

Red Hat Enterprise Linux 6

Virtualization Getting Started Guide

Virtualization Documentation



Red Hat Enterprise Linux 6 Virtualization Getting Started Guide

Virtualization Documentation

Edition 0.2

Author

Copyright © 2011 Red Hat, Inc.

The text of and illustrations in this document are licensed by Red Hat under a Creative Commons Attribution–Share Alike 3.0 Unported license ("CC-BY-SA"). An explanation of CC-BY-SA is available at <http://creativecommons.org/licenses/by-sa/3.0/>. In accordance with CC-BY-SA, if you distribute this document or an adaptation of it, you must provide the URL for the original version.

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

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

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

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

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

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

All other trademarks are the property of their respective owners.

1801 Varsity Drive
Raleigh, NC 27606-2072 USA
Phone: +1 919 754 3700
Phone: 888 733 4281
Fax: +1 919 754 3701

The Red Hat Enterprise Linux Virtualization Getting Started Guide describes the basics of virtualization and the virtualization products and technologies that are available with Red Hat Enterprise Linux.

Preface	v
1. Document Conventions	v
1.1. Typographic Conventions	v
1.2. Pull-quote Conventions	vi
1.3. Notes and Warnings	vii
2. Getting Help and Giving Feedback	vii
2.1. Do You Need Help?	vii
2.2. We Need Feedback!	viii
1. Introduction	1
1.1. Who should read this guide?	1
1.2. Red Hat Enterprise Virtualization (RHEV)	1
2. Getting started with Virtualization	3
2.1. What is virtualization?	3
2.2. Migration	3
2.3. Virtualized to virtualized migration (V2V)	4
3. Advantages and misconceptions of virtualization	5
3.1. Virtualization costs	5
3.2. Virtualization learning curve	5
3.3. Performance	5
3.4. Disaster recovery	6
3.5. Security	6
3.5.1. Virtualization security features	6
3.6. Virtualization for servers and individuals	7
4. Introduction to Red Hat virtualization products	9
4.1. KVM and virtualization in Red Hat Enterprise Linux	9
4.2. libvirt and the libvirt tools	10
4.3. Virtualized hardware devices	11
4.3.1. Virtualized and emulated devices	11
4.3.2. Para-virtualized drivers	13
4.3.3. Physical host devices	14
4.3.4. Guest CPU models	15
4.4. Storage	16
4.4.1. Storage pools	17
5. Virtualization Tools	19
5.1. virsh	19
5.2. virt-manager	19
5.3. virt-install	19
5.4. guestfish	19
5.5. Other useful tools	20
A. Revision History	25

Preface

1. 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](https://fedorahosted.org/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.

1.1. 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:

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

¹ <https://fedorahosted.org/liberation-fonts/>

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.

1.2. 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:

```
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"));
    }
}
```

1.3. 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.

2. Getting Help and Giving Feedback

2.1. Do You Need Help?

If you experience difficulty with a procedure described in this documentation, visit the Red Hat Customer Portal at <http://access.redhat.com>. Through the customer portal, you can:

- search or browse through a knowledgebase of technical support articles about Red Hat products.
- submit a support case to Red Hat Global Support Services (GSS).

- access other product documentation.

Red Hat also hosts a large number of electronic mailing lists for discussion of Red Hat software and technology. You can find a list of publicly available mailing lists at <https://www.redhat.com/mailman/listinfo>. Click on the name of any mailing list to subscribe to that list or to access the list archives.

2.2. 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 Linux**.

When submitting a bug report, be sure to mention the manual's identifier: *doc-Virtualization_Getting_Started_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, please include the section number and some of the surrounding text so we can find it easily.

Introduction

The Virtualization Getting Started Guide introduces the basics of virtualization and assists with the navigation of other virtualization documentation and products that Red Hat provides.

This guide also explains the advantages of virtualization and dispels some common myths that exist regarding virtualization.

1.1. Who should read this guide?

This guide is designed for anyone wishing to understand the basics of virtualization, but may be of particular interest to:

- Readers who are new to virtualization, who may be unsure of the benefits offered.
- Those considering deployment of virtualized machines in their environment.
- Those looking for an overview of the virtualization technologies that Red Hat produces and supports.

1.2. Red Hat Enterprise Virtualization (RHEV)

Red Hat Enterprise Virtualization for Servers is an end-to-end virtualization solution that is designed to enable pervasive datacenter virtualization, and significantly enhance capital and operational efficiency. Red Hat Enterprise Virtualization 2.2 introduces several new features including desktop virtualization with the Red Hat Enterprise Virtualization for Desktop add-on subscription.

With Red Hat Enterprise Virtualization for Desktops, complete desktop environments are hosted as virtual desktops on servers located in a centralized datacenter. Users connect to these virtual desktops using either inexpensive thin clients or repurposed PCs.

Information on both these products can be found at <http://www.redhat.com/virtualization/rhev/>.

The full collection of Red Hat Enterprise Virtualization (RHEV) documentation can be found at <http://docs.redhat.com/>.

In addition to the documentation for Red Hat Enterprise Virtualization products and this guide, the following titles cover virtualization with Red Hat Enterprise Linux:

- *Virtualization Host Configuration and Guest Installation Guide*: This guide provides information on system requirements and restrictions, package details, host configuration and detailed instructions for installing different types of guests.
- *Virtualization Administration Guide*: This guide provides information on best server practices, security, KVM, remote management of guests, KSM, administration tasks, storage, volumes, virt-manager, guest disk access with offline tools, virtual networking, and troubleshooting.

Getting started with Virtualization

2.1. What is virtualization?

Virtualization is a broad computing term used for running software, usually multiple operating systems, concurrently and in isolation from other programs on a single system. Most existing implementations of virtualization use a *hypervisor*, a software layer or subsystem, sometimes called a *domain*, that controls hardware and provides *guest* operating systems with access to underlying hardware. The hypervisor allows multiple operating systems, called *guests*, to run on the same physical system by offering virtualized hardware to the guest operating system. There are various methods for virtualizing operating systems:

Full virtualization

Full virtualization uses the hardware features of the processor to provide guests with total abstraction of the underlying physical system. This creates a new virtual system, called a *virtual machine*, or *VM* for short, in which guest operating systems can run without modifications. The guest operating system and any applications on the guest are unaware of their virtualized environment and run normally. Hardware-assisted virtualization is the technique used for full virtualization with KVM (Kernel-based Virtual Machine) in Red Hat Enterprise Linux.

Para-virtualization

Para-virtualization employs a collection of software and data structures that are presented to the guest virtual machine, requiring software modifications in the guest to use the para-virtualized environment. Para-virtualization can encompass the entire kernel, as is the case for Xen para-virtualized guests, or simply drivers that virtualize I/O devices.

Software virtualization (or emulation)

Software virtualization uses slower binary translation and other emulation techniques to run unmodified operating systems. Software virtualization is unsupported by Red Hat Enterprise Linux.



Note

For more information and detailed instructions on guest installation, refer to the *Red Hat Enterprise Linux 6 Virtualization Host Configuration and Guest Installation Guide*.

2.2. Migration

Migration describes the process of moving a virtualized guest from one host to another. This is possible because guests are running in a virtualized environment instead of directly on the hardware.

Migration is useful for:

Load balancing

When a host machine is overloaded, one or many of its guests could be migrated to hosts with less load.

Hardware independence

When the need arises to upgrade, add, or remove hardware devices on the host, guests can be safely relocated to other hosts. This means that guests do not experience any downtime due to hardware improvements.

Energy saving

Guests can be redistributed to other hosts and the unloaded host systems can be powered off to save energy and cut costs in low usage periods.

Geographic migration

Guests can be moved to another physical location for lower latency or for other special circumstances.

It is important to understand that migration only moves the virtualized guest's memory. The guest's storage is located on networked devices, which are shared between the source host and destination hosts.

Shared, networked storage must be used for storing guest images to be migrated. Without shared storage, migration is not possible. It is recommended to use libvirt-managed storage pools for shared storage.

Offline migration

An offline migration suspends the guest, and then moves an image of the guest's memory to the destination host. The guest is then resumed on the destination host and the memory used by the guest on the source host is freed.

Live migration

Live migration is the process of migrating a *running* guest from one physical host to another physical host.



Note

For more information on migration refer to the *Red Hat Enterprise Linux 6 Virtualization Administration Guide*.

2.3. Virtualized to virtualized migration (V2V)

Red Hat Enterprise Linux 6 provides tools for converting virtualized guests from other types of hypervisors to KVM. The `virt-v2v` tool converts and imports virtual machines from Xen, other versions of KVM, and VMware ESX.



Note

For more information on V2V, refer to the *Red Hat Enterprise Linux 6 V2V Guide*.

Advantages and misconceptions of virtualization

There are many advantages to virtualization and perhaps an equal amount of misconceptions surrounding it. This chapter will explore these points.

3.1. Virtualization costs

A common misconception is that virtualization is too expensive to justify the change. Virtualization can be expensive to start up but often it saves money in the long term. It is important to perform a Return on Investment (ROI) analysis to determine the best use of virtualization in your environment, but consider the following points:

Less power

Using virtualization negates much of the need for multiple physical platforms. This equates to less power being drawn, resulting in reduced energy costs.

Less cooling

Similar to the previous point, needing less physical platforms means requiring less cooling. The initial cost of the physical platforms and required cooling, combined with their maintenance and power consumption is drastically cut by using virtualization.

Less maintenance time

Provided adequate planning is performed before migrating physical systems to virtualized ones, less time is spent maintaining them. This means less money being spent on parts and labor.

Maintaining installed software

Older versions of software may not run on newer, bare metal machines directly. However, by running the older software virtually on a larger, faster system, the life of the software may be extended while taking advantage of the performance from the newer system.

Predictable costs

A Red Hat Enterprise Linux subscription provides support for virtualization at the normal fixed rate, making it easy to predict costs.

Less space

By consolidating servers onto fewer machines, less physical space is taken up. This can mean the space normally taken up with server hardware can be used for something else, new systems for example.

3.2. Virtualization learning curve

A misconception exists that virtualization is difficult to learn. In truth, virtualization is no more difficult or easy to learn than any new process. The skills required for managing and supporting a physical environment are easily transferable to a virtual one. Virtual environments function much the same as their physical counterparts ensuring the learning curve remains a slight one.

3.3. Performance

On older versions of virtualization that only supported a single CPU in a guest VM, a noticeable performance hit was experienced. This limitation created a longer lasting misconception that current virtualization solutions are slow. This is no longer the case. Advances in technology allow virtual machines to run at much faster speeds.

3.4. Disaster recovery

Disaster recovery is quicker and easier when the systems are virtualized. On a physical system, if something serious goes wrong, a complete re-install of the operating system is usually required. This makes the recovery time take hours. However, if the systems are virtualized this is much faster. If the requirements for live migration are followed, virtualized machines can be restarted on another host and the longest possible delay would be restoring guest data. Also, because each of the virtualized systems are completely separate to each other, the downtime of one will have no affect on any others.

3.5. Security

A virtual machine uses SELinux and sVirt in order to improve security in virtualization. This section will include an overview of the security options available.

3.5.1. Virtualization security features

SELinux

SELinux was developed by the US National Security Agency and others to provide Mandatory Access Control (MAC) for Linux. Under control of SELinux, all processes and files are given what is known as a *type*, and access is limited by fine-grained controls. SELinux limits the abilities of an attacker and works to prevent many common security exploits such as buffer overflow attacks and privilege escalation.

SELinux strengthens the security model of Red Hat Enterprise Linux hosts and virtualized Red Hat Enterprise Linux guests. SELinux is configured and tested to work, by default, with all virtualization tools shipped with Red Hat Enterprise Linux 6.



Note

For more information on SELinux, refer to the SELinux documentation at <http://docs.redhat.com>.



Note

For more information on specific details regarding SELinux and its relationship to virtualization, refer to the *Red Hat Enterprise Linux 6 Virtualization Administration Guide*.

sVirt

sVirt is a technology included in Red Hat Enterprise Linux 6 that integrates SELinux and virtualization. It applies Mandatory Access Control (MAC) to improve security when using virtualized guests, and improves security and hardens the system against bugs in the hypervisor that might be used as an attack vector for the host or to another virtualized guest.

**Note**

For more information on sVirt, refer to the *Red Hat Enterprise Linux 6 Virtualization Administration Guide*.

3.6. Virtualization for servers and individuals

Virtualization is not just for servers, it can be useful for individuals as well. Desktop virtualization offers centralized management, an improved desktop solution, and better disaster recovery. Using connection software it is possible to connect to a desktop remotely.

For servers, virtualization is not only for larger networks, but for any situation with two or more servers. It provides live migration, high availability, fault tolerance, and streamlined backups.

Introduction to Red Hat virtualization products

This chapter introduces the various virtualization products available in Red Hat Enterprise Linux.

4.1. KVM and virtualization in Red Hat Enterprise Linux

What is KVM?

KVM (Kernel-based Virtual Machine) is a full virtualization solution for Linux on AMD64 and Intel 64 hardware that is built into the standard Red Hat Enterprise Linux 6 kernel. It can run multiple, unmodified virtualized guest Windows and Linux operating systems. The KVM hypervisor in Red Hat Enterprise Linux is managed with the libvirt API and tools built for libvirt (such as **virt-manager** and **virsh**). Virtualized guests are executed and run as multi-threaded Linux processes which are controlled by these tools.

Overcommitting

The KVM hypervisor supports *overcommitting* of system resources. Overcommitting means allocating more virtualized CPUs or memory than the available resources on the system. Memory overcommitting allows hosts to utilize memory and virtual memory to increase guest densities.



Important

A single guest can **not** use more CPU or memory than physically available. Overcommitting does, however, support the operation of **multiple** guests that have a total CPU and/or memory requirement greater than the physical host.

Overcommitting involves possible risks to system stability. For more information on overcommitting with KVM, and the precautions that should be taken, refer to the *Red Hat Enterprise Linux 6 Virtualization Administration Guide*.

KSM

Kernel SamePage Merging (KSM) is used by the KVM hypervisor to allow KVM guests to share identical memory pages. These shared pages are usually common libraries or other identical, high-use data. KSM allows for greater guest density of identical or similar guest operating systems by avoiding memory duplication.



Note

For more information on KSM, refer to the *Red Hat Enterprise Linux 6 Virtualization Administration Guide*.

KVM Guest VM Compatibility

To verify whether your processor supports the virtualization extensions and for information on enabling the virtualization extensions if they are disabled, refer to the *Red Hat Enterprise Linux Virtualization Administration Guide*.

Red Hat Enterprise Linux 6 servers have certain support limits.

The following URLs explain the processor and memory amount limitations for Red Hat Enterprise Linux:

- For host systems: <http://www.redhat.com/rhel/compare/>
- For hypervisors: <http://www.redhat.com/rhel/virtualization/compare/>

The following URL shows a complete chart of supported operating systems and host and guest combinations:

- http://www.redhat.com/rhel/server/virtualization_support.html#virt_matrix

4.2. libvirt and the libvirt tools

Libvirt is a hypervisor-independent virtualization API that is able to interact with the virtualization capabilities of a range of operating systems.

Libvirt provides:

- A common, generic and stable layer to securely manage virtualized guests on a host.
- A common interface for managing local systems and networked hosts.
- All of the APIs required to provision, create, modify, monitor, control, migrate and stop virtualized guests if the hypervisor supports these operations. Although multiple hosts may be accessed with libvirt simultaneously, the APIs are limited to single node operations.

Libvirt is designed as a building block for higher level management tools and applications, for example, **virt-manager** and the **virsh** command line management tools. Libvirt focuses on managing single hosts, with the exception of migration capabilities and provides APIs to enumerate, monitor and use the resources available on the managed node, including CPUs, memory, storage, networking and Non-Uniform Memory Access (NUMA) partitions. The management tools can be located on separate physical machines from the host using secure protocols.

Red Hat Enterprise Linux 6 supports libvirt and included libvirt-based tools as its default method for virtualization management.

Libvirt is available as free software under the GNU Lesser General Public License. The libvirt project aims to provide a long term stable C API to virtualization management tools, running on top of varying hypervisor technologies. The *libvirt* package supports Xen on Red Hat Enterprise Linux 5, and it supports KVM on both Red Hat Enterprise Linux 5 and Red Hat Enterprise Linux 6.

virsh

The **virsh** command-line tool is built on the **libvirt** management API and operates as an alternative to the graphical **virt-manager** application. The **virsh** command can be used in read-only mode by unprivileged users or, with root access, full administration functionality. The **virsh** command is ideal for scripting virtualization administration.

The **virsh** tool is included in the *libvirt-client* package.

virt-manager

virt-manager is a graphical desktop tool for managing virtualized guests. It can be used to perform virtualization administration, virtualized guest creation, migration and configuration tasks and allows access to graphical guest consoles. The ability to view virtualized guests, host statistics, device information and performance graphs is also provided. The local hypervisor and remote hypervisors can be managed through a single interface.



Note

For more information on **virt-manager**, refer to the *Red Hat Enterprise Linux 6 Virtualization Administration Guide*.

4.3. Virtualized hardware devices

Virtualization on Red Hat Enterprise Linux 6 presents three distinct types of system devices to virtualized guests. The three types include:

- Emulated software devices.
- Para-virtualized devices.
- Physically shared devices.

These hardware devices all appear as being physically attached to the virtualized guest but the device drivers work in different ways.

4.3.1. Virtualized and emulated devices

KVM implements many core devices for virtualized guests in software. These emulated hardware devices are crucial for virtualizing operating systems.

Emulated devices are virtual devices which exist entirely in software.

Emulated drivers may use either a physical device or a virtual software device. Emulated drivers are a translation layer between the guest and the Linux kernel (which manages the source device). The device level instructions are completely translated by the KVM hypervisor. Any device, of the same type (that is, storage, network, keyboard, and mouse) and recognized by the Linux kernel, may be used as the backing source device for the emulated drivers.

Virtualized CPUs (vCPUs)

A host system has a number of virtual CPUs (vCPUs) that can be presented to guest operating systems for their use. The number of virtual CPUs that can be offered to guests is finite and is determined by the number of physical processor cores on the host.

Emulated graphics devices

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

- A Cirrus CLGD 5446 PCI VGA card (using the *cirrus* device).
- A standard VGA graphics card with Bochs VESA extensions (hardware level, including all non-standard modes).

Emulated system components

The following core system components are emulated to provide basic system functions:

- Intel i440FX host PCI bridge.
- PIIX3 PCI to ISA bridge.
- PS/2 mouse and keyboard.
- EvTouch USB Graphics Tablet.
- PCI UHCI USB controller and a virtualized USB hub.
- PCI network adapters.
- Emulated serial ports.
- EHCI controller, virtualized USB storage and a USB mouse

Emulated sound devices

Red Hat Enterprise Linux 6.1 and above provide an emulated (Intel) HDA sound device, `intel-hda`. This device is supported on the following guest operating systems:

- Red Hat Enterprise Linux 6, for i386 and x86_64 architectures.
- Red Hat Enterprise Linux 5, for i386 and x86_64 architectures.
- Red Hat Enterprise Linux 4, for i386 and x86_64 architectures.
- Windows 2008 R2, for the x86_64 architecture.
- Windows 7, for i386 and x86_64 architectures.

The following two emulated sound devices are also available but are not recommended due to compatibility issues with certain guests:

- `ac97`, an emulated Intel 82801AA AC97 Audio compatible sound card.
- `es1370`, an emulated ENSONIQ AudioPCI ES1370 sound card.

Emulated watchdog devices

Red Hat Enterprise Linux 6.0 and above provides two emulated watchdog devices. A watchdog can be used to automatically reboot a guest when it becomes overloaded or unresponsive.

The watchdog devices are supported by the guest operating system Red Hat Enterprise Linux 6.2 and above, for i386 and x86_64 architectures.

You will need to install the **watchdog** package in the guest.

The two devices are:

- `i6300esb`, an emulated Intel 6300 ESB PCI watchdog device. This is the recommended device to use.
- `ib700`, an emulated iBase 700 ISA watchdog device.

Emulated network drivers

There are two emulated network drivers available for network devices:

- The `e1000` driver emulates an Intel E1000 network adapter (Intel 82540EM, 82573L, 82544GC).
- The `rtl8139` driver emulates a Realtek 8139 network adapter.

Emulated storage drivers

Storage devices and storage pools can use these emulated 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 emulated IDE driver

KVM provides two emulated PCI IDE interfaces. An 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. Emulated IDE driver is also used for virtualized CD-ROM and DVD-ROM drives.

The emulated floppy disk drive driver

The emulated floppy disk drive driver is used for creating virtualized floppy drives.

4.3.2. Para-virtualized drivers

Para-virtualized drivers are drivers for virtual devices that increase the I/O performance of virtualized guests.

Para-virtualized drivers decrease I/O latency and increase I/O throughput to near bare-metal levels. It is recommended to use the para-virtualized drivers for virtualized guests running I/O intensive applications.

The para-virtualized drivers must be installed on the guest operating system. By default, the para-virtualized drivers are included in Red Hat Enterprise Linux 4.7 and newer, Red Hat Enterprise Linux 5.4 and newer and Red Hat Enterprise Linux 6.0 and newer. The para-virtualized drivers must be manually installed on Windows guests.



Note

For more information on using the para-virtualized drivers refer to the *Red Hat Enterprise Linux 6 Virtualization Host Configuration and Guest Installation Guide*.

Para-virtualized network driver (`virtio-net`)

The para-virtualized network driver is a Red Hat branded virtual network device. It can be used as the driver for existing network devices or new network devices for virtualized guests.

Para-virtualized block driver (`virtio-blk`)

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).

The para-virtualized clock

Guests using the Time Stamp Counter (TSC) as a clock source may suffer timing issues.

KVM works around hosts that do not have a constant Time Stamp Counter by providing guests with a para-virtualized clock.

The para-virtualized serial driver (virtio-serial)

The para-virtualized serial driver is a bytestream-oriented, character stream driver, and provides a simple communication interface between the host's user space and the guest's user space.

The balloon driver (virtio-balloon)

The balloon driver can designate part of the guest's RAM as not being in use (a process known as balloon *inflation*) so that the memory can be freed for the host (or for other guests on that host) to use. When the guest needs the memory again, the balloon can be *deflated* and the host can distribute the RAM back to the guest.

4.3.3. Physical host devices

Certain hardware platforms allow virtualized guests to directly access various hardware devices and components. This process in virtualization is known as *device assignment*. Device assignment is also known as *passthrough*.

PCI device assignment

The KVM hypervisor supports attaching PCI devices on the host system to virtualized guests. PCI device assignment allows guests to have exclusive access to PCI devices for a range of tasks. It allows PCI devices to appear and behave as if they were physically attached to the guest operating system.

Device assignment is supported on PCI Express devices, with the exception of graphics cards. Parallel PCI devices may be supported as assigned devices, but they have severe limitations due to security and system configuration conflicts.



Note

For more information on Device assignment, refer to the *Red Hat Enterprise Linux 6 Virtualization Host Configuration and Guest Installation Guide*.

USB passthrough

The KVM hypervisor supports attaching USB devices on the host system to virtualized guests. USB device assignment allows guests to have exclusive access to USB devices for a range of tasks. It allows USB devices to appear and behave as if they were physically attached to the guest operating system.



Note

For more information on USB passthrough, refer to the *Red Hat Enterprise Linux 6 Virtualization Administration Guide*.

SR-IOV

SR-IOV (Single Root I/O Virtualization) is a PCI Express standard that extends a single physical PCI function to share its PCI resources as separate, virtual functions (VFs). Each function is capable of being used by a different guest via PCI device assignment.

An SR-IOV capable PCI-e device provides a Single Root Function (for example, a single Ethernet port), and presents multiple, separate virtual devices as separate, unique PCI device functions, each with its own unique PCI configuration space, memory-mapped registers and separate (MSI-based) interrupts.



Note

For more information on SR-IOV, refer to the *Red Hat Enterprise Linux 6 Virtualization Host Configuration and Guest Installation Guide*.

NPIV

N_Port ID Virtualization (NPIV) is a function available with some Fibre Channel devices. NPIV shares a single physical N_Port as multiple N_Port IDs. NPIV provides similar functionality for Fibre Channel Host Bus Adaptors (HBAs) that SR-IOV provides for PCIe interfaces. With NPIV, virtualized guests can be provided with a virtual Fibre Channel initiator to Storage Area Networks (SANs).

NPIV can provide high density virtualized environments with enterprise-level storage solutions.



Note

For more information on NPIV, refer to the *Red Hat Enterprise Linux 6 Virtualization Administration Guide*.

4.3.4. Guest CPU models

Historically, CPU model definitions were hard-coded in **qemu**. This method of defining CPU models was inflexible, and made it difficult to create virtual CPUs with feature sets that matched existing physical CPUs. Typically, users modified a basic CPU model definition with feature flags in order to provide the CPU characteristics required by a guest. Unless these feature sets were carefully controlled, safe migration — which requires feature sets between current and prospective hosts to match — was difficult to support.

qemu-kvm has now replaced most hard-wired definitions with configuration file based CPU model definitions. Definitions for a number of current processor models are now included by default, allowing users to specify features more accurately and migrate more safely.

Chapter 4. Introduction to Red Hat virtualization products

A list of supported CPU models can be viewed with the `/usr/libexec/qemu-kvm -cpu ?model` command. This command outputs the *name* used to select the CPU model at the command line, and a model identifier that corresponds to a commercial instance of that processor class. The CPU models that Red Hat Enterprise Linux supports can be found in the *qemu-kvm Whitelist* chapter in the *Virtualization Administration Guide*.

Configuration details for all of these CPU models can be output with the `/usr/libexec/qemu-kvm -cpu ?dump` command, but they are also stored in the `/usr/share/qemu-kvm/cpu-model/cpu-x86_64.conf` file by default. Each CPU model definition begins with `[cpudev]`, like so:

```
[cpudev]
name = "Nehalem"
level = "2"
vendor = "GenuineIntel"
family = "6"
model = "2"
stepping = "3"
feature_edx = "sse2 sse fxsr mmx pat cmov pge sep apic cx8 mce \
              pae msr tsc pse de fpu mtrr clflush mca pse36"
feature_ecx = "sse3 ssse3"
extfeature_edx = "fxsr mmx pat cmov pge apic cx8 mce pae msr \
                tsc pse de fpu lm syscall nx"
extfeature_ecx = "lahf_lm"
xlevel = "0x8000000A"
model_id = "Intel Celeron_4x0 (Nehalem/Merom Class Core 2)"
```

The four CPUID fields, `feature_edx`, `feature_ecx`, `extfeature_edx` and `extfeature_ecx`, accept named flag values from the corresponding feature sets listed by the `/usr/libexec/qemu-kvm -cpu ?cpuid` command:

```
# qemu-kvm -cpu ?cpuid
Recognized CPUID flags:
 f_edx: pbe ia64 tm ht ss sse2 sse fxsr mmx acpi ds clflush pn \
        pse36 pat cmov mca pge mtrr sep apic cx8 mce pae msr tsc \
        pse de vme fpu
 f_ecx: hypervisor avx osxsave xsave aes popcnt movbe x2apic \
        sse4.2|sse4_2 sse4.1|sse4_1 dca pdcm xtpr cx16 fma cid \
        ssse3 tm2 est smx vmx ds_cpl monitor dtes64 pclmuldq \
        pni|sse3
 extf_edx: 3dnow 3dnowext lm rdtscp pdpe1gb fxsr_opt fxsr mmx \
          mmxext nx pse36 pat cmov mca pge mtrr syscall apic cx8 \
          mce pae msr tsc pse de vme fpu
 extf_ecx: nodeid_msr cvt16 fma4 wdt skinit xop ibs osw \
          3dnowprefetch misalignsse sse4a abm cr8legacy extapic svm \
          cmp_legacy lahf_lm
```

These feature sets are described in greater detail in the appropriate Intel and AMD specifications.

It is important to use the **check** flag to verify that all configured features are available.

```
# /usr/libexec/qemu-kvm -cpu Nehalem,check
warning: host cpuid 0000_0001 lacks requested flag 'sse4.2|sse4_2' [0x00100000]
warning: host cpuid 0000_0001 lacks requested flag 'popcnt' [0x00800000]
```

If a defined feature is not available, those features will fail silently by default.

4.4. Storage

Storage for virtualized guests is abstracted from the physical storage used by the guest. It is attached to virtualized guests using the para-virtualized or emulated block device drivers.

4.4.1. Storage pools

A *storage pool* is a file, directory, or storage device managed by libvirt for the purpose of providing storage to virtualized guests. Storage pools are divided into storage *volumes* that store virtualized guest images or are attached to virtualized guests as additional storage.

Local storage pools

Local storage pools are directly attached to the host server. They include local directories, directly attached disks, physical partitions, and LVM volume groups on local devices. Local storage pools are useful for development, testing and small deployments that do not require migration or large numbers of virtualized guests. Local storage pools may not be suitable for many production environments as they do not support live migration.

Networked (shared) storage pools

Networked storage pools are storage devices shared over a network using standard protocols. Networked storage is required for migrating virtualized guests between hosts. Networked storage pools are managed by libvirt.

Storage Volumes

Storage pools are further divided into storage volumes. Storage volumes are an abstraction of physical partitions, LVM logical volumes, file-based disk images and other storage types handled by libvirt. Storage volumes are presented to virtualized guests as local storage devices regardless of the underlying hardware.



Note

For more information on storage and virtualization refer to the *Red Hat Enterprise Linux 6 Virtualization Administration Guide*.

Virtualization Tools

Read this chapter for an introduction to the many tools available to assist with virtualization.

5.1. virsh

This tool is the main management interface for `virsh` guest domains. It can be used to create, pause and shut down domains, as well as list current domains. This tool is installed as part of the `libvirt-client` package.

Refer to the *Red Hat Enterprise Linux Virtualization Administration Guide* for more information about managing guests with `virsh`.

5.2. virt-manager

A graphical tool for managing virtual machines. It provides the ability to control the life cycle of existing machines, provision new machines, manage virtual networks, access the graphical console of virtual machines, and view performance statistics. This tool ships in its own package: `virt-manager`.

Refer to the *Red Hat Enterprise Linux Virtualization Administration Guide* for more information about managing guests with `virt-manager`.

5.3. virt-install

A command line tool to provision new virtual machines. It supports both text-based and graphical installations, using serial console, SDL, SPICE, or VNC client/server pair graphics. Installation media can be local, or exist remotely on an NFS, HTTP, or FTP server. The tool can also be configured to run unattended and kickstart the guest when installation is complete, allowing for easy automation of installation. This tool is installed as part of the `python-virtinst` package.

Refer to the *Red Hat Enterprise Linux Virtualization Host Configuration and Guest Installation Guide* for more information about `virt-install`.

5.4. guestfish

A command line tool for examining and modifying the file systems of virtual machines. This tool uses `libguestfs`, and exposes all functionality provided by the `guestfs` API. This tool ships in its own package: `guestfish`.

Refer to the *Red Hat Enterprise Linux Virtualization Administration Guide* for more information about `guestfish`.



Warning

Using `guestfish` on running virtual machines can cause disk corruption in the virtual machine. Use `guestfish` with the `--ro` (read-only) option if the disk image or virtual machine may be running.

5.5. Other useful tools

guestmount

A command line tool used to mount virtual machine file systems and disk images on the host machine. This tool is installed as part of the *libguestfs-mount* package.



Warning

Using **guestmount** on running virtual machines can cause disk corruption in the virtual machine. Do not use **guestmount** on a running virtual machine.

virt-cat

A command line tool that can be used to quickly view the contents of one or more files in a specified virtual machine or disk image. This tool is installed as part of the *libguestfs-tools* package.

virt-df

A command line tool used to show the actual physical disk usage of virtual guests. Similar to the command line tool **df**. Note that this tool does not work across remote connections. It is installed as part of the *libguestfs-tools* package.

virt-edit

A command line tool used to edit files that exist on a specified virtual machine. This tool is installed as part of the *libguestfs-tools* package.



Warning

Using **virt-edit** on running virtual machines can cause disk corruption in the virtual machine. **virt-edit** attempts to prevent users from editing files on running virtual machines, but cannot catch all cases. Do not use **virt-edit** on a running virtual machine.

virt-filesystems

A command line tool used to discover file systems, partitions, logical volumes and their sizes in a disk image or virtual machine. One common use is in shell scripts, to iterate over all file systems in a disk image. This tool is installed as part of the *libguestfs-tools* package.



Note

This tool replaces **virt-list-filesystems** and **virt-list-partitions**.

virt-inspector

A command line tool that can examine a virtual machine or disk image to determine the version of its operating system and other information. It can also produce XML output, which can be piped into other programs. Note that **virt-inspector** can only inspect one domain at a time. This tool

virt-inspector2

An alternative tool to **virt-inspector**, written in C. This tool is installed as part of the *libguestfs-tools* package.

virt-ls

A command line tool that lists files and directories inside a virtual machine. This tool is installed as part of the *libguestfs-tools* package.

virt-make-fs

A command line tool for creating a file system based on a tar archive or files in a directory. It is similar to tools like **mkisofs** and **mk squashfs**, but it can create common file system types such as ext2, ext3 and NTFS, and the size of the file system created can be equal to or greater than the size of the files it is based on. This tool is provided as part of the *libguestfs-tools* package.

virt-rescue

A command line tool that provides a rescue shell and some simple recovery tools for unbootable virtual machines and disk images. It can be run on any virtual machine known to libvirt, or directly on disk images. This tool is installed as part of the *libguestfs-tools* package.



Warning

Using **virt-rescue** on running virtual machines can cause disk corruption in the virtual machine. **virt-rescue** attempts to prevent its own use on running virtual machines, but cannot catch all cases.

Using the command with the **--ro** (read-only) option will not cause disk corruption, but may give strange or inconsistent results. It is better to avoid using **virt-rescue** on a running virtual machine.

virt-resize

A command line tool to resize virtual machine disks, and resize or delete any partitions on a virtual machine disk. It works by copying the guest image and leaving the original disk image untouched. This tool is installed as part of the *libguestfs-tools* package.



Important

Using **virt-resize** on running virtual machines can give inconsistent results. It is best to shut down virtual machines before attempting to resize them.

virt-tar

A command line archive tool for downloading and uploading parts of a guest file system. This tool is commonly used for making backups, uploading data, reviewing guest activity, and fixing or customizing guests. This tool is installed as part of the *libguestfs-tools* package.



Warning

Using **virt-tar** with the **-u** (upload) option on running virtual machines can cause disk corruption in the virtual machine. **virt-tar** attempts to prevent its own use on running virtual machines, but cannot catch all cases.

Using **virt-tar** with the **-x** (extract) option on running virtual machines will not cause disk corruption, but may give strange or inconsistent results. It is best to shut down virtual machines before attempting to extract files from them.

virt-top

A command line utility similar to **top**, which shows stats related to virtualized domains. This tool ships in its own package: *virt-top*.

virt-v2v

A graphical tool to convert virtual machines from Xen and VMware hypervisors to run on KVM. This tool ships in its own package: *virt-v2v*.

virt-viewer

A minimal tool for displaying the graphical console of a virtual machine via the VNC and SPICE protocols. This tool ships in its own package: *virt-viewer*.

virt-what

A shell script that detects whether a program is running in a virtual machine. This tool ships in its own package: *virt-what*.

virt-who

The *virt-who* package is a Red Hat Enterprise Linux host agent that queries libvirt for guest UUIDs. It then passes that data to the local entitlement server for the purposes of issuing certificates. This tool ships in its own package: *virt-who*.

virt-win-reg

A command line tool to export and merge Windows Registry entries from a Windows guest, and perform simple Registry operations. This tool is installed as part of the *libguestfs-tools* package.



Warning

Using **virt-win-reg** on running virtual machines will cause irreversible disk corruption in the virtual machine. **virt-win-reg** attempts to prevent its own use on running virtual machines, but cannot catch all cases.



Warning

Modifying the Windows Registry is an inherently risky operation, as the format is deliberately obscure and undocumented. Changes to the registry can leave the system unbootable, so ensure you have a reliable backup before you use the **--merge** option.

virt-xml-validate

A command line tool to validate libvirt XML files for compliance with the published schema. This tool is installed as part of the *libvirt-client* package.

Appendix A. Revision History

Revision **Mon 02 December 2011** Jacquelynn East jeast@redhat.com
0.2-53

GA release for Red Hat Enterprise Linux 6.2.

Revision **Fri 14 October 2011** Jacquelynn East jeast@redhat.com
0.2-47

BZ#744156 add paragraph about emulated watchdogs.

Revision **Sun 18 September 2011** Scott Radvan sradvan@redhat.com
0.2-45

Minor wording issues.

Revision **Fri 16 September 2011** Jacquelynn East jeast@redhat.com
0.2-44

BZ#734614

Revision **Fri 16 September 2011** Jacquelynn East jeast@redhat.com
0.2-43

BZ#734618 minor edit

Revision **Fri 2 September 2011** Jacquelynn East jeast@redhat.com
0.2-37

BZ#734619, BZ#734614

Revision **Thu 1 September 2011** Jacquelynn East jeast@redhat.com
0.2-34

BZ#734619, BZ#734511, BZ#734618, BZ#734616, BZ#715476, BZ#734613

Revision **Wed 31 August 2011** Jacquelynn East jeast@redhat.com
0.2-33

BZ#734618, BZ#734613, BZ#734619

Revision **Thu 25 August 2011** Scott Radvan sradvan@redhat.com
0.2-34

6.2 development.

Revision **Fri July 29 2011** Jacquelynn East jeast@redhat.com
0.2-24

Appendix A. Revision History

Extensive edits, combine security section into advantages

Revision **Wed July 27 2011** **Jacquelynn East** jeast@redhat.com
0.2-22

Advantages chapter completed BZ#715476

Revision **Tue July 26 2011** **Jacquelynn East** jeast@redhat.com
0.2-20

More of the Advantages draft

Revision **Mon July 25 2011** **Jacquelynn East** jeast@redhat.com
0.2-17

Minor edits for BZ#715473 and BZ#715474

Revision **Mon July 25 2011** **Jacquelynn East** jeast@redhat.com
0.2-15

Chapter 4 draft BZ#715476

Revision 0.2-4 **Thu June 23 2011** **Jacquelynn East** jeast@redhat.com
Completed chapter 1

Revision **Wed May 4 2011** **Scott Radvan** sradvan@redhat.com
0.1-01

Arrange basic layout and book infrastructure. Import introductory text.

Revision **Wed May 4 2011** **Scott Radvan** sradvan@redhat.com
0.0-01

Initial creation of book by Publican.