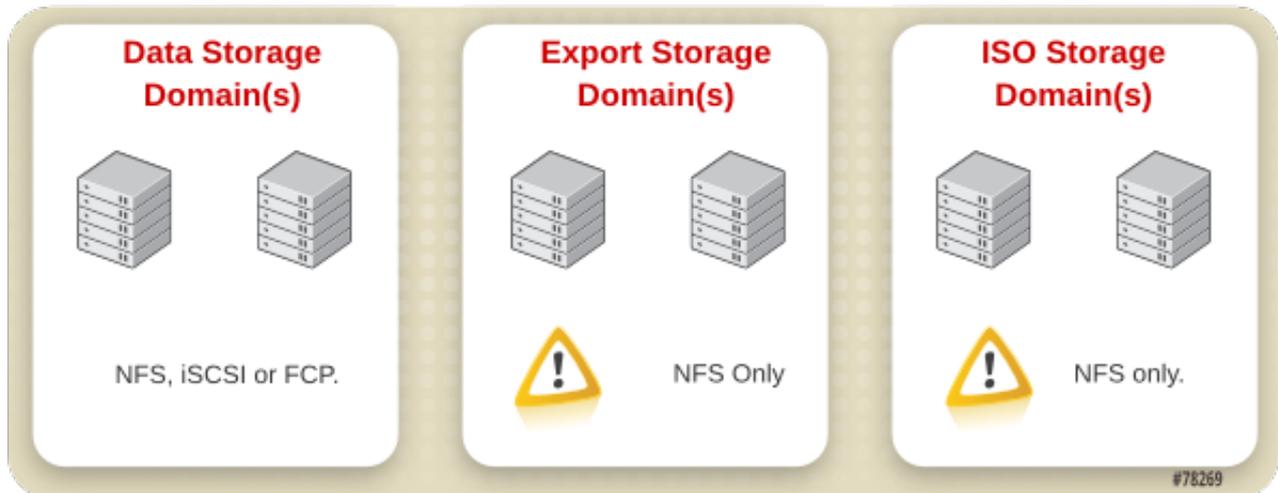
Figure 3.2. Storage Types

Figure 3.2, "Storage Types" displays the three types of storage domains and the storage types each storage domain supports, which are:

- The *Data Storage Domain* stores the hard disk images of all virtual machines in the Red Hat Enterprise Virtualization environment. All disks for a given virtual machine must reside on the same data storage domain. Disk images may contain an installed operating system or data stored or generated by a virtual machine. As depicted in *Figure 3.2, "Storage Types"*, data storage domains support NFS, iSCSI and FCP storage. A data domain cannot be shared between multiple data centers. Additionally, it is required that the data center and data storage domain use the same protocol (for example, both must be iSCSI based).

- 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. As depicted in *Figure 3.2, "Storage Types"*, 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 Enterprise 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. As depicted in *Figure 3.2, "Storage Types"*, ISO storage domains only support NFS storage. ISO storage domains allow all hosts within the data center to share ISOs, eliminating the need for physical optical media.

3.2. Role: The Storage Pool Manager

Red Hat Enterprise Virtualization hosts deal with storage domain structure related metadata (image/snapshot creation, image/snapshot deletion, and volume/domain extension) based on a single writer and multiple readers configuration. The Red Hat Enterprise Virtualization host that can can make changes to the structure of the data domain is known as the Storage Pool Manager. All hosts can read structural metadata but only the storage pool manager can write domain structure metadata for the data center. The storage pool manager 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.

To assign the storage pool manager role, the Red Hat Enterprise Virtualization Manager causes a potential SPM host to attempt to assume a *storage-centric lease*. The Manager issues 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 retains the storage-centric lease until the Red Hat Enterprise Virtualization Manager requests the a new host assume the role of SPM. This will happen if:

• The SPM host can not access all storage domains, but can access the master storage domain.

• The 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 host crashes.

VDSM on a host uses a distributed algorithm to take a mutex (storage-centric lease) on the storage pool to ensure that it is the only host anywhere that is the SPM. This algorithm is storage based; it does not communicate with other hosts through the network. Mutex communications are written to a special logical volume in the data 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.
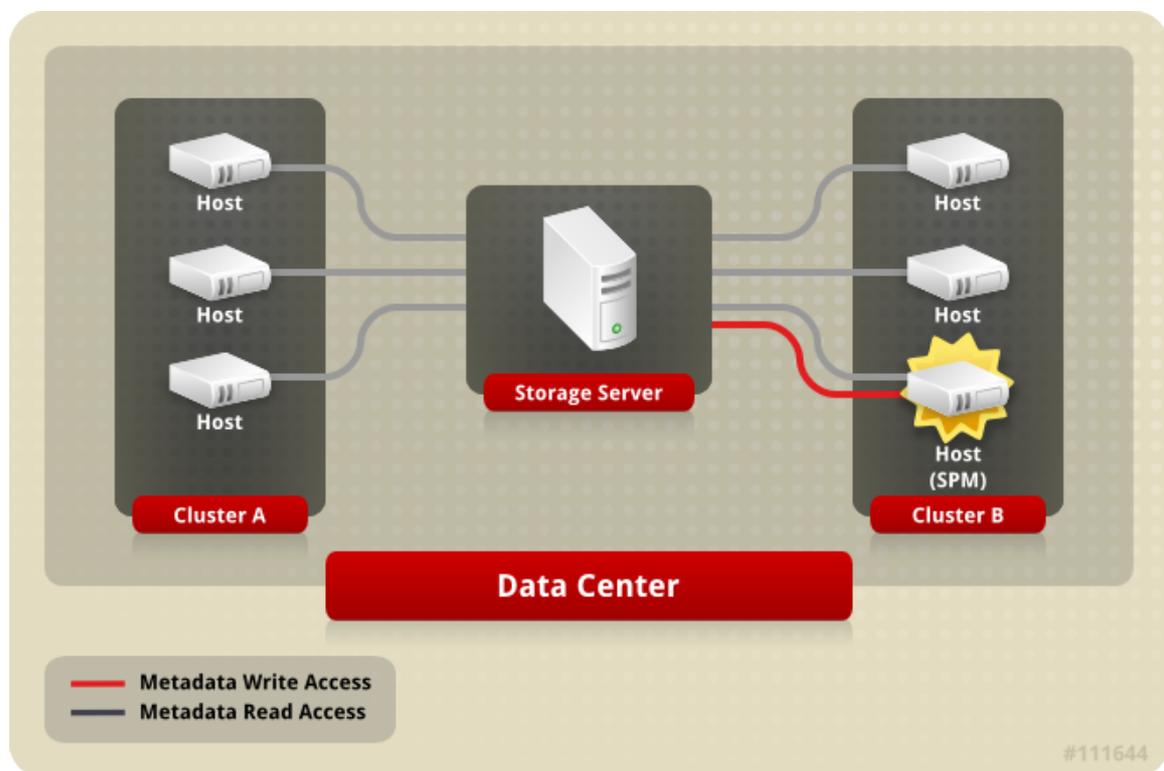
Figure 3.3. The Storage Pool Manager exclusively writes structural metadata.

Storage Pool Manager Selection Process

1. The storage pool manager selection process is initiated and managed by the Red Hat Enterprise Virtualization Manager. First, the Red Hat Enterprise Virtualization Manager requests that VDSM confirm which host has the storage-centric lease.

   • The Red Hat Enterprise 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:

     • "getSPMstatus": 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 Enterprise 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 an administrator to fence it manually. The storage pool manager role cannot be assigned to a new host until the previous storage pool manager is fenced.

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

3. If the storage pool manager role assignment fails on a new host, the Red Hat Enterprise Virtualization Manager adds the host to a list containing the hosts the operation has failed on. On subsequent iterations of the SPM selection, the Red Hat Enterprise Virtualization Manager attempts to assign the role to a host that is not included in the list.

4. The Red Hat Enterprise 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 Enterprise Virtualization Manager initiates the storage pool manager selection process.

3.3. Storage Attributes

The following attributes are given to storage objects either by the Red Hat Virtualization Manager or by a system administrator using the Red Hat Enterprise Virtualization Manager.

3.3.1. Storage Format Types

The Red Hat Enterprise Virtualization environment supports two storage formats: RAW and QCOW2.

3.3.1.1. QCOW2

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. This mapping enables advanced features like snapshots. Creating a new snapshot creates a new copy on write layer, either a new file or logical volume, with an initial mapping that points all logical blocks to the offsets in the backing file or volume. When writing to a QCOW2 volume, the relevant block is read from the backing volume, modified with the new information and written into the new snapshot QCOW2 volume. Then the map is updated to point to the new place. Benefits QCOW2 offers over using RAW representation include:
- Copy-on-write support, where a volume only represents changes made to an underlying disk image.

- Snapshot support, where a volume can represent multiple snapshots of the images history.

3.3.1.2. RAW

The RAW storage format has a performance advantage over QCOW2 in that no formatting is applied to images stored in the RAW format. Reading and writing images stored in RAW format requires no additional work on the part of the host or Manager. When the guest file system writes to a given offset

in its virtual disk, the I/O will be 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.

3.3.2. Storage Allocation Policies

3.3.2.1. Preallocated Storage

All of the storage required for a virtual machine is allocated prior to virtual machine creation. For example, if a 20 GB logical volume is created for the data partition of a virtual machine, 20 GB is allocated on disk. Enough storage must be allocated in advance to each virtual machine by an administrator to handle the forecast requirements of the virtual machine along with some added buffer. 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 Enterprise Virtualization Manager to over-commit 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 that if thin provisioning functionality provided by the storage back end is being used, preallocated storage should still be selected from the administration portal when provisioning storage for virtual machines.

3.3.2.2. Sparsely Allocated Storage

In this model, the administrator defines the total storage to be logically assigned to a virtual machine and the storage is allocated on disk on an as needed basis. When the space has been allocated, it is not discarded even if the data that uses the storage has been deleted. 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 that if thin provisioning functionality is provided by the storage back end, it should be used as the preferred implementation of thin provisioning. If this is the case, then storage should be provisioned from the graphical user interface as preallocated, leaving thin provisioning to the back end solution.

3.4. Storage Functions

Red Hat Enterprise Virtualization storage components interact to provide storage functions such as:

- *Multipathing* allows paths between all LUNs in the Red Hat Enterprise Virtualization environment to be mapped and alternate paths to be determined. This applies to block devices, although the equivalent functionality can be achieved with a sufficiently robust network setup for network attached storage.

- *Provisioning storage* allows the logical over-commitment of resources to a virtual machine and the physical assignment of the resource to the virtual machine on-demand.

- *Logical volume extension* allows images to be provided with additional storage resources when required.

- *Snapshots* provide backups of a virtual machine's system state at a certain point in time.

3.4.1. Multipathing

When the Red Hat Enterprise Virtualization Manager discovers all of the connected LUNs, multipathing is defined over the storage network. All valid paths between the LUNs in the network are mapped and consequently alternate paths are defined in case the primary path fails.

Multipathing protects against a single point of failure and provides increased bandwidth and improved security within the network. All volume groups are created on top of multipath devices, even if multiple paths are not defined for a given device.

Multipathing provides:

* Redundancy

  Multipathing provides failover protection. If any element of an I/O path (the cable, switch, or controller) fails, an alternate path is found.

* Improved Performance.

  Multipathing spreads I/O operations over the paths. By default this is done in a round-robin fashion. However, other methods are also supported, including for example *Asynchronous Logical Unit Access* (ALUA).

Figure 3.4, "Active/Passive Multipath Configuration with One RAID Device" shows an active/passive configuration with two I/O paths from the server to a RAID device. There are 2 HBAs on the server, 2 SAN switches, and 2 RAID controllers.
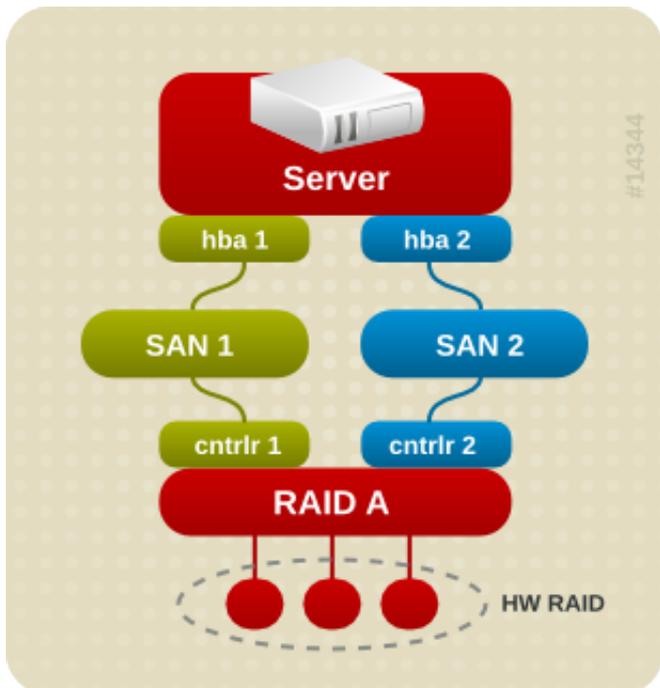
Figure 3.4. Active/Passive Multipath Configuration with One RAID Device

In this configuration, there is one I/O path that goes through hba1, SAN1, and controller 1 and a second I/O path that goes through hba2, SAN2, and controller2. There are many points of possible failure in this configuration:

* HBA failure

* FC cable failure

* SAN switch failure

* Array controller port failure

A failure at any of these points will cause a switch to an alternate I/O path.

3.4.2. Provisioning Storage

The Red Hat Enterprise Virtualization Manager provides provisioning policies to optimize storage usage within the virtualization environment. Thin provisioning policies allow administrators to over-commit resources and provision storage based on the actual storage usage of their virtualization environment.

3.4.2.1. Over-commitment

Thin provisioning is a storage policy that allows storage to be allocated to virtual machines on an on-demand basis, enabling the over-commitment of available resources. While the Red Hat Enterprise Virtualization Manager provides its own thin provisioning function it is recommended that thin provisioning functionalities provided by the storage back end for a given environment be used instead.

Over-commitment is a storage function which allows the Red Hat Enterprise Virtualization Manager to logically allocate more storage than is physically available. Generally, virtual machines use much less storage than what has been allocated to them. Over-commitment allows a virtual machine to operate completely unaware of the resources that are actually available to it at a given time.

The Red Hat Enterprise Virtualization Manager uses VDSM to define a threshold within the logically assigned storage which is used to determine that the resource usage remains within the bounds of the storage available. 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 Enterprise Virtualization Manager knows that the logical volume in question has sufficient storage to continue operations.

Because a disk image can be logically defined with more storage than is physically available to a logical volume, usage can rise to exceed the threshold limit. QEMU indicates when the storage usage exceeds the threshold, which means the logical volume assigned the storage will soon run out of physical storage. VDSM then requests that Red Hat Enterprise Virtualization Manager request the SPM to extend the logical volume. This can continue 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, the administrator must manually add storage capacity to expand the volume group. For details about logical volume extension, refer to *Section 3.4.3, "Logical Volume Extension"*.

3.4.3. Logical Volume Extension

The Red Hat Enterprise Virtualization Manager uses thin provisioning to over-commit a certain amount of storage to a disk image. Because the over-committing function logically defines more storage than is physically available, a virtual machine may attempt to use too much storage and subsequently run out of storage for its disk image. In such a situation, logical volume extension is used to provide additional storage and facilitate the continued operations for the virtual machine.

Red Hat Enterprise Virtualization provides a thin provisioning mechanism over LVM. When using QCOW2 formatted storage, Red Hat Enterprise 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 100GB disk backed by a 1GB 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 which is a dedicated logical volume in the data storage domain. A host that needs the SPM to extend a logical volume writes a message to the storage mailbox logical volume on the data storage domain in an area designated to that particular host . 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.

In a situation where the physical storage available to a volume group itself is nearly exhausted, multiple images can run out of usable storage with no means to replenish their resources. A volume group that exhausts its memory 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 the administrator must intervene to add a new LUN to the volume group.

When the administrator adds a new LUN to the volume group, the Storage Pool Manager automatically distributes the additional storage or memory resources 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.

3.4.4. 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 images as a read only 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. The layer in use before the snapshot becomes read-only and 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 also to preserve the data present *before* the snapshot is taken. Snapshots provide similar functionality to the **Undo** function present in many word processors.

Snapshots within the Red Hat Enterprise Virtualization environment cause new *Copy On Write* (COW) layers to be created for changes made to a virtual machine disk image after a snapshot is taken. The initial snapshot causes any existing volumes to be marked read only. These can be either COW or RAW. From that point on, all changes and new data are written to a new, COW layer until another snapshot is taken.

The three primary snapshot operations are:
- Snapshot creation, which involves the first snapshot created for a virtual machine.

- Snapshot 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.

- Snapshot deletion, which involves deleting a restoration point that is no longer required.

For task based information about snapshot operations, refer to the *Red Hat Enterprise Virtualization Administration Guide*.

Snapshot Creation

In Red Hat Enterprise 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 read-only 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 Enterprise 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 3.5, "Initial Snapshot Creation"*, the creation of a snapshot causes the volumes that comprise a virtual machine disk image to be marked as read only and serve as the base image for all subsequent snapshots.
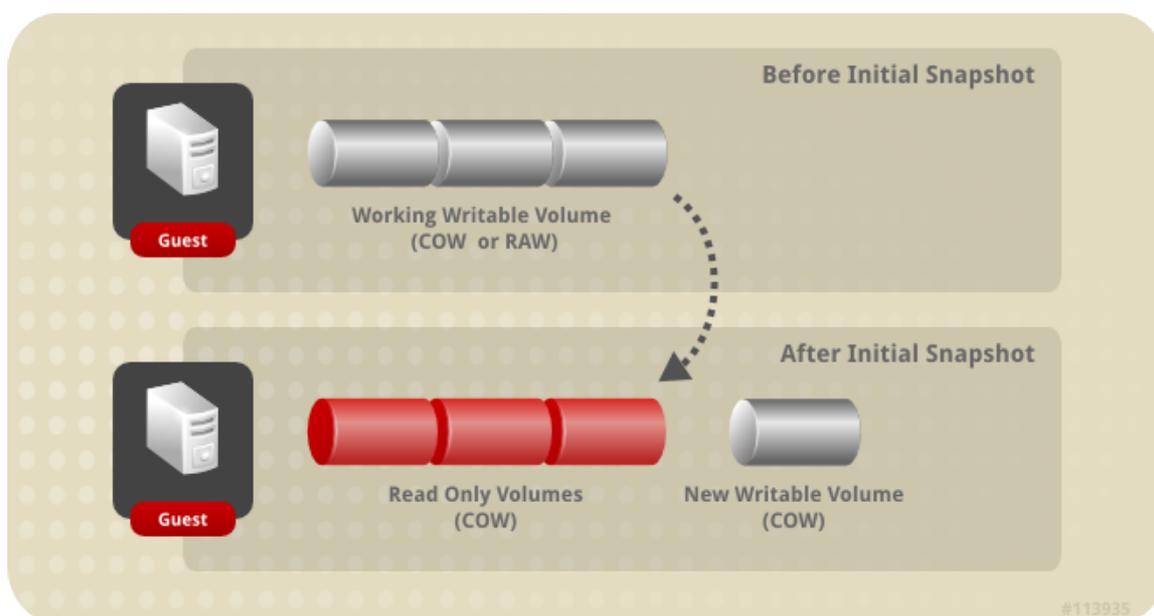
Figure 3.5. Initial Snapshot Creation

Each new snapshot causes the COW layer currently in use for virtual machine write operations to be marked read only.

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, read only COW layer, the data is read from the read only layer, and written into the newest, writable COW layer. Virtual machines locate data by checking each COW layer from most recent to oldest, transparently to the virtual machine.
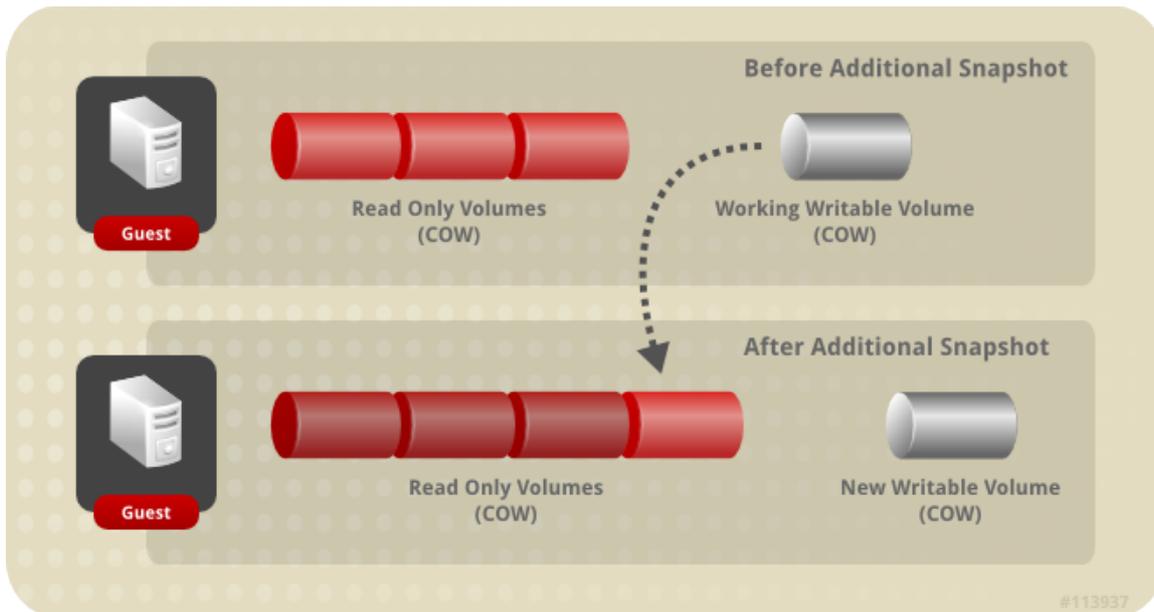
Figure 3.6. Additional Snapshot Creation

> **Note**
>
> Red Hat Enterprise Virtualization requires a virtual machine to be shut down before a snapshot of it is created.

Snapshot Previews

To select which snapshot a virtual machine 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 3.7, "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.
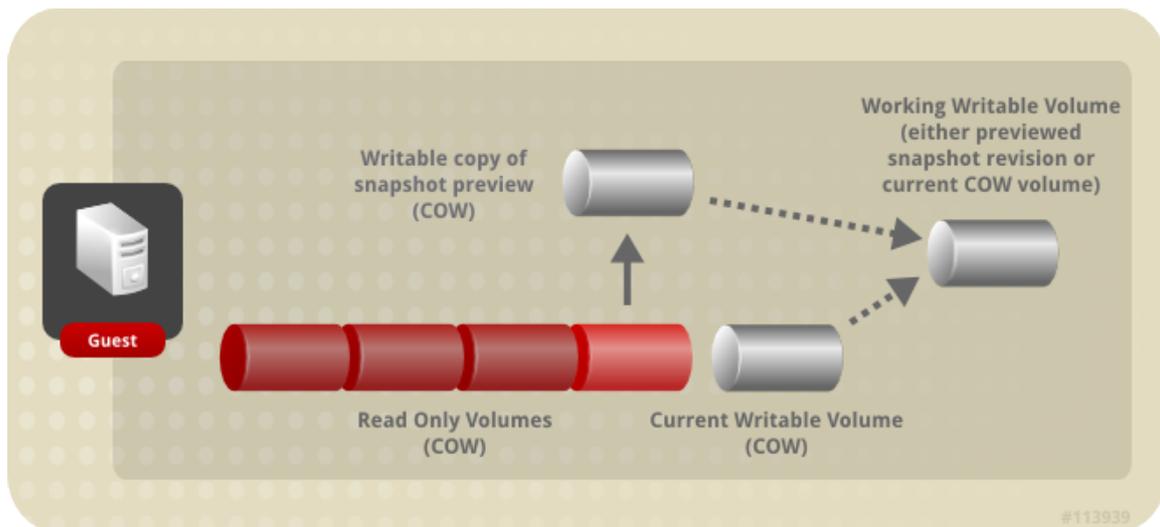
Figure 3.7. Preview Snapshot

Snapshot Deletion

If a snapshot or series of snapshots are no longer required, the administrator can delete one or more snapshots. The deletion of a snapshot does not necessarily cause the data in the snapshot to be deleted. For example, if the third snapshot out of five snapshots is deleted, the unchanged data in the third snapshot must be preserved for the fourth and fifth snapshots to be usable. If the fifth snapshot out of five is deleted, then the data that has been modified or created since the snapshot was taken will be discarded. Snapshot deletion is not an operation to preserve storage capacity within the Red Hat Enterprise Virtualization environment. Snapshot deletion allows an administrator to remove a potential data restoration point when it becomes clear that it will not be necessary to return a virtual machine hard disk image data to the point in time the snapshot preserves.

When the administrator deletes a snapshot, the data from the deleted snapshot and the snapshot created after the deleted snapshot are merged into a single COW volume. After the two snapshots are merged, the resultant volume contains any data that was created or modified prior to the deleted snapshot and after the deleted snapshot. No data has been removed, only the ability to restore a point in time in the life of the virtual machine hard disk image. As displayed in *Figure 3.8, "Snapshot Deletion"*, snapshot 2 is selected for deletion. As a consequence, snapshot 2 and snapshot 3 are merged, saving the changes in both snapshots in the COW volume for snapshot 3 (i.e. the newer snapshot) as the replacement for the deleted snapshot.
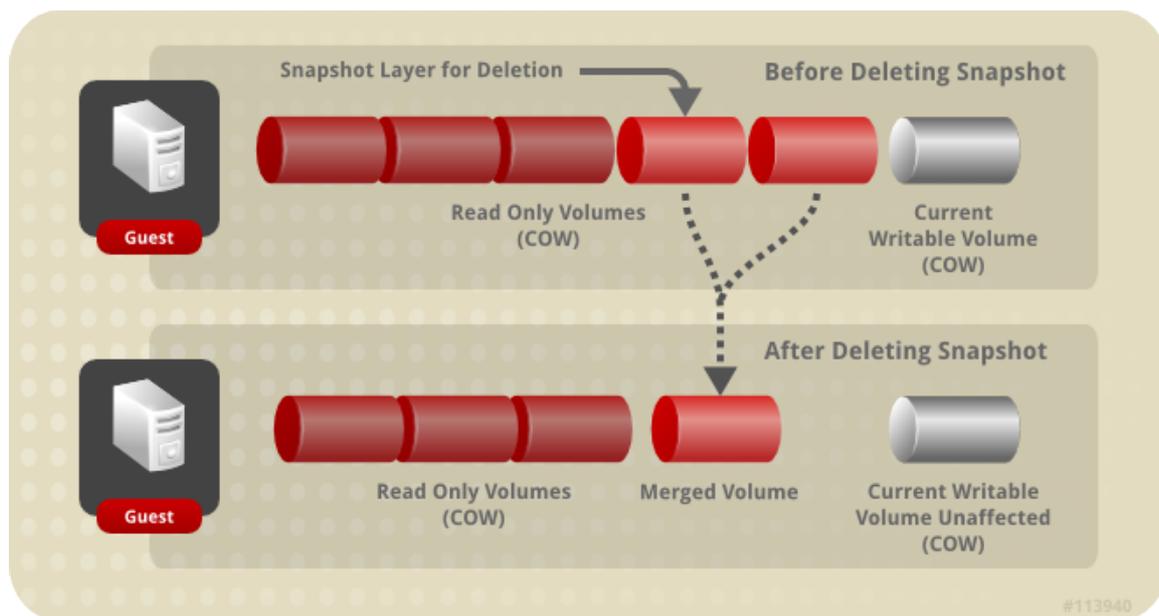
Figure 3.8. Snapshot Deletion

Network Architecture

A well designed and built network ensures, for example, that high bandwidth tasks receive adequate bandwidth, that user interactions are not crippled by frustrating 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.

Red Hat Enterprise Virtualization networking is discussed in this chapter 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.

4.1. Introduction: Basic Networking Terms

Red Hat Enterprise 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.

4.1.1. Network Interface Controller (NIC)

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 Enterprise 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 Enterprise Virtualization Manager assigns each VNIC a unique MAC address.

4.1.2. 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 Enterprise 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 Red Hat Enterprise 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.

4.1.3. Bond

A *Bond* aggregates multiple NICs in a parallel manner to provide combined speed that is beyond single NIC speeds. Bonding provides increased fault tolerance by increasing the number of failures required for networking to fail completely. The NICs that form a bond device must be of the same make and model in order to ensure that both devices support the same options and modes.

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

Bonding Modes

Red Hat Enterprise Virtualization uses mode 4 by default but supports the following common bonding modes:

- Mode 1 (active-backup policy) sets all interfaces to the backup state while one remains active. Upon failure on the active interface, a backup interface replaces it as the only active interface in the bond. The MAC address of the bond in mode 1 is visible on only one port (the network adapter), to prevent confusion for the switch. Mode 1 provides fault tolerance and is supported in Red Hat Enterprise Virtualization.

- Mode 2 (XOR policy) selects an interface to transmit packages to based on the result of an XOR operation on the source and destination MAC addresses multiplied by the modulo slave count. This calculation ensures that the same interface is selected for each destination MAC address used. Mode 2 provides fault tolerance and load balancing and is supported in Red Hat Enterprise Virtualization.

- Mode 3 (broadcast policy) transmits all packages to all interfaces. Mode 3 provides fault tolerance and is supported in Red Hat Enterprise Virtualization.

- Mode 4 (IEEE 802.3ad policy) creates aggregation groups for which included interfaces share the speed and duplex settings. Mode 4 uses all interfaces in the active aggregation group in accordance with the IEEE 802.3ad specification and is supported in Red Hat Enterprise Virtualization.

- Mode 5 (adaptive transmit load balancing policy) ensures the outgoing traffic distribution is according to the load on each interface and that the current interface receives all incoming traffic. If the interface assigned to receive traffic fails, another interface is assigned the receiving role instead. Mode 5 is supported in Red Hat Enterprise Virtualization.

4.1.4. Virtual Network Interface Controller(VNIC)

An NIC is the physical network interface controller for the host. A VNIC is a virtual NIC based on the physical NIC. Each host can have one or more NICs and each NIC can be a base for multiple VNICs. Every virtual machine with a network interface results in a new VNIC with a unique MAC address on the host where the virtual machine runs. These VNICs are then added to the network bridge which implements the logical network that a virtual machine is connected to. For details about NICs, refer to *Section 4.1.1, "Network Interface Controller (NIC)"*. For details about bridges, refer to *Section 4.1.2, "Bridge"*.

Running the **ifconfig** command on a Red Hat Enterprise Virtualization shows all of the VNICs that are associated with virtual machines on a host. Also visible are any network bridges that have been created to implement logical networks, and any NICs used by the host.

```
[root@ecs-cloud-rhevh-01 ~]# ifconfig
eth0      Link encap:Ethernet  HWaddr E4:1F:13:B7:FD:D4
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:2527437 errors:0 dropped:0 overruns:0 frame:0
          TX packets:7353099 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:1842636390 (1.7 GiB)  TX bytes:4527273914 (4.2 GiB)
          Interrupt:169 Memory:92000000-92012800

bond0     Link encap:Ethernet  HWaddr 00:1B:21:98:25:E4
          UP BROADCAST RUNNING MASTER MULTICAST  MTU:1500  Metric:1
          RX packets:1207008987 errors:0 dropped:2132 overruns:0 frame:0
          TX packets:1172475485 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:1564609462833 (1.4 TiB)  TX bytes:885715805671 (824.8 GiB)


rhevm     Link encap:Ethernet  HWaddr E4:1F:13:B7:FD:D4
          inet addr:10.64.14.122  Bcast:10.64.15.255  Mask:255.255.254.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:445040 errors:0 dropped:0 overruns:0 frame:0
          TX packets:4721866 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:41575335 (39.6 MiB)  TX bytes:4171361904 (3.8 GiB)

storage   Link encap:Ethernet  HWaddr 00:1B:21:98:25:E4
          inet addr:192.168.29.10  Bcast:192.168.29.255  Mask:255.255.255.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:86956273 errors:0 dropped:0 overruns:0 frame:0
          TX packets:62074574 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:106661218057 (99.3 GiB)  TX bytes:83616530712 (77.8 GiB)

vnet000 Link encap:Ethernet  HWaddr FE:1A:4A:40:0E:04
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:477233 errors:0 dropped:0 overruns:0 frame:0
          TX packets:630027 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:500
          RX bytes:123257049 (117.5 MiB)  TX bytes:387924090 (369.9 MiB)

vnet001 Link encap:Ethernet  HWaddr FE:1A:4A:40:0E:30
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:1642 errors:0 dropped:0 overruns:0 frame:0
          TX packets:120753 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:500
          RX bytes:318222 (310.7 KiB)  TX bytes:14323345 (13.6 MiB)

vnet002 Link encap:Ethernet  HWaddr FE:1A:4A:40:0E:2E
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:239673 errors:0 dropped:0 overruns:0 frame:0
          TX packets:555398 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:500
          RX bytes:17514233 (16.7 MiB)  TX bytes:620136453 (591.4 MiB)
```

The given console output shows one bond device, bond0; one ethernet NIC eth0; two network bridges: storage and rhevm; and a number of VNICs that are associated virtual machine network interfaces using virtio drivers. For more information on virtualized hardware refer to *Appendix C, Virtualized Hardware*.

The VNICs displayed in the given console output are members of the network bridge for a logical network. Bridge membership can be displayed using the `brctl show` command:

```
[root@ecs-cloud-rhevh-01 ~]# brctl show
bridge name bridge id  STP enabled interfaces
rhevm  8000.e41f13b7fdd4 no  vnet002
       vnet001
       vnet000
       eth0
storage  8000.001b219825e4 no  bond0
```

The given console output shows that the virtio VNICs are members of the `rhevm` bridge, because all of the virtual machines that the VNICs are associated with are connected to the **rhevm** network. The `eth0` NIC is also a member of the `rhevm` bridge. The `eth0` device is cabled to a switch that provides connectivity beyond the host.

Figure 4.1, "Networking within a cluster". depicts that each host in this setup has three NICs, except Host C. VNICs cannot exist without either a physical NIC or bridge attached to the host, but virtual machines can remain unconnected to any network or VNIC/NIC. A virtual machine connects directly to a VNIC, which uses the bridge and physical NIC to form a network link with objects connected to a given logical network.

4.1.5. Virtual LAN (VLAN)

A *VLAN (Virtual LAN)* is an attribute that can be applied to network packets. Network packets can be "tagged" into a particular numbered VLAN. A VLAN is a security feature used to completely isolate network traffic at the switch level as VLANs are completely separate and mutually exclusive. The Red Hat Enterprise Virtualization 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.

4.2. Networking in Data Centers and Clusters.

Networking within a cluster and a data center refers to objects involved in networking at those levels. *Figure 4.1, "Networking within a cluster"*. displays data center and cluster networking objects and their connections in Red Hat Enterprise Virtualization.
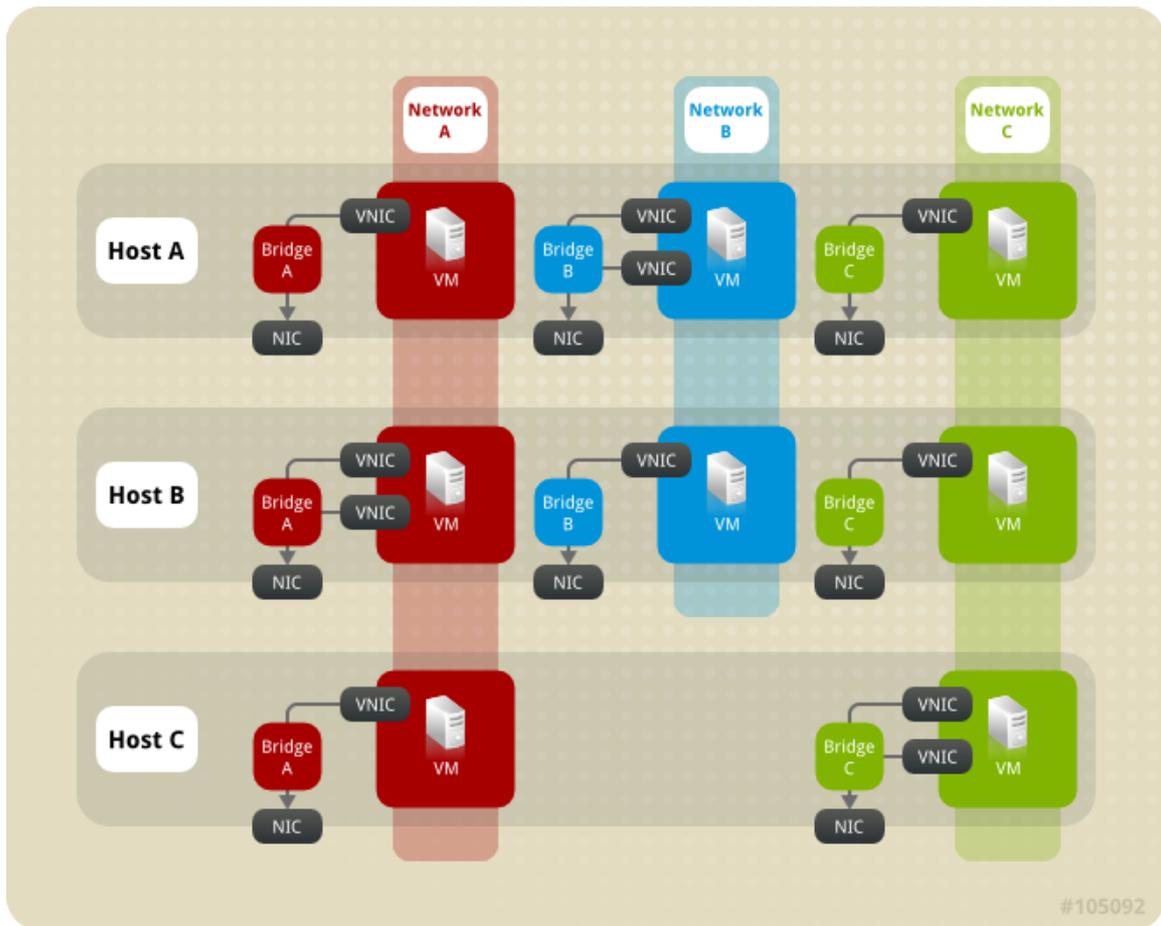
Figure 4.1. Networking within a cluster

Cluster level networking objects include:

- Clusters

- Logical Networks

The listed objects have unique networking properties in Red Hat Enterprise Virtualization and are discussed in detail individually.

4.2.1. Cluster Networking

A data center is a logical grouping of multiple clusters and each cluster is a logical group of multiple hosts. *Figure 4.1, "Networking within a cluster"*. depicts the contents of a single cluster.

Hosts in a cluster all have access to the same data center storage domains. Hosts in a cluster also have logical networks applied at the cluster level. For a 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 Enterprise Virtualization Manager.

4.2.2. Logical Networks

Logical networking allows the Red Hat Enterprise Virtualization environment to separate network traffic by type. For example, the `rhevm` network is created by default during the installation of the Red Hat Enterprise 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.

Logical networks are defined at the data center level, and added to a host. For a logical network to be operational, it must be implemented for every host in a given cluster. Each logical network in a Red Hat Enterprise Virtualization environment is backed by a network bridge device on a host. So when a new 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 Enterprise Virtualization Manager automatically creates required bridges when a host has been put into maintenance mode.

The network bridge device that is created by the Red Hat Enterprise Virtualization Manager to back a logical network device is associated with a physical network device. If the physical NIC that a bridge includes 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.
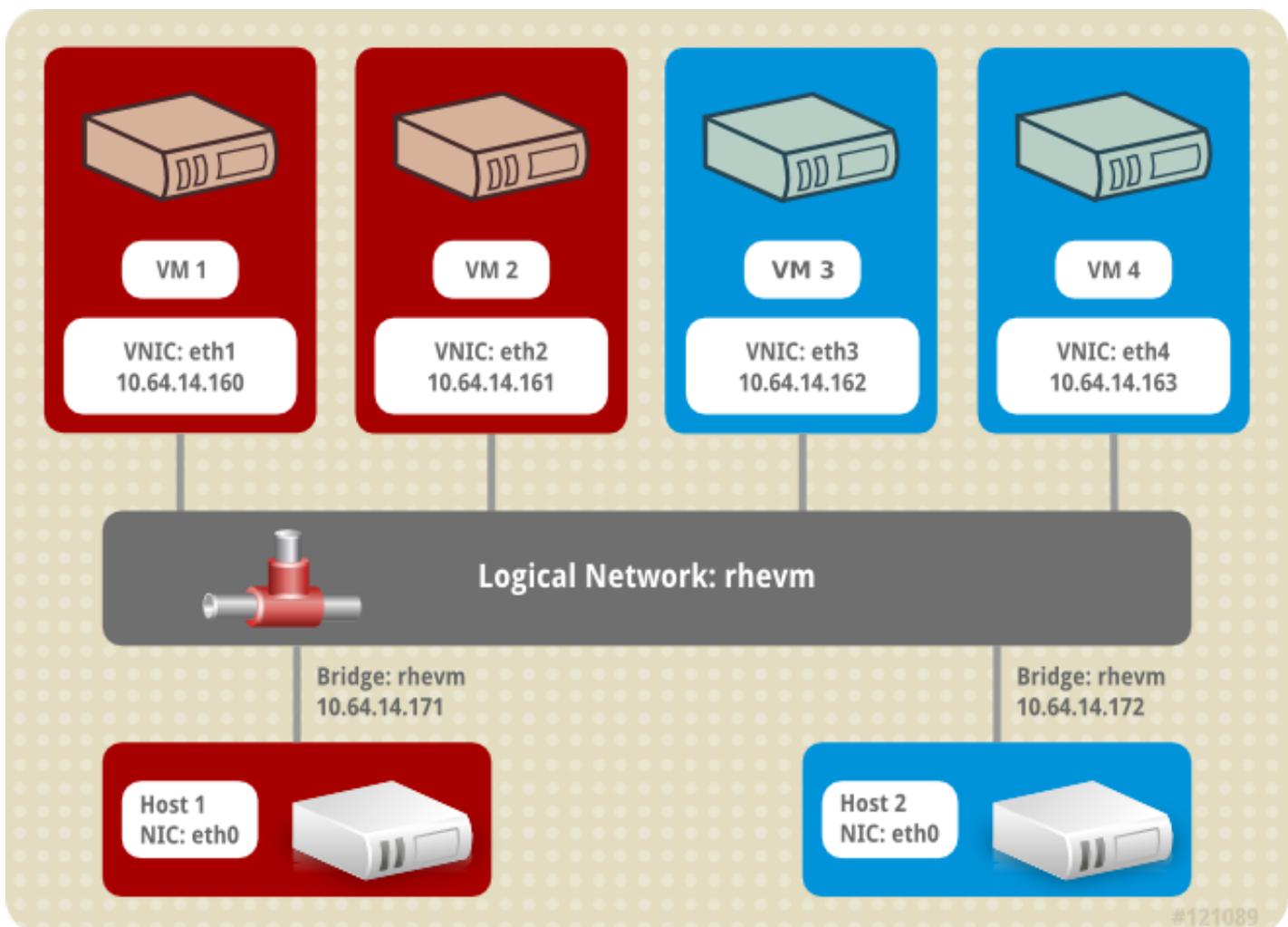
Figure 4.2. The rhevm logical network.

Example 4.1. 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, rhevm 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.

4.3. Networking in Hosts and Virtual Machines

Host networking refers to the configuration options possible for networking connectivity at the host level. Virtual machine networking refers to virtual machine network interactions.

4.3.1. Host Networking Configurations

Common types of networking configurations for Red Hat Enterprise Virtualization hosts include:

- Bridge and NIC configuration.

- Bridge, VLAN, and NIC configuration.

- Bridge, Bond, and VLAN configuration.

- Multiple Bridge, Multiple VLAN, and NIC configuration.

4.3.1.1. Bridge Configuration

The simplest host configuration in Red Hat Enterprise Virtualization is the Bridge and NIC configuration. As *Figure 4.3, "Bridge and NIC configuration"* depicts, this configuration uses a bridge to connect one or more virtual machines (or guests) to the host's NIC.
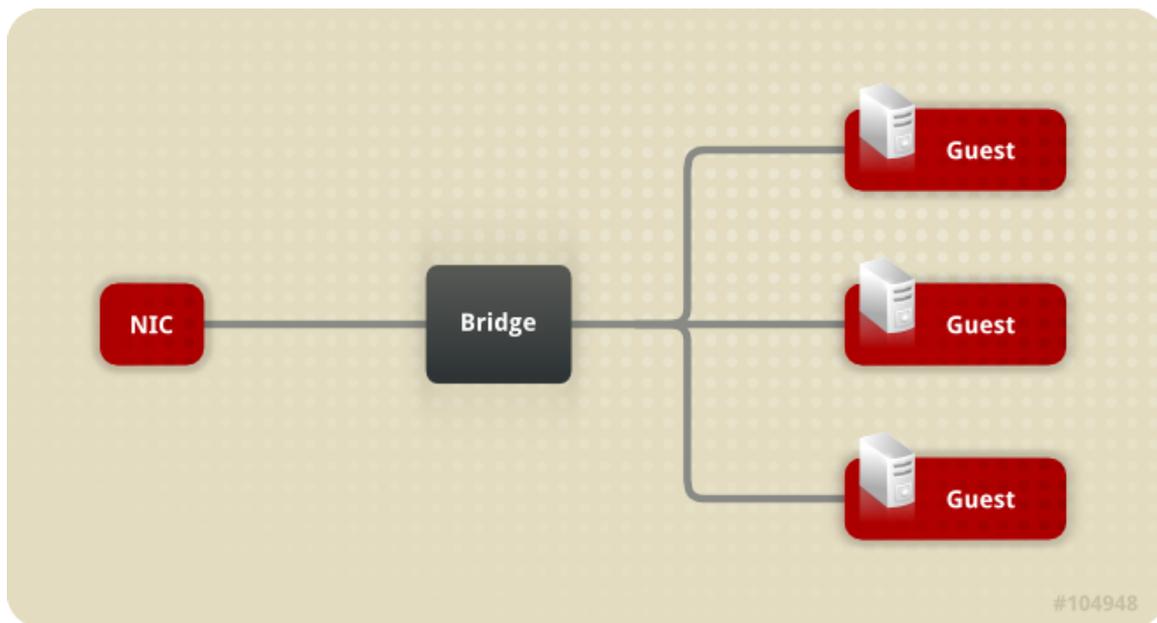
Figure 4.3. Bridge and NIC configuration

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

4.3.1.2. VLAN Configuration

Figure 4.4, "Bridge, VLAN, and NIC configuration" depicts an alternative configuration that includes a virtual LAN (VLAN) to connect the host NIC and bridge.
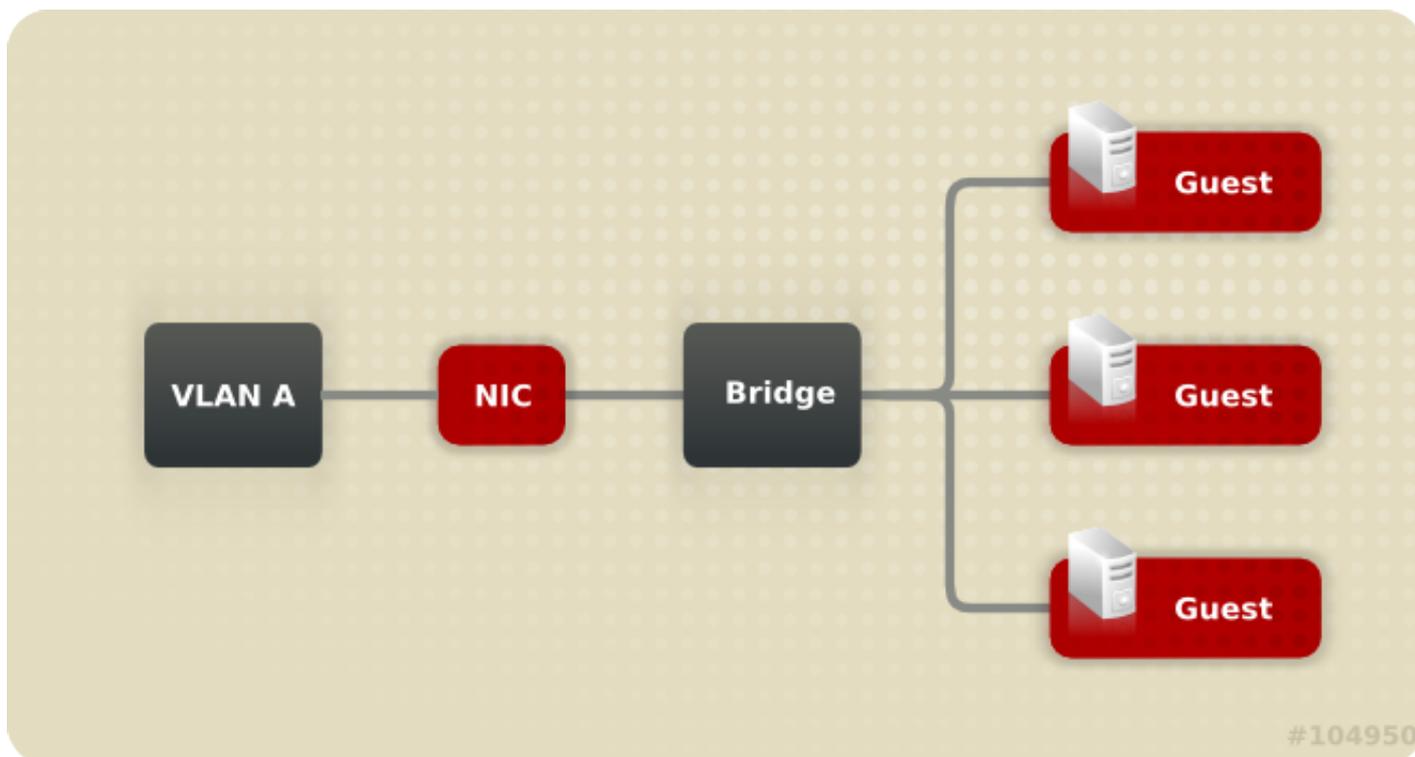
Figure 4.4. Bridge, VLAN, and NIC configuration

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. For more information on VLANs, refer to *Section 4.1.2, "Bridge"*

4.3.1.3. Bridge and Bond Configuration

Figure 4.5, "Bridge, Bond, and NIC configuration" displays a configuration that includes a bond to connect multiple host NICs to the same bridge and network.
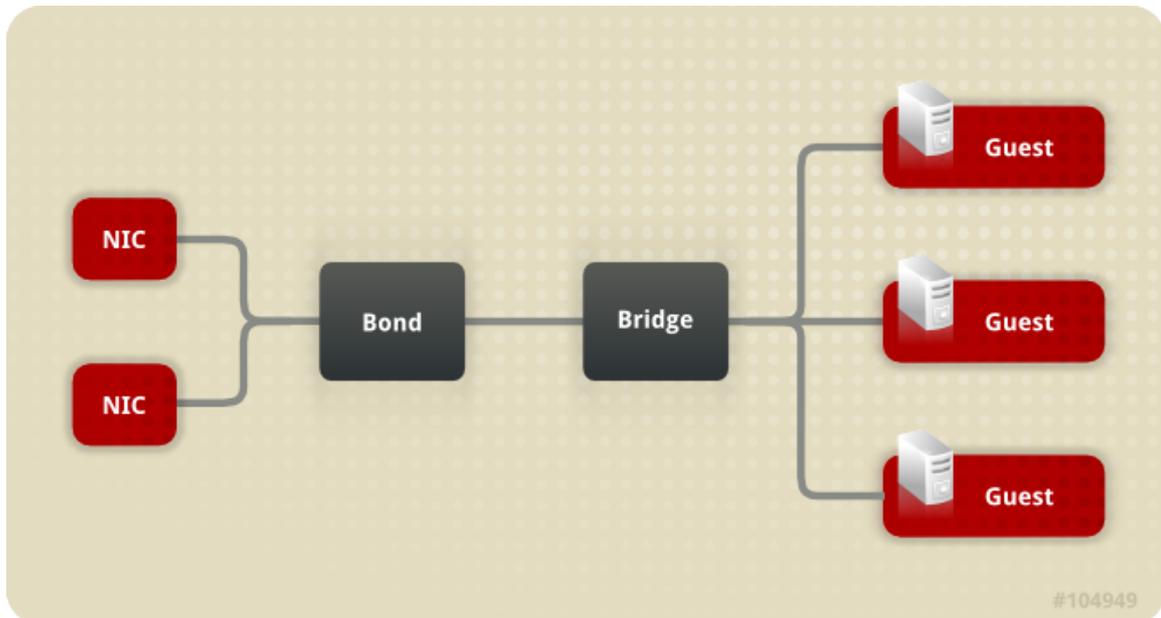
Figure 4.5. Bridge, Bond, and NIC configuration

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.

4.3.1.4. Multiple Bridge, Multiple VLAN, and NIC configuration

Figure 4.6, "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.
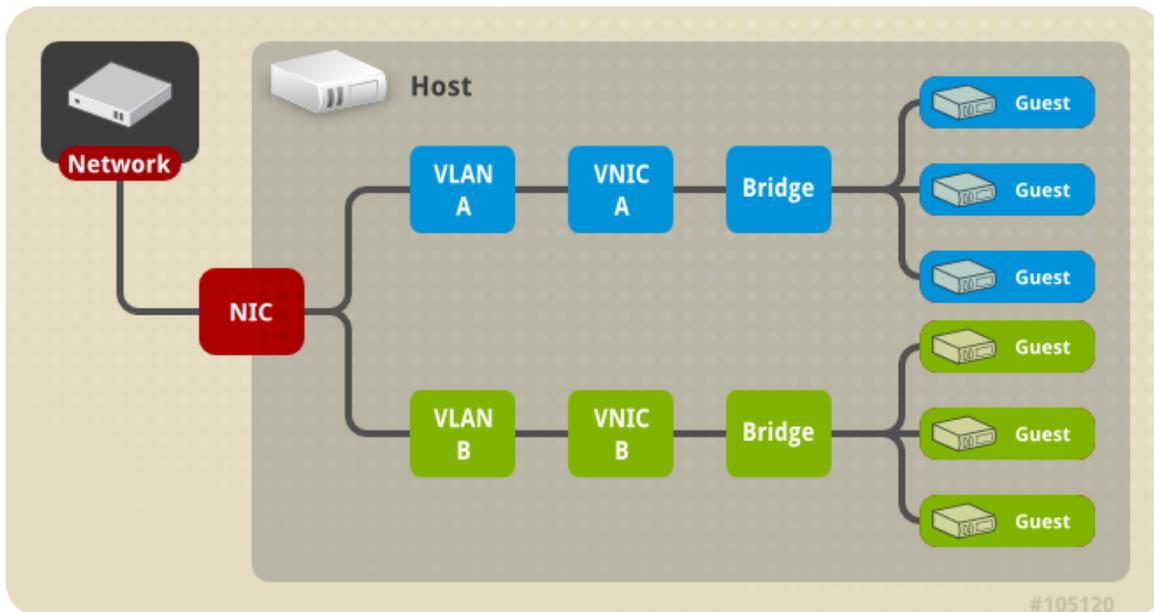
Figure 4.6. Multiple Bridge, Multiple VLAN, and NIC configuration

4.3.1.5. Multiple Bridge, Multiple VLAN, and Bond Configuration

Figure 4.7, "Multiple Bridge, Multiple VLAN, and Multiple NIC with Bond connection". displays a configuration that bonds multiple NICs to facilitate a connection with multiple VLANs.
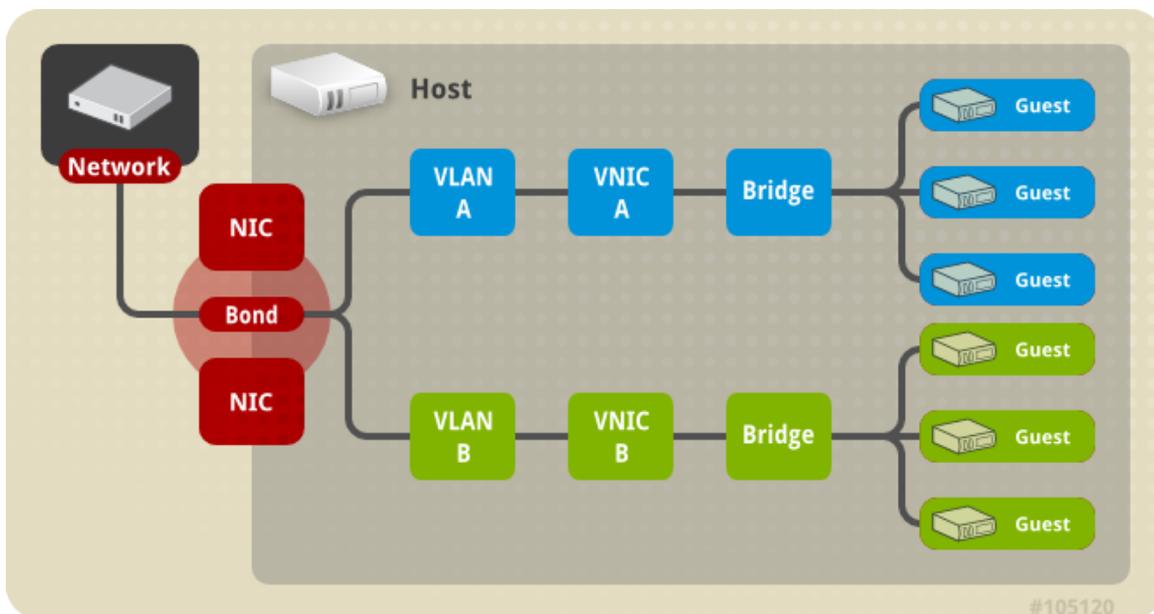
Figure 4.7. Multiple Bridge, Multiple VLAN, and Multiple NIC with Bond connection

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.

4.3.2. Virtual Machine Connectivity

In Red Hat Enterprise Virtualization, a virtual machine has its NIC put on a logical network at the time that it is created by an administrator. 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 `rhevm` logical network, its VNIC is added as a member of the `rhevm` bridge of the host on which that virtual machine runs.

Power Management and Fencing

The Red Hat Enterprise Virtualization environment is most flexible and resilient when power management and fencing have been configured. Power management allows the Red Hat Enterprise 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 Enterprise Virtualization environment 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 Enterprise Virtualization Manager connects to a power management devices using a network IP address or hostname. In the context of Red Hat Enterprise Virtualization, a power management device and a fencing device are the same thing.

5.1. Power Management

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

*   *Advanced Lights Out Manager* (alom).

*   *American Power Conversion* (apc).

*   Bladecenter.

*   *Dell Remote Access Card 5* (drac5).

*   *Electronic Power Switch* (eps).

*   *Integrated Lights Out* (ilo, ilo2, ilo3).

*   *Intelligent Platform Management Interface* (ipmilan).

*   *Remote Supervisor Adapter* (rsa).

*   rsb.

*   *Western Telematic, Inc* (wti).

*   *Cisco Unified Computing System* (cisco_ucs).

In order to communicate with the listed power management devices, the Red Hat Enterprise Virtualization Manager makes use of *fence agents*. The Red Hat Enterprise 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.

Resiliency is accomplished through properly configured power management for a Red Hat Enterprise Virtualization environment. Fencing agents allow the Red Hat Enterprise Virtualization manger to communicate with host power management devices in order to bypass the operating system on a problem host, and isolate the host from the rest its environment with a power cycle. The Manager can then reassign the SPM role(see *Section 3.2, "Role: The Storage Pool Manager"*), if it was held by the problem host, and safely restart any highly available virtual machines on other hosts.

5.2. Fencing

In the context of the Red Hat Enterprise 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 is essential in insuring that the role of SPM is always assigned to a functional host. If the problem host was the SPM, the SPM role is relinquished and reassigned to an 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 Enterprise Virtualization Manager to safely release the lock on a virtual machine hard disk image because the Manager can use a fence agent to confirm that a problem host has truly been rebooted. When this confirmation is received, the Red Hat Enterprise Virtualization Manager can safely 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 Enterprise Virtualization Manager allows a grace period of thirty (30) seconds to pass before any action is taken in order 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 first stop the host, confirm it has stopped, start the host and confirm that it 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 a host to become non-responsive has been resolved by a reboot, then it will automatically be set to **Up** status and be capable of starting and hosting virtual machines.

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 Enterprise 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.

6.1. 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 Enterprise 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. There are three load balancing policies: **None**, **Even Distribution**, and **Power Saving**.

6.1.1. 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.

6.1.2. Load Balancing Policy: Even Distribution

An even distribution load balancing policy selects the host for a new virtual machine according to lowest CPU utilization. The maximum service level is the maximum CPU utilization that is allowed for hosts in a cluster, beyond which environment performance will degrade. The even distribution policy allows an administrator to set a maximum service level for running virtual machines. The length of time a host is allowed to continue at this maximum service level before the Red Hat Enterprise Virtualization Manager intervenes is also set by an administrator. If a host has reached the maximum service level and stays there for more then the set time, virtual machines on that host are migrated one by one to the host in the cluster that has the lowest CPU utilization. Host resources are checked once per minute, and one virtual machine is migrated at a time until the host CPU utilization is below the maximum service threshold.

6.1.3. Load Balancing Policy: Power Saving

A power saving load balancing policy selects the host for a new virtual machine according to lowest CPU utilization. The maximum service level is the maximum CPU utilization that is allowed for hosts in a cluster, beyond which environment performance will degrade. The minimum service level is the minimum CPU utilization allowed before the continued operation of a host is considered an inefficient use of electricity. The even distribution policy allows an administrator to set a maximum and minimum service level for running virtual machines. The length of time a host is allowed to continue at this maximum or minimum service level before the Red Hat Enterprise Virtualization Manager intervenes is also set by an administrator. If a host has reaches the maximum service level and stays there for more the the set time, the virtual machines on that host are migrated one by one to the host that has the lowest CPU utilization. The process continues until the host CPU utilization is below maximum service level. If a host CPU utilization falls below the minimum service level the virtual machines are migrated to other hosts in the cluster if their maximum service level permits. When an under-utilized host is cleared of its remaining virtual machines, it can be shut down by an administrator to preserve power.

6.2. Scheduling

In Red Hat Enterprise Virtualization, scheduling refers to the way the Red Hat Enterprise 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. If multiple hosts are eligible targets, one will be selected based on the load balancing policy for the cluster. For example, if an even distribution 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 status (SPM, see *Section 3.2, "Role: The Storage Pool Manager"* for more information) 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.

6.3. Migration

The Red Hat Enterprise 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 Enterprise 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 Enterprise Virtualization Manager to determine virtual machine CPU usage.

Migration Statistics

A bandwidth limit of 30 Mbps is imposed on each virtual machine migration. A migration will time out after a certain amount of time has passed. The time out happens out after either 300 seconds, or 300 seconds multiplied by a factor of the virtual machine memory divided by 2048, which ever is larger.

By default, concurrent outgoing migrations are limited to one per CPU core per host, or 5, which ever is smaller.

Directory Services

The Red Hat Enterprise Virtualization platform relies on directory services for user authentication and authorization. Interactions with all Manager interfaces, including the User Portal, Power User Portal, Administration Portal, and REST API are limited to authenticated, authorized users. Virtual machines within the Red Hat Enterprise Virtualization can use the same directory services to provide authentication and authorization, however they must be configured to do so. Currently the two supported providers of directory services for use with the Red Hat Enterprise Virtualization Manager are *Identity, Policy, and Audit* (IPA) and Microsoft *Active Directory* (AD). The Red Hat Enterprise 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 Enterprise 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 Enterprise 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 Enterprise Virtualization environment. After the installation is complete, more domains can be added.

7.1. Local Authentication: Internal Domain

The Red Hat Enterprise Virtualization Manager creates a limited, internal administration domain during installation. This domain is not the same as an AD or IPA domain, because it exists based on a key in the Red Hat Enterprise Virtualization postgres 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 Enterprise 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 IPA authentication domains, and delegating permissions to users from external domains.

7.2. Remote Authentication Using GSSAPI

In the context of Red Hat Enterprise Virtualization, remote authentication refers to authentication that is handled remotely from the Red Hat Enterprise Virtualization Manager. Remote authentication is used for user or API connections coming to the Manager from within an AD or IPA domain. The Red Hat Enterprise Virtualization Manager must be configured by an administrator using the `rhevm-`

`manage-domains` tool to be a part of an AD or IPA domain. This requires that the Manager be provided with credentials for an account from the AD or IPA 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 Enterprise 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.
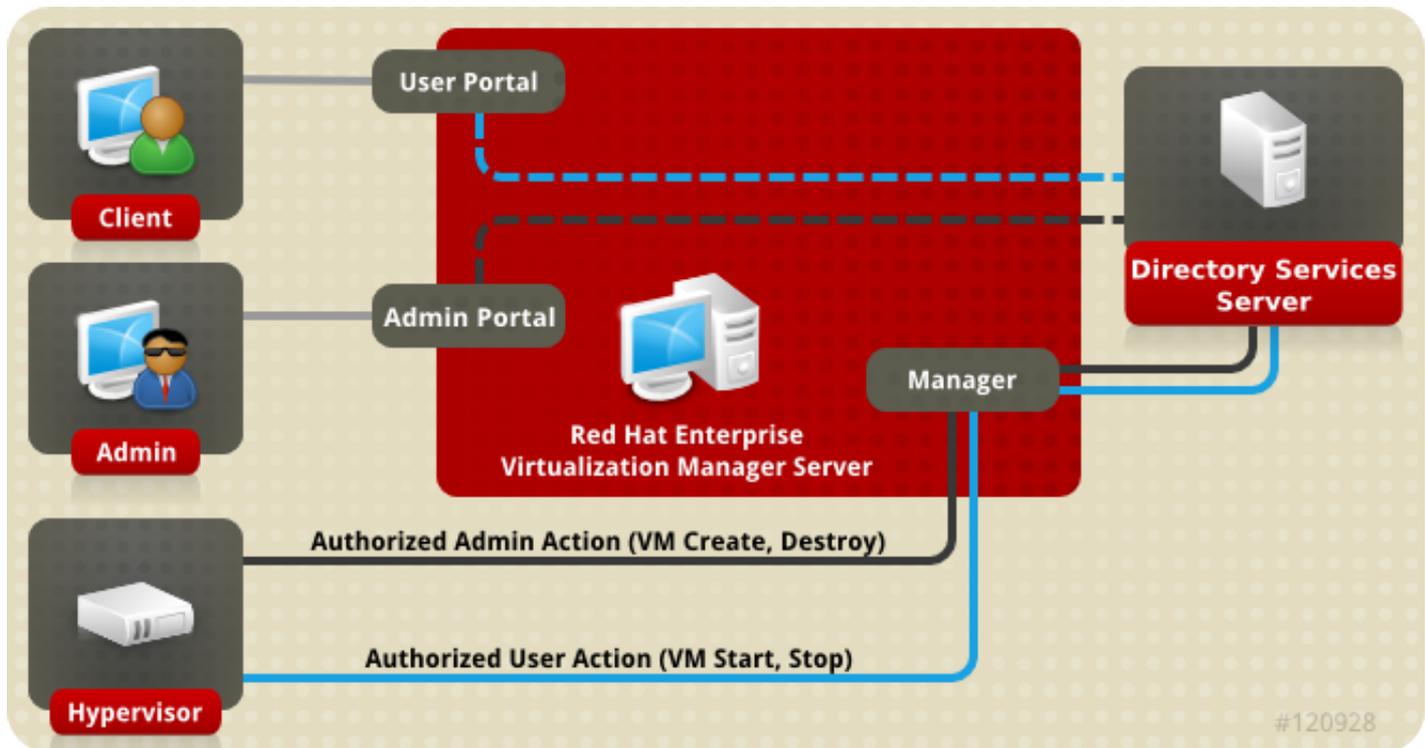
Figure 7.1. GSSAPI Authentication

Templates and Pools

The Red Hat Enterprise 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.

8.1. 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 Enterprise Virtualization environment is discussed it the *Red Hat Enterprise Virtualization Administration 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 machine 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, heavy use of the read only backing image can cause the data being accessed to be cached, resulting in a net performance increase.

8.2. 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 8.1. 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.

Reporting database views

Red Hat Enterprise Virtualization Manager collects historical data and statistics for use in constructing reports. This chapter details the reporting views available in the system, and their columns and data types. Creating reports is explained in the *Red Hat Enterprise Virtualization Administration Guide*.

9.1. Statistics History Views

This section describes the statistics history views available to the user for querying and generating reports.

9.1.1. v3_0_datacenter_samples_history_view \v3_0_datacenter_hourly_history_view \v3_0_datacenter_daily_history_view

Historical statistics for each data center in the system.

Table 9.1. v3_0_datacenter_samples_history_view\v3_0_datacenter_hourly_history_view \v3_0_datacenter_daily_history_view

Name	Type	Description
history_id	integer	The unique ID of this row in the table.
history_datetime	timestamp with time zone	The timestamp of this history row (rounded to minute, hour, day as per the aggregation level).
datacenter_id	uuid	The unique ID of the data center.
datacenter_status	smallint	• -1 - Unknown Status (used only to indicate a problem with the ETL -- PLEASE NOTIFY SUPPORT) • 1 - Up • 2 - Maintenance • 3 - Problematic
minutes_in_status	decimal	The total number of minutes that the data center was in the status shown in the datacenter_status column for the aggregation period. For example, if a data center was up for 55 minutes and in maintenance mode for 5 minutes during an hour, two rows will show for this hour. One will have a datacenter_status of Up and minutes_in_status

Name	Type	Description
		of 55, the other will have a datacenter_status of Maintenance and a minutes_in_status of 5.
datacenter_configuration_version	integer	The data center configuration version at the time of sample.

9.1.2. v3_0_storage_domain_samples_history_view-v3_0_storage_domain_hourly_history_view \v3_0_storage_domain_daily_history_view

Historical statistics for each storage domain in the system.

Table 9.2. v3_0_storage_domain_samples_history_view\v3_0_storage_domain_hourly_history_view-v3_0_storage_domain_daily_history_view

Name	Type	Description
history_id	integer	The unique ID of this row in the table.
history_datetime	timestamp with time zone	The timestamp of this history row (rounded to minute, hour, day as per the aggregation level).
storage_domain_id	uuid	Unique ID of the storage domain in the system.
available_disk_size_gb	integer	The total available (unused) capacity on the disk, expressed in gigabytes (GB).
used_disk_size_gb	integer	The total used capacity on the disk, expressed in gigabytes (GB).
storage_configuration_version	integer	The storage domain configuration version at the time of sample.

9.1.3. v3_0_host_samples_history_view \v3_0_host_hourly_history_view\v3_0_host_daily_history_view

Historical statistics for each host in the system.

Table 9.3. v3_0_host_samples_history_view\v3_0_host_hourly_history_view \v3_0_host_daily_history_view

Name	Type	Description
history_id	integer	The unique ID of this row in the table.
history_datetime	timestamp with time zone	The timestamp of this history row (rounded to minute, hour, day as per the aggregation level).

Name	Type	Description
host_id	uuid	Unique ID of the host in the system.
host_status	smallint	• -1 - Unknown Status (used only to indicate a problem with the ETL -- PLEASE NOTIFY SUPPORT) • 1 - Up • 2 - Maintenance • 3 - Problematic
minutes_in_status	decimal	The total number of minutes that the host was in the status shown in the status column for the aggregation period. For example, if a host was up for 55 minutes and down for 5 minutes during an hour, two rows will show for this hour. One will have a status of Up and minutes_in_status of 55, the other will have a status of down and a minutes_in_status of 5.
memory_usage_percent	smallint	Percentage of used memory on the host.
max_memory_usage	smallint	Percentage of used memory on the host.
cpu_usage_percent	smallint	Used CPU percentage on the host.
max_cpu_usage	smallint	The maximum CPU usage for the aggregation period, expressed as a percentage. For hourly aggregations, this is the maximum collected sample value. For daily aggregations, it is the maximum hourly average value.
ksm_cpu_percent	smallint	CPU percentage ksm on the host is using.
max_ksm_cpu_percent	smallint	The maximum KSM usage for the aggregation period, expressed as a percentage. For hourly aggregations, this is the maximum collected sample value. For daily aggregations, it is the maximum hourly average value.

Name	Type	Description
active_vms	smallint	The average number of active virtual machines for this aggregation.
max_active_vms	smallint	The maximum active number of virtual machines for the aggregation period. For hourly aggregations, this is the maximum collected sample value. For daily aggregations, it is the maximum hourly average value.
total_vms	smallint	The average number of all virtual machines on the host for this aggregation.
max_total_vms	smallint	The maximum total number of virtual machines for the aggregation period. For hourly aggregations, this is the maximum collected sample value. For daily aggregations, it is the maximum hourly average value.
total_vms_vcpus	smallint	Total number of VCPUs allocated to the host.
max_total_vms_vcpus	smallint	The maximum total virtual machine VCPU number for the aggregation period. For hourly aggregations, this is the maximum collected sample value. For daily aggregations, it is the maximum hourly average value.
cpu_load	smallint	The CPU load of the host.
max_cpu_load	smallint	The maximum CPU load for the aggregation period. For hourly aggregations, this is the maximum collected sample value. For daily aggregations, it is the maximum hourly average value.
system_cpu_usage_percent	smallint	Used CPU percentage on the host.
max_cpu_usage_percent	smallint	The maximum system CPU usage for the aggregation period, expressed as a percentage. For hourly aggregations, this is the maximum collected sample value. For daily aggregations, it

Name	Type	Description
		is the maximum hourly average value.
user_cpu_usage_percent	smallint	Used user CPU percentage on the host.
max_user_cpu_usage_percent	smallint	The maximum user CPU usage for the aggregation period, expressed as a percentage. For hourly aggregations, this is the maximum collected sample value. For daily aggregations, it is the maximum hourly average value.
swap_used_mb	integer	Used swap size usage of the host in megabytes (MB).
max_swap_used_mb	integer	The maximum user swap size usage of the host for the aggregation period in megabytes (MB), expressed as a percentage. For hourly aggregations, this is the maximum collected sample value. For daily aggregations, it is the maximum hourly average value.
host_configuration_version	integer	The host configuration version at the time of sample.

9.1.4. v3_0_host_interface_samples_history_view \v3_0_host_interface_hourly_history_view \v3_0_host_interface_daily_history_view

Historical statistics for each host network interface in the system.

Table 9.4. v3_0_host_interface_samples_history_view\v3_0_host_interface_hourly_history_view \v3_0_host_interface_daily_history_view

Name	Type	Description
history_id	integer	The unique ID of this row in the table.
history_datetime	timestamp with time zone	The timestamp of this history view (rounded to minute, hour, day as per the aggregation level).
host_interface_id	uuid	Unique identifier of the interface in the system.
receive_rate_percent	smallint	Used receive rate percentage on the host.
max_receive_rate_percent	smallint	The maximum receive rate for the aggregation period,

Name	Type	Description
		expressed as a percentage. For hourly aggregations, this is the maximum collected sample value. For daily aggregations, it is the maximum hourly average value.
transmit_rate_percent	smallint	Used transmit rate percentage on the host.
max_transmit_rate_percent	smallint	The maximum transmit rate for the aggregation period, expressed as a percentage. For hourly aggregations, this is the maximum collected sample value. For daily aggregations, it is the maximum hourly average value.
host_interface_configuration_version	integer	The host interface configuration version at the time of sample.

9.1.5. v3_0_vm_samples_history_view \v3_0_vm_hourly_history_view\v3_0_vm_daily_history_view

Historical statistics for the virtual machines in the system.

Table 9.5. v3_0_vm_samples_history_view\v3_0_vm_hourly_history_view \v3_0_vm_daily_history_view

Name	Type	Description
history_id	integer	The unique ID of this row in the table.
history_datetime	timestamp with time zone	The timestamp of this history row (rounded to minute, hour, day as per the aggregation level).
vm_id	uuid	Unique ID of the virtual machine in the system.
vm_status	smallint	-1 - Unknown Status (used only to indicate problems with the ETL -- PLEASE NOTIFY SUPPORT) 0 - Down 1 - Up 2 - Paused 3 - Problematic
minutes_in_status	decimal	The total number of minutes that the virtual machine was in the status shown in the status

Name	Type	Description
		column for the aggregation period. For example, if a virtual machine was up for 55 minutes and down for 5 minutes during an hour, two rows will show for this hour. One will have a status of Up and minutes_in_status, the other will have a status of Down and a minutes_in_status of 5.
cpu_usage_percent	smallint	The percentage of the CPU in use by the virtual machine.
max_cpu_usage	smallint	The maximum CPU usage for the aggregation period, expressed as a percentage. For hourly aggregations, this is the maximum collected sample value. For daily aggregations, it is the maximum hourly average value.
memory_usage_percent	smallint	Percentage of used memory in the virtual machine. The guest tools must be installed on the virtual machine for memory usage to be recorded.
max_memory_usage	smallint	The maximum memory usage for the aggregation period, expressed as a percentage. For hourly aggregations, this is the maximum collected sample value. For daily aggregations, it is the maximum hourly average value. The guest tools must be installed on the virtual machine for memory usage to be recorded.
user_cpu_usage_percent	smallint	Used user CPU percentage on the host.
max_user_cpu_usage_percent	smallint	The maximum user CPU usage for the aggregation period, expressed as a percentage. For hourly aggregations, this is the maximum collected sample value. For daily aggregation, it is the maximum hourly average value.
system_cpu_usage_percent	smallint	Used system CPU percentage on the host.

Name	Type	Description
max_system_cpu_usage_percent	smallint	The maximum system CPU usage for the aggregation period, expressed as a percentage. For hourly aggregations, this is the maximum collected sample value. For daily aggregations, it is the maximum hourly average value.
vm_ip	varchar(255)	The IP address of the first NIC. Only shown if the guest agent is installed.
current_user_name	varchar(255)	Name of user logged into the virtual machine console, if a guest agent is installed.
currently_running_on_host	uuid	The unique ID of the host the virtual machine is running on.
vm_configuration_version	integer	The virtual machine configuration version at the time of sample.
current_host_configuration_version	integer	The current host the virtual machine is running on.

9.1.6. v3_0_vm_interface_samples_history_view \v3_0_vm_interface_hourly_history_view \v3_0_vm_interface_daily_history_view

Historical statistics for the virtual machine network interfaces in the system.

Table 9.6. v3_0_vm_interface_samples_history_view\v3_0_vm_interface_hourly_history_view \v3_0_vm_interface_daily_history_view

Name	Type	Description
history_id	integer	The unique ID of this row in the table.
history_datetime	timestamp with time zone	The timestamp of this history row (rounded to minute, hour, day as per the aggregation level).
vm_interface_id	uuid	Unique identifier of the interface in the system.
receive_rate_percent	smallint	Used receive rate percentage on the host.
max_receive_rate_percent	smallint	The maximum receive rate for the aggregation period, expressed as a percentage. For hourly aggregations, this is the maximum collected sample value. For daily aggregations, it

Name	Type	Description
		is the maximum hourly average value.
transmit_rate_percent	smallint	Used transmit rate percentage on the host.
max_transmit_rate_percent	smallint	The maximum transmit rate for the aggregation period, expressed as a percentage. For hourly aggregations, this is the maximum collected sample value. For daily aggregations, it is the maximum hourly average rate.
vm_interface_configuration_version	integer	The virtual machine interface configuration version at the time of sample.

9.1.7. v3_0_vm_disk_daily_history_view \v3_0_vm_disk_hourly_history_view \v3_0_vm_disk_samples_history_view

Historical statistics for the virtual disks in the system.

Table 9.7. v3_0_vm_disk_daily_history_view\v3_0_vm_disk_hourly_history_view \v3_0_vm_disk_samples_history_view

Name	Type	Description
history_id	integer	The unique ID of this row in the table.
history_datetime	timestamp with time zone	The timestamp of this history row (rounded to minute, hour, day as per the aggregation level).
vm_disk_id	uuid	Unique ID of the disk in the system.
vm_disk_status	integer	• 0 - Unassigned • 1 - OK • 2 - Locked • 3 - Invalid • 4 - Illegal
minutes_in_status	decimal	The total number of minutes that the virtual machine disk was in the status shown in the status column for the aggregation period. For example, if a virtual machine disk was locked for 55 minutes and OK for 5 minutes during

Name	Type	Description
		an hour, two rows will show for this hour. One will have a status of Locked and minutes_in_status of 55, the other will have a status of OK and a minutes_in_status of 5.
vm_actual_disk_size_mb	integer	The actual size allocated to the disk.
read_rate_bytes_per_second	integer	Read rate to disk in bytes per second.
max_read_rate_bytes_per_second	integer	The maximum read rate for the aggregation period. For hourly aggregations, this is the maximum collected sample value. For daily aggregations, it is the maximum hourly average value.
write_rate_bytes_per_second	integer	Write rate to disk in bytes per second.
max_write_rate_bytes_per_second	integer	The maximum write rate for the aggregation period. For hourly aggregations, this is the maximum collected sample value. For daily aggregations, it is the maximum hourly average value.
read_latency_seconds	decimal	The virtual machine disk read latency measured in seconds.
max_read_latency_seconds	decimal	The maximum write latency for the aggregation period, measured in seconds. For hourly aggregations, this is the maximum collected sample value. For daily aggregations, it is the maximum hourly average value.
write_latency_seconds	decimal	The virtual machine disk write latency measured in seconds.
max_write_latency_seconds	decimal	The maximum write latency for the aggregation period, measured in seconds. For hourly aggregations, this is the maximum collected sample value. For daily aggregations, it is the maximum hourly average value.
flush_latency_seconds	decimal	The virtual machine disk flush latency measured in seconds.

Name	Type	Description
max_flush_latency_seconds	decimal	The maximum flush latency for the aggregation period, measured in seconds. For hourly aggregations, this is the maximum collected sample value. For daily aggregations, it is the maximum hourly average value.
vm_disk_configuration_version	integer	The virtual machine disk configuration version at the time of sample.

9.2. Configuration History Views

This section describes the configuration views available to the user for querying and generating reports.

delete_date does not appear for living entities

`delete_date` does not appear in latest views because these views provide the latest configuration of living entities, which, by definition, have not been deleted.

9.2.1. v3_0_datacenter_configuration_view \v3_0_latest_datacenter_configuration_view

Data centers configuration history in the system.

Table 9.8. v3_0_datacenter_configuration_view\v3_0_latest_datacenter_configuration_view

Name	Type	Description
history_id	integer	The ID of the configuration version in the history database.
datacenter_id	uuid	The unique ID of the data center in the system.
datacenter_name	varchar(40)	Name of the data center, as displayed in the edit dialog.
datacenter_description	varchar(4000)	Description of the data center, as displayed in the edit dialog.
storage_type	smallint	• 0 -Unknown • 1 - NFS • 2 - FCP • 3 - iSCSI • 4 - Local • 6 - All

Name	Type	Description
create_date	timestamp with time zone	The date this entity was added to the system.
update_date	timestamp with time zone	The date this entity was changed in the system.
delete_date	timestamp with time zone	The date this entity was deleted from the system.

9.2.2. v3_0_datacenter_storage_domain_map_view \v3_0_latest_datacenter_configuration_view

A historical map showing the relationships between storage domains and data centers in the system.

Table 9.9. v3_0_datacenter_storage_domain_map_view\v3_0_latest_datacenter_configuration_view

Name	Type	Description
history_id	integer	The ID of the configuration version in the history database.
storage_domain_id	uuid	The unique ID of this storage domain in the system.
datacenter_id	uuid	The unique ID of the data center in the system.
attach_date	timestamp with time zone	The date the storage domain was attached to the data center.
detach_date	timestamp with time zone	The date the storage domain was detached from the data center.

9.2.3. v3_0_storage_domain_configuration_view \v3_0_latest_storage_domain_configuration_view

Storage domains configuration history in the system.

Table 9.10. v3_0_storage_domain_configuration_view \v3_0_latest_storage_domain_configuration_view

Name	Type	Description
history_id	integer	The ID of the configuration version in the history database.
storage_domain_id	uuid	The unique ID of this storage domain in the system.
storage_domain_name	varchar(250)	Storage domain name.
storage_domain_type	smallint	• 0 - Data (Master) • 1 - Data • 2 - ISO • 3 - Export
storage_type	smallint	• 0 - Unknown

Name	Type	Description
		• 1 - NFS
		• 2 - FCP
		• 3 - iSCSI
		• 4 - Local
		• 6 - All
create_date	timestamp with time zone	The date this entity was added to the system.
update_date	timestamp with time zone	The date this entity was changed in the system.
delete_date	timestamp with time zone	The date this entity was deleted from the system.

9.2.4. v3_0_cluster_configuration_view \v3_0_latest_cluster_configuration_view

Clusters configuration history in the system.

Table 9.11. v3_0_cluster_configuration_view\v3_0_latest_cluster_configuration_view

Name	Type	Description
history_id	integer	The ID of the configuration version in the history database.
cluster_id	uuid	The unique identifier of the datacenter this cluster resides in.
cluster_name	varchar(40)	Name of the cluster, as displayed in the edit dialog.
cluster_description	varchar(4000)	As defined in the edit dialog.
datacenter_id	uuid	The unique identifier of the datacenter this cluster resides in.
cpu_name	varchar(255)	As displayed in the edit dialog.
compatibility_version	varchar(40)	As displayed in the edit dialog.
datacenter_configuration_version	integer	The data center configuration version at the time of creation or update.
create_date	timestamp with time zone	The date this entity was added to the system.
update_date	timestamp with time zone	The date this entity was changed in the system.
delete_date	timestamp with time zone	The date this entity was deleted from the system.

9.2.5. v3_0_host_configuration_view \v3_0_latest_host_configuration_view

Host configuration history in the system.

Table 9.12. v3_0_host_configuration_view\v3_0_latest_host_configuration_view

Name	Type	Description
history_id	integer	The ID of the configuration version in the history database.
host_id	uuid	The unique ID of the host in the system.
host_unique_id	varchar(128)	This field is a combination of the host physical UUID and one of its MAC addresses, and is used to detect hosts already registered in the system.
host_name	varchar(255)	Name of the host (same as in the edit dialog).
cluster_id	uuid	The unique ID of the cluster that this host belongs to.
host_type	smallint	• 0 - RHEL Host • 2 - RHEV Hypervisor Node
fqn_or_ip	varchar(255)	The host's DNS name or its IP address for Red Hat Enterprise Virtualization Manager to communicate with (as displayed in the edit dialog).
memory_size_mb	integer	The host's physical memory capacity, expressed in megabytes (MB).
swap_size_mb	integer	The host swap partition size.
cpu_model	varchar(255)	The host's CPU model.
number_of_cores	smallint	Total number of CPU cores in the host.
host_os	varchar(255)	The host's operating system version.
pm_ip_address	varchar(255)	Power Management server IP address.
kernel_version	varchar(255)	The host's kernel version.
kvm_version	varchar(255)	The host's KVM version.
vdsm_version	varchar(40)	The host's VDSM version.
vdsm_port	integer	As displayed in the edit dialog.
cluster_configuration_version	integer	The cluster configuration version at the time of creation or update.

Name	Type	Description
create_date	timestamp with time zone	The date this entity was added to the system.
update_date	timestamp with time zone	The date this entity was changed in the system.
delete_date	timestamp with time zone	The date this entity was deleted from the system.

9.2.6. v3_0_host_configuration_view \v3_0_latest_host_interface_configuration_view

Host interface configuration history in the system.

Table 9.13. v3_0_host_configuration_view\v3_0_latest_host_configuration_view

Name	Type	Description
history_id	integer	The ID of the configuration version in the history database.
host_interface_id	uuid	The unique ID of this interface in the system.
host_interface_name	varchar(50)	The interface name as reported by the host.
host_id	uuid	Unique ID of the host this interface belongs to.
host_interface_type	smallint	• 0 - rt18139_pv • 1 - rt18139 • 2 - e1000 • 3 - pv
host_interface_speed_bps	integer	The interface speed in bits per second.
mac_address	varchar(20)	The interface MAC address.
network_name	varchar(50)	The logical network associated with the interface.
ip_address	varchar(50)	As displayed in the edit dialog.
gateway	varchar(20)	As displayed in the edit dialog.
bond	Boolean	A flag to indicate if this interface is a bonded interface.
bond_name	varchar(50)	The name of the bond this interface is part of (if it is part of a bond).
vlan_id	integer	As displayed in the edit dialog.
host_configuration_version	integer	The host configuration version at the time of creation or update.

Name	Type	Description
create_date	timestamp with time zone	The date this entity was added to the system.
update_date	timestamp with time zone	The date this entity was changed in the system.
delete_date	timestamp with time zone	The date this entity was deleted from the system.

9.2.7. v3_0_vm_configuration_view \v3_0_latest_vm_configuration_view

A list of all virtual machines in the system.

Table 9.14. v3_0_vm_configuration_view\v3_0_latest_vm_configuration_view

Name	Type	Description
history_id	integer	The ID of the configuration version in the history database.
vm_id	uuid	The unique ID of this VM in the system.
vm_name	varchar(255)	The name of the VM.
vm_description	varchar(4000)	As displayed in the edit dialog.
vm_type	smallint	• 0 - Desktop • 1 - Server
cluster_id	uuid	The unique ID of the cluster this VM belongs to.
template_id	uuid	The unique ID of the template this VM is derived from. The field is for future use, as the templates are not synchronized to the history database in this version.
template_name	varchar(40)	Name of the template from which this VM is derived.
cpu_per_socket	smallint	Virtual CPUs per socket.
number_of_sockets	smallint	Total number of virtual CPU sockets.
memory_size_mb	integer	Total memory allocated to the VM, expressed in megabytes (MB).
operating_system	smallint	• 0 - Unknown • 1 - Windows XP • 3 - Windows 2003 • 4 - Windows 2008 • 5 - Other Linux

Name	Type	Description
		• 6 - Other
		• 7 - RHEL 5
		• 8 - RHEL 4
		• 9 - RHEL 3
		• 10 - Windows2003 x64
		• 11 - Windows 7
		• 12 - Windows 7 x64
		• 13 - RHEL 5 x64
		• 14 - RHEL 4 x64
		• 15 - RHEL 3 x64
		• 16 - Windows 2008 x64
		• 17 - Windows 2008R2 x64
		• 18 - RHEL 6
		• 19 - RHEL 6 x64
ad_domain	varchar(40)	As displayed in the edit dialog.
default_host	uuid	As displayed in the edit dialog, the ID of the default host in the system.
high_availability	Boolean	As displayed in the edit dialog.
initialized	Boolean	A flag to indicate if this VM was started at least once for Sysprep initialization purposes.
stateless	Boolean	As displayed in the edit dialog.
fail_back	Boolean	As displayed in the edit dialog.
auto_suspend	Boolean	As displayed in the edit dialog.
usb_policy	smallint	As displayed in the edit dialog.
time_zone	varchar(40)	As displayed in the edit dialog.
cluster_configuration_version	integer	The cluster configuration version at the time of creation or update.
default_host_configuration_version	integer	The host configuration version at the time of creation or update.
create_date	timestamp with time zone	The date this entity was added to the system.
update_date	timestamp with time zone	The date this entity was changed in the system.

Name	Type	Description
delete_date	timestamp with time zone	The date this entity was deleted from the system.

9.2.8. v3_0_vm_configuration_view \latest_vm_interface_configuration_view

Virtual interfaces configuration history in the system

Table 9.15. v3_0_vm_configuration_view\latest_vm_interface_configuration_view

Name	Type	Description
history_id	integer	The ID of the configuration version in the history database.
vm_interface_id	uuid	The unique ID of this interface in the system.
vm_interface_name	varchar(50)	As displayed in the edit dialog.
vm_id	uuid	The ID of the virtual machine this interface belongs to.
vm_interface_type	smallint	The type of the virtual interface. • 0 - rt18139_pv • 1 - rt18139 • 2 - e1000 • 3 - pv
vm_interface_speed_bps	integer	The average speed of the interface during the aggregation in bits per second.
mac_address	varchar(20)	As displayed in the edit dialog.
network_name	varchar(50)	As displayed in the edit dialog.
vm_configuration_version	integer	The virtual machine configuration version at the time of creation or update.
create_date	timestamp with time zone	The date this entity was added to the system.
update_date	timestamp with time zone	The date this entity was changed in the system.
delete_date	timestamp with time zone	The date this entity was deleted from the system.

9.2.9. v3_0_disks_vm_map_view\v3_0_latest_disks_vm_map_view

A historical map showing the relationships between virtual disks and virtual machines in the system.

Table 9.16. v3_0_disks_vm_map_view\v3_0_latest_disks_vm_map_view

Name	Type	Description
history_id	integer	The ID of the configuration version in the history database.
vm_disk_id	uuid	The unique ID of this virtual disk in the system.
vm_id	uuid	The unique ID of the virtual machine in the system.
attach_date	timestamp with time zone	The date the virtual disk was attached to the virtual machine.
detach_date	timestamp with time zone	The date the virtual disk was detached from the virtual machine.

9.2.10. v3_0_vm_disk_configuration_view \v3_0_latest_vm_disk_configuration_view

Virtual disks configuration history in the system.

Table 9.17. v3_0_vm_disk_configuration_view\v3_0_latest_vm_disk_configuration_view

Name	Type	Description
history_id	integer	The ID of the configuration version in the history database.
vm_disk_id	uuid	The unique ID of this disk in the system.
storage_domain_id	uuid	The ID of the storage domain this disk image belongs to.
vm_internal_drive_mapping	varchar	The virtual machine internal drive mapping.
vm_disk_description	varchar(4000)	As displayed in the edit dialog.
vm_disk_space_size_mb	integer	The defined size of the disk in megabytes (MB).
disk_type	integer	As displayed in the edit dialog. Only System and data are currently used. • 0 - Unassigned • 1 - System • 2 - Data • 3 - Shared • 4 - Swap • 5 - Temp
vm_disk_format	integer	As displayed in the edit dialog. • 3 - Unassigned

Name	Type	Description
		• 4 - COW
		• 5 - RAW
vm_disk_interface	integer	• 0 - IDE
		• 1 - SCSI (not supported)
		• 2 - VirtIO
create_date	timestamp with time zone	The date this entity was added to the system.
update_date	timestamp with time zone	The date this entity was changed in the system.
delete_date	timestamp with time zone	The date this entity was deleted from the system.

Appendix A. Additional References

These additional documentation resources do not form part of the Red Hat Enterprise Virtualization documentation suite. They do however contain useful information for System Administrators managing Red Hat Enterprise Virtualization environments and are available at *http://docs.redhat.com/*.

Red Hat Enterprise Linux — Deployment 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 — Hypervisor Deployment Guide
> The complete guide to installing, deploying and maintaining Red Hat Enterprise Virtualization Hypervisors.

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

Appendix B. Minimum requirements and supported limits

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

B.1. data center

In a managed virtual environment the highest level container for all resources is the data center. A number of limitations apply to the resources which can be contained within each data center.

Table B.1. data center Limitations

Item	Limitations
Number of storage domains	• A minimum of 2 storage domains per data center is recommended. One data storage domain is required, and an ISO storage domain per data center is recommended.
Number of hosts	• A maximum of 200 hosts per data center is supported.

B.2. cluster

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 — Further Information**
>
> Further information about clusters is available in the *Red Hat Enterprise Virtualization Administration Guide*.

B.3. Storage Domain

Storage domains provide space for the storage of virtual machine 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 B.2. storage domain Limitations

Item	Limitations
Storage Types	• Supported storage types are:

Item	Limitations
	• Fibre Channel Protocol (FCP) • Internet Small Computer System Interface (iSCSI) • Network File System (NFS) • All data storage domains within a data center must be of the same type. The type is specified as a step in the creation of the storage domain. • The data storage domain can be any of FCP, iSCSI, and NFS • Legacy FCP or iSCSI export storage domains from Red Hat Enterprise Virtualization 2.2 environments can be attached to data centers in Red Hat Enterprise Virtualization 3.0. New ISO and export storage domains must be provided by NFS.
Logical Unit Numbers (LUNs)	• No more than 300 LUNs are permitted for each storage domain that is provided by iSCSI or FCP.

> **Note — Further Information**
>
> Further information about storage domains is available in the *Red Hat Enterprise Virtualization Administration Guide*.

B.4. Red Hat Enterprise Virtualization Manager

Red Hat Enterprise Virtualization Manager servers must run Windows Server 2008 (R2). A number of additional hardware requirements must also be met.

Table B.3. Red Hat Enterprise Virtualization Manager Limitations

Item	Limitations
RAM	• A minimum of 3 GB of RAM is required.
PCI Devices	• At least one network controller with a minimum bandwidth of 1 Gbps is recommended
Storage	• A minimum of 3 GB of available local disk space is recommended.

> **Note — Further Information**
>
> Further information about Red Hat Enterprise Virtualization Manager is available in the *Red Hat Enterprise Virtualization Installation Guide*.

B.5. Hypervisor Requirements

Red Hat Enterprise Virtualization Hypervisors have a number of hardware requirements and supported limits.

Table B.4. Red Hat Enterprise Virtualization Hypervisor Requirements and Supported Limits

Item	Support Limit
CPU	• A minimum of 1 physical CPU is required. All CPUs must support: • the Intel® 64 or AMD64 CPU extensions, and • the AMD-V™ or Intel VT® hardware virtualization extensions. • A maximum of 128 physical CPUs is supported.
RAM	• A minimum of 512 MB of RAM is required. • A minimum of an additional 512 MB for each virtual machine is recommended. The amount of RAM required for each guest varies depending on: • the guest operating system's requirements, • the guests' application requirements, and • memory activity and usage of guests. Additionally KVM is able to over-commit physical RAM for virtualized guests. It does this by only allocating RAM for guests as required and shifting underutilized guests into swap. • A maximum of 1 TB of RAM is supported.
Storage	The minimum supported internal storage for a Hypervisor is the total of the following list: • The root partitions require at least 512 MB of storage. • The configuration partition requires at least 8 MB of storage.

Item	Support Limit
	• The recommended minimum size of the logging partition is 2048 MB.
	• The data partition requires at least 256 MB of storage. Use of a smaller data partition may prevent future upgrades of the Hypervisor from the Red Hat Enterprise Virtualization Manager. By default all disk space remaining after allocation of swap space will be allocated to the data partition.
	• The swap partition requires at least 8 MB of storage. The recommended size of the swap partition varies depending on both the system the Hypervisor is being installed upon and the anticipated level of overcommit for the environment. Overcommit allows the Red Hat Enterprise Virtualization environment to present more RAM to guests than is actually physically present. The default overcommit ratio is `0.5`.
	The recommended size of the swap partition can be determined by:
	• Multiplying the amount of system RAM by the expected overcommit ratio, and adding
	• 2 GB of swap space for systems with 4 GB of RAM or less, or
	• 4 GB of swap space for systems with between 4 GB and 16 GB of RAM, or
	• 8 GB of swap space for systems with between 16 GB and 64 GB of RAM, or
	• 16 GB of swap space for systems with between 64 GB and 256 GB of RAM.
	Example B.1. Calculating Swap Partition Size
	For a system with 8 GB of RAM this means the formula for determining the amount of swap space to allocate is:
	```\n(8 GB x 0.5) + 4 GB = 8 GB\n```
	Please note that these are the *minimum* storage requirements for Hypervisor installation. It is recommended to use the default allocations which use more storage space.

Item	Support Limit
PCI Devices	• At least one network controller is required with a recommended minimum bandwidth of 1 Gbps.

Important — Virtualization Extensions

When the Red Hat Enterprise Virtualization Hypervisor 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 Hypervisor 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.

> ⭐ **Important — `Fakeraid` Devices are not Supported**
>
> The Red Hat Enterprise Virtualization Hypervisor does not support installation on `fakeraid` devices. Where a `fakeraid` device is present it must be reconfigured such that it no longer runs in RAID mode.
>
> 1. Access the RAID controller's BIOS and remove all logical drives from it.
>
> 2. Change controller mode to be non-RAID. This may be referred to as compatibility or JBOD mode.
>
> Access the manufacturer provided documentation for further information related to the specific device in use.

B.6. Guest Requirements and Support Limits

The following requirements and support limits apply to guests that are run on the Hypervisor:

Table B.5. Virtualized Hardware

Item	Limitations
CPU	• A maximum of 64 virtualized CPUs per guest is supported.
RAM	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. A number of support limits also apply. • A minimum of 512 MB of virtualized RAM per guest is supported. Creation of guests with less than 512 MB of virtualized RAM while possible is not supported. • A maximum of 256 GB of virtualized RAM per 64 bit guest is supported. • A maximum of 4 GB of virtualized RAM per 32 bit guest is supported. Note that not all 32 bit operating systems are able to register an entire 4 GB of RAM.
PCI devices	• A maximum of 32 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.

Item	Limitations
Storage	• A maximum of 8 virtualized storage devices per guest is supported.

B.7. SPICE

SPICE currently supports a maximum resolution of 2560x1600 pixels.

Appendix C. Virtualized Hardware

Red Hat Enterprise 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.

C.1. Central Processing Unit (CPU)

Each Red Hat Enterprise Virtualization Hypervisor within a Cluster has a number of *virtual CPUs* (vCPUS). The virtual CPUs are in turn exposed to guests running on the hypervisors. All virtual CPUs exposed by Hypervisors within a Cluster are of the type selected when the Cluster was initially created via Red Hat Enterprise 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.

`AMD Opteron G1`.
> See *Table C.1, "AMD Opteron G1"* for specifications.

`AMD Opteron G2`
> See *Table C.2, "AMD Opteron G2"* for specifications.

`AMD Opteron G3`
> See *Table C.3, "AMD Opteron G3"* for specifications.

`Intel Xeon Core 2`
> See *Table C.4, "Intel Xeon Core2"* for specifications.

`Intel Xeon 45nm Core2`
> See *Table C.5, "Intel Xeon 45nm Core2"* for specifications.

```
Intel Xeon Core i7
```
    See *Table C.6, "Intel Xeon Core i7"* for specifications.

> ### Note — Support for *x2APIC*
>
> All virtual CPU models provided by Red Hat Enterprise Linux 6 hosts include support for *x2APIC*. This provides an *Advanced Programmable Interrupt Controller (APIC)* to better handle hardware interrupts.

C.1.1. CPU Specifications

The reference tables provided in this section detail the specifications of the virtual CPUs which Red Hat Enterprise Virtualization is able to expose to guests.

Table C.1. AMD Opteron G1

Field	Value
`model`	`Opteron_G1 AMD Opteron 240 (Gen 1 Class Opteron)`
`family`	`15`
`model`	`6`
`stepping`	`1`
`level`	`5`
`xlevel`	`0x80000008`
`vendor`	`AuthenticAMD`
`feature_edx`	`sse2 sse fxsr mmx clflush pse36 pat cmov mca pge mtrr sep apic cx8 mce pae msr tsc pse de fpu`
`feature_ecx`	`x2apic pni|sse3`
`extfeature_edx`	`lm fxsr mmx nx pat cmov pge syscall apic cx8 mce pae msr tsc pse de fpu`
`extfeature_ecx`	

Table C.2. AMD Opteron G2

Field	Value
`model`	`Opteron_G2 AMD Opteron 22xx (Gen 2 Class Opteron)`
`family`	`15`
`model`	`6`
`stepping`	`1`
`level`	`5`
`xlevel`	`0x80000008`
`vendor`	`AuthenticAMD`
`feature_edx`	`sse2 sse fxsr mmx clflush pse36 pat cmov mca pge mtrr sep apic cx8 mce pae msr tsc pse de fpu`

Field	Value
feature_ecx	x2apic cx16 pni|sse3
extfeature_edx	lm rdtscp fxsr mmx nx pat cmov pge syscall apic cx8 mce pae msr tsc pse de fpu
extfeature_ecx	svm lahf_lm

Table C.3. AMD Opteron G3

Field	Value
model	Opteron_G3 AMD Opteron 23xx (Gen 3 Class Opteron)
family	15
model	6
stepping	1
level	5
xlevel	0x80000008
vendor	AuthenticAMD
feature_edx	sse2 sse fxsr mmx clflush pse36 pat cmov mca pge mtrr sep apic cx8 mce pae msr tsc pse de fpu
feature_ecx	popcnt x2apic cx16 monitor pni|sse3
extfeature_edx	lm rdtscp fxsr mmx nx pat cmov pge syscall apic cx8 mce pae msr tsc pse de fpu
extfeature_ecx	misalignsse sse4a abm svm lahf_lm

Table C.4. Intel Xeon Core2

Field	Value
model	Conroe Intel Celeron_4x0 (Conroe/ Merom Class Core 2)
family	6
model	15
stepping	3
level	2
xlevel	0x8000000a
vendor	GenuineIntel
feature_edx	sse2 sse fxsr mmx clflush pse36 pat cmov mca pge mtrr sep apic cx8 mce pae msr tsc pse de fpu
feature_ecx	x2apic ssse3 pni|sse3
extfeature_edx	lm fxsr mmx nx pat cmov pge syscall apic cx8 mce pae msr tsc pse de fpu
extfeature_ecx	lahf_lm

Table C.5. Intel Xeon 45nm Core2

Field	Value
model	Penryn Intel Core 2 Duo P9xxx (Penryn Class Core 2)
family	6
model	23
stepping	3
level	2
xlevel	0x8000000a
vendor	GenuineIntel
feature_edx	sse2 sse fxsr mmx clflush pse36 pat cmov mca pge mtrr sep apic cx8 mce pae msr tsc pse de fpu
feature_ecx	x2apic sse4.1|sse4_1 cx16 ssse3 pni|sse3
extfeature_edx	lm fxsr mmx nx pat cmov pge syscall apic cx8 mce pae msr tsc pse de fpu
extfeature_ecx	lahf_lm

Table C.6. Intel Xeon Core i7

Field	Value
model	Nehalem Intel Core i7 9xx (Nehalem Class Core i7)
family	6
model	26
stepping	3
level	2
xlevel	0x8000000a
vendor	GenuineIntel
feature_edx	sse2 sse fxsr mmx clflush pse36 pat cmov mca pge mtrr sep apic cx8 mce pae msr tsc pse de fpu
feature_ecx	popcnt x2apic sse4.2|sse4_2 sse4.1|sse4_1 cx16 ssse3 pni|sse3
extfeature_edx	lm fxsr mmx nx pat cmov pge syscall apic cx8 mce pae msr tsc pse de fpu
extfeature_ecx	lahf_lm

C.2. 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.

C.3. Network devices

Red Hat Enterprise 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 Enterprise 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. Information on the number of PCI devices that can be exposed to each guest is available in *Section B.6, "Guest Requirements and Support Limits"*

C.4. 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 **Bochs** VESA extensions (hardware level, including all non-standard modes).

C.5. 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).

C.6. 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.

C.7. 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.

C.8. 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).

Appendix D. Revision History

Revision 1-0   **Mon Oct 17 2011**   **Tim Hildred** *thildred@redhat.com*

Made changes suggested by QA.

Revision 1-0   **Tues Oct 4 2011**   **Tim Hildred** *thildred@redhat.com*

Polished guide, integrated more feedback.

Revision 1-0   **Fri Sept 19 2011**   **Tim Hildred** *thildred@redhat.com*

Integrated SME feedback, added links to ease navigation through document.

Revision 1-0   **Fri Sept 9 2011**   **Tim Hildred** *thildred@redhat.com*

Updated existing content. Generated new content. Prepared for draft release with Red Hat
Enterprise Virtualization Beta 2.

Revision 1-0   **Tue Aug 24 2010**   **Stephen Gordon** *sgordon@redhat.com*

Initial edition.