What is Fedora?

Fedora® is a Linux based operating system that provides users with access to the latest free and open source software, in a stable, secure and easy to manage form. Fedora is the largest of many free software creations of the Fedora Project, a partnership of free software community members from around the globe. Because of its predominance, the word "Fedora" is often used interchangeably to mean both the Fedora Project and the Fedora operating system.

Fedora Project
www.fedoraproject.org

Fedora Method

Fedora is a center for innovation in free and open source software, and creates a community where contributors of all kinds -- developers, documenters, artists, system administrators, and other free software and open source enthusiasts -- come together to advance the ecosystem for the benefit of everybody. The Fedora community contributes everything it builds back to the free and open source world and continues to make advances of significance to the broader community, as evidenced by the regular and rapid incorporation of its features into other Linux distributions. Regardless of which Linux distribution you use, you are relying on code developed within the Fedora Project.

Fultus™

Published by Fultus Corporation
www.fultus.com

Fedora Project • Fedora 12 Virtualization Guide

Fedora Project
www.fedoraproject.org

fedora

Fedora 12

Virtualization Guide

By Fedora Project Documentation Team

Fedora 12

Virtualization Guide

The definitive guide for virtualization on Fedora

Fultus™ Books

Fedora 12
Virtualization Guide

ISBN-10: 1-59682-183-3
ISBN-13: 978-1-59682-183-5

Cover design and book layout by Fultus Corporation

Published by Fultus Corporation

Publisher Web: *www.fultus.com*
Linbrary - Linux Library: *www.linbrary.com*
Online Bookstore: *store.fultus.com*
email: *production@fultus.com*

Fediora Release	Book Name	Edition
F12	**Virtualization Guide**	**1**

Authors	Emails
Christoph Curran	*ccurran@redhat.com*

Table of Contents

List of Tables

Legal Notice

Abstract

The Fedora 12 Virtualization Guide contains information on installation, configuring, administering, tips, tricks and troubleshooting virtualization technologies used in Fedora 12.

Preface

This book is the Fedora 12 Virtualization Guide. The Guide covers all aspects of using and managing virtualization on Fedora 12.

1. About this book

This book is divided into 7 parts:

- System Requirements
- Installation
- Configuration
- Administration
- Reference
- Tips and Tricks
- Troubleshooting

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.

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 key caps and key-combinations. For example:

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

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 key cap, all presented in mono-spaced bold and all distinguishable thanks to context.

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

Press **Enter** to execute the command.

Press **Ctrl+Alt+F1** to switch to the first virtual terminal. Press **Ctrl+Alt+F7** 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 m`ono-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 **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.

Note the **>** shorthand used to indicate traversal through a menu and its sub-menus. This avoids difficult-to-follow phrasing such as 'Select **Mouse** from the **Preferences** sub-menu in the **System** menu of the main menu bar'.

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:

> When the Apache HTTP Server accepts requests, it dispatches child processes or threads to handle them. This group of child processes or threads is known as a *server-pool*. Under Apache HTTP Server 2.0, the responsibility for creating and maintaining these server-pools has been abstracted to a group of modules called *Multi-Processing Modules* (*MPMs*). Unlike other modules, only one module from the MPM group can be loaded by the Apache HTTP Server.

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' won't cause data loss but may cause irritation and frustration.

Warning

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

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/bugzilla/* against the product **Fedora Documentation.**

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

Part I.
Installation

Virtualization installation topics

These chapters describe setting up the host and installing virtualized guests with Fedora. It is recommended to read these chapters carefully to ensure successful installation of virtualized guest operating systems.

Chapter 1.
Installing the virtualization packages

1.1. Installing KVM with a new Fedora installation

This section covers installing virtualization tools and KVM package as part of a fresh Fedora 12 installation.

Need help installing?

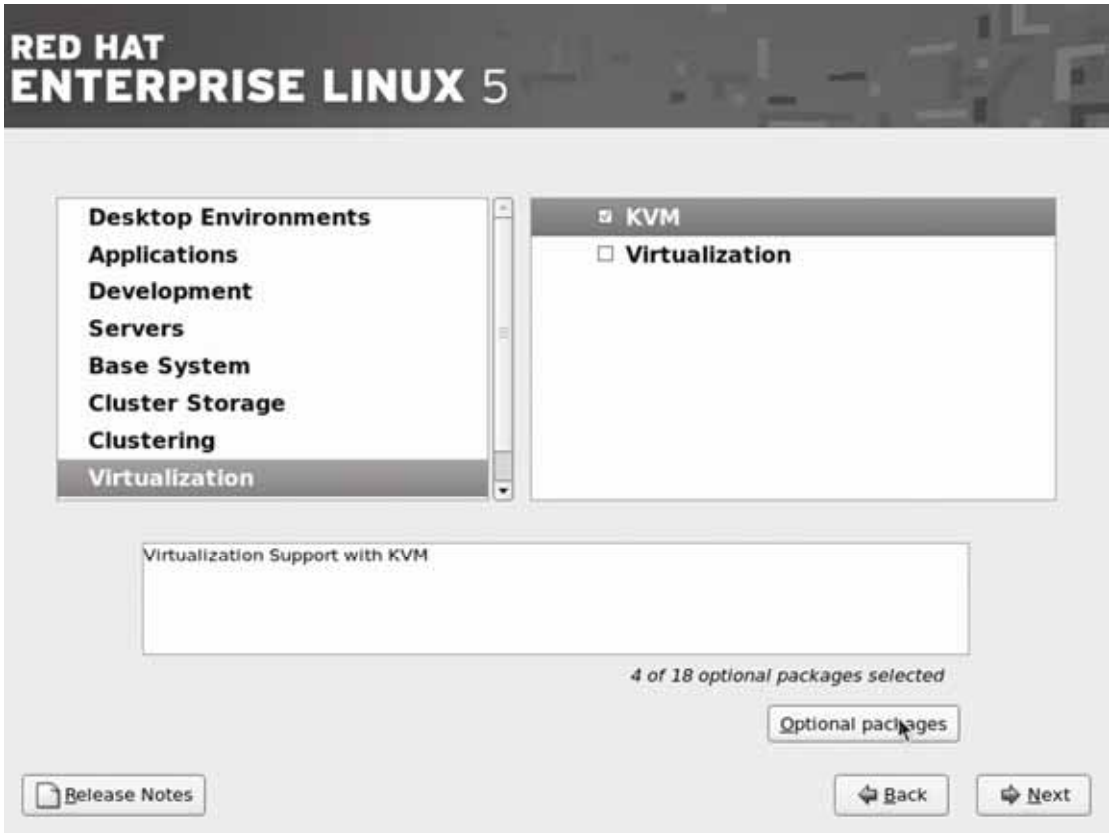
The *Fedora 12 Installation Guide* (available from *http://docs.fedoraproject.org*) covers installing Fedora 12 in detail.

1. Start an interactive Fedora installation from the Fedora 12 Installation CD-ROM, DVD or PXE.

2. Complete the other steps up to the package selection step.

Select the **Virtualization** package group and the **Customize Now** radio button.

3. Select the **KVM** package group. Deselect the **Virtualization** package group. This selects the KVM hypervisor, `virt-manager`, `libvirt` and `virt-viewer` for installation.



4. Customize the packages (if required)

   Customize the **Virtualization** group if you require other virtualization packages.

   Press **Close** followed by **Next** to continue the installation.

Installing KVM packages with Kickstart files

This section describes how to use a Kickstart file to install Fedora with the KVM hypervisor packages. Kickstart files allow for large, automated installations without a user manually installing each individual system. The steps in this section will assist you in creating and using a Kickstart file to install Fedora with the virtualization packages.

In the `%packages` section of your Kickstart file, append the following package group:

```
%packages
@kvm
```

More information on Kickstart files can be found on the Fedora Project website, *http://docs.fedoraproject.org*, in the *Fedora 12 Installation Guide*.

1.2. Installing KVM packages on an existing Fedora system

The section describes the steps for installing the KVM hypervisor on a working Fedora 12 or newer.

Installing the KVM hypervisor with yum

To use virtualization on Fedora you require the `kvm` package. The `kvm` package contains the KVM kernel module providing the KVM hypervisor on the default Linux kernel.

To install the `kvm` package, run:

```
# yum install kvm
```

Now, install additional virtualization management packages.

Recommended virtualization packages:

`python-virtinst`

Provides the `virt-install` command for creating virtual machines.

`libvirt`

`libvirt` is an API library for interacting with hypervisors. `libvirt` uses the `xm` virtualization framework and the `virsh` command line tool to manage and control virtual machines.

`libvirt-python`

The libvirt-python package contains a module that permits applications written in the Python programming language to use the interface supplied by the `libvirt` API.

`virt-manager`

`virt-manager`, also known as **Virtual Machine Manager**, provides a graphical tool for administering virtual machines. It uses `libvirt` library as the management API.

Install the other recommended virtualization packages:

```
# yum install virt-manager libvirt libvirt-python python-virtinst
```

Chapter 2.
Virtualized guest installation overview

After you have installed the virtualization packages on the host system you can create guest operating systems. This chapter describes the general processes for installing guest operating systems on virtual machines. You can create guests using the **New** button in **virt-manager** or use the command line interface `virt-install`. Both methods are covered by this chapter.

Detailed installation instructions are available for specific versions of Fedora, other Linux distributions, Solaris and Windows. Refer to *Chapter 3, Guest operating system installation procedures* for those procedures.

2.1. Creating guests with virt-install

You can use the `virt-install` command to create virtualized guests from the command line. `virt-install` is used either interactively or as part of a script to automate the creation of virtual machines. Using `virt-install` with Kickstart files allows for unattended installation of virtual machines.

The `virt-install` tool provides a number of options one can pass on the command line. To see a complete list of options run:

```
$ virt-install --help
```

The `virt-install` man page also documents each command option and important variables.

`qemu-img` is a related command which may be used before `virt-install` to configure storage options.

An important option is the *`--vnc`* option which opens a graphical window for the guest's installation.

This example creates a Red Hat Enterprise Linux 3 guest, named *`rhel3support`*, from a CD-ROM, with virtual networking and with a 5 GB file-based block device image. This example uses the KVM hypervisor.

```
# virt-install --accelerate --hvm --connect qemu:///system \
```

```
      --network network:default \
      --name rhel3support --ram=756\
      --file=/var/lib/libvirt/images/rhel3support.img \
      --file-size=6 --vnc --cdrom=/dev/sr0
```

Example 2.1. Using virt-install with KVM to create a Red Hat Enterprise Linux 3 guest

```
# virt-install --name Fedora11 --ram 512 --
file=/var/lib/libvirt/images/Fedora11.img \
      --file-size=3 --vnc --cdrom=/var/lib/libvirt/images/Fedora11.iso
```

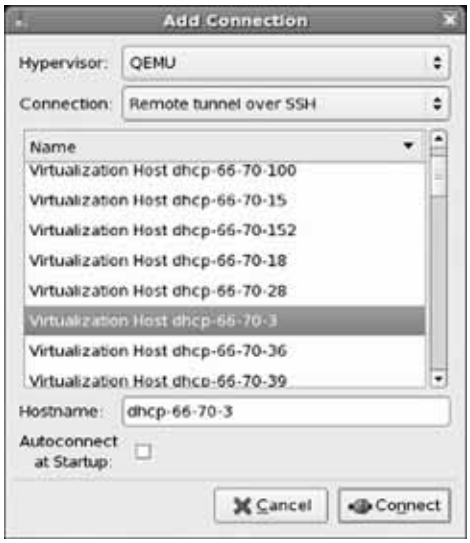
Example 2.2. Using virt-install to create a Fedora 11 guest

2.2. Creating guests with virt-manager

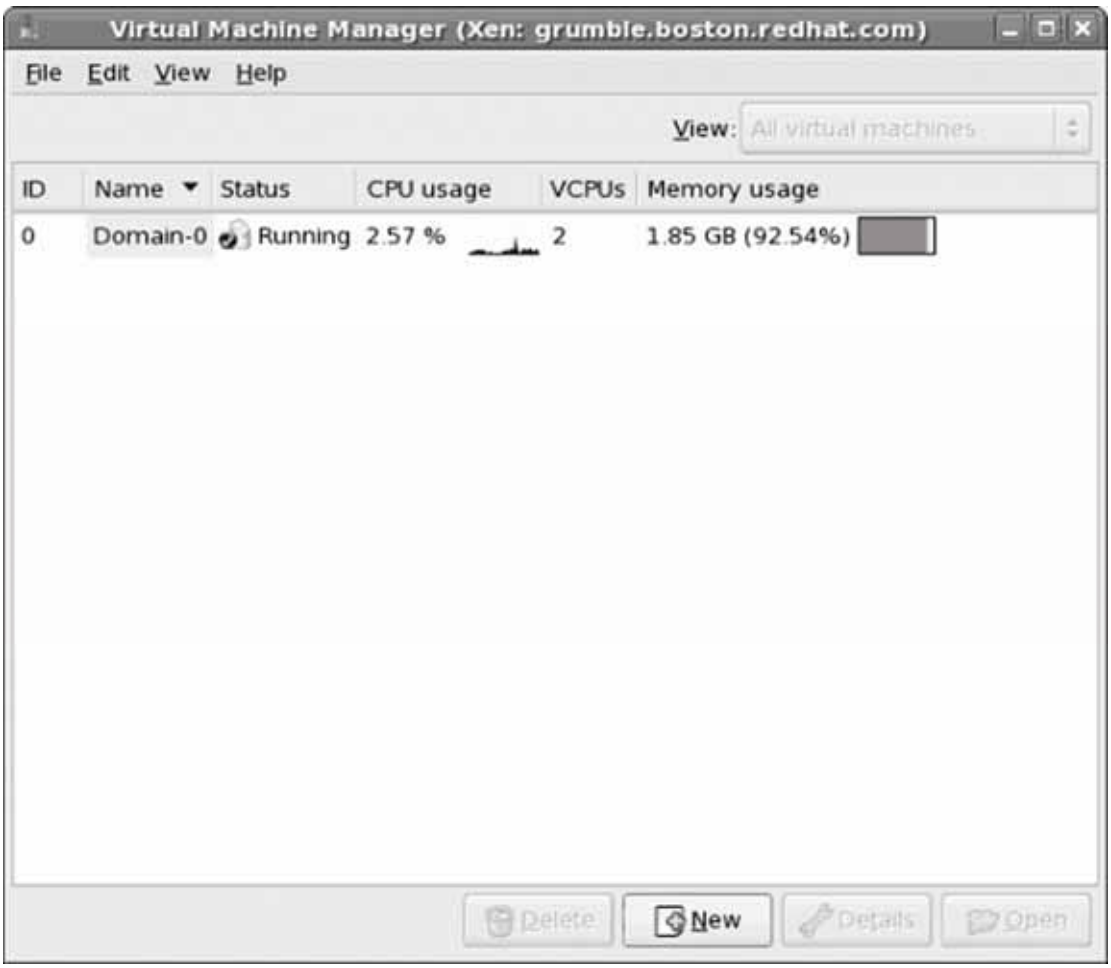
`virt-manager`, also known as Virtual Machine Manager, is a graphical tool for creating and managing virtualized guests.

Procedure 2.1. Creating a virtualized guest with virt-manager

1. To start **virt-manager** run the following command as root:
   ```
   # virt-manager &
   ```
   The `virt-manager` command opens a graphical user interface window. Various functions are not available to users without root privileges or `sudo` configured, including the **New** button and you will not be able to create a new virtualized guest.
2. Open the **File -> Open Connection**. The dialog box below appears. . Select a hypervisor and click the **Connect** button:



3. The **virt-manager** window allows you to create a new virtual machine. Click the **New** button to create a new guest. This opens the wizard shown in the screenshot.



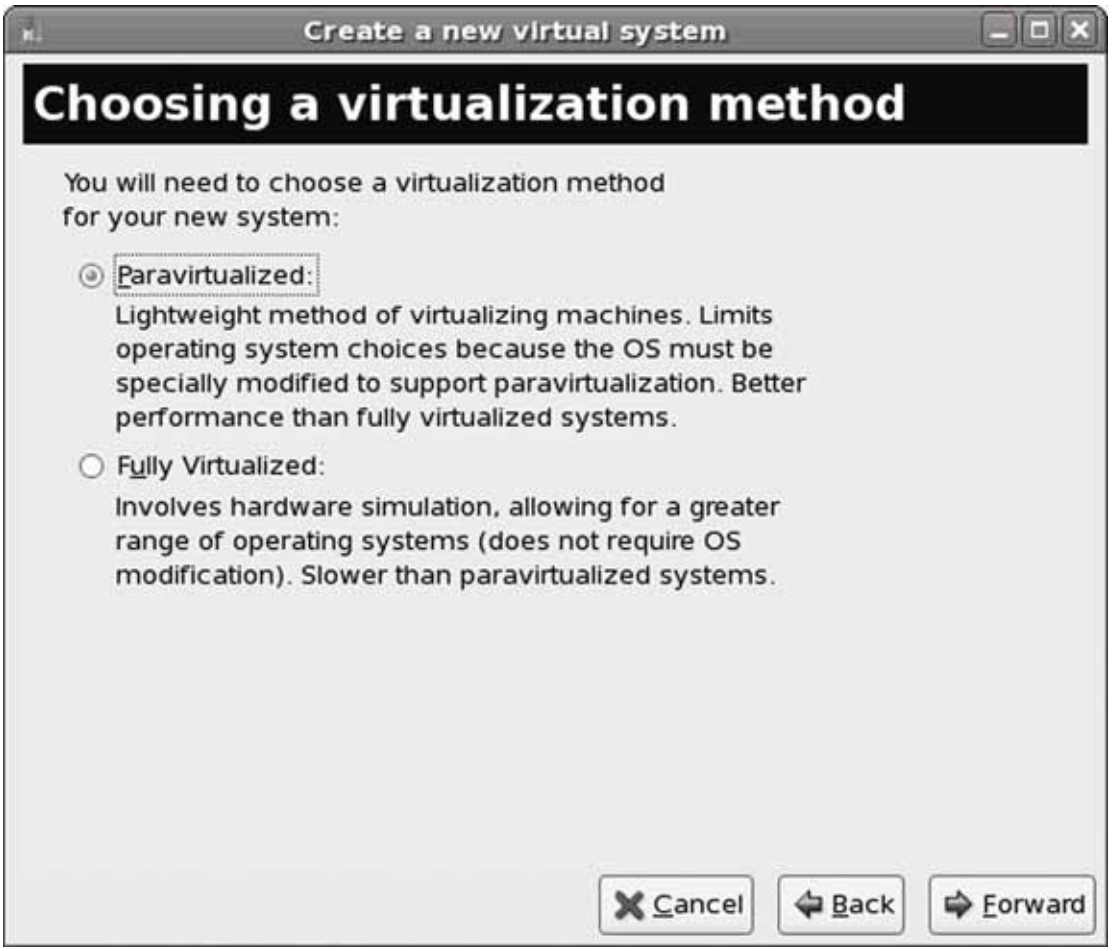
4. The **Create a new virtual system** window provides a summary of the information you must provide in order to create a virtual machine:

Review the information for your installation and click the **Forward** button.

5. The **Choosing a virtualization method** window appears. Choose between **Para-virtualized** or **Fully virtualized**.

   Full virtualization requires a system with Intel® VT or AMD-V processor. If the virtualization extensions are not present the **fully virtualized** radio button or the **Enable kernel/hardware acceleration** will not be selectable. The **Para-virtualized** option will be grayed out if `kernel-xen` is not the kernel running presently.

   If you connected to a KVM hypervisor only full virtualization is available.



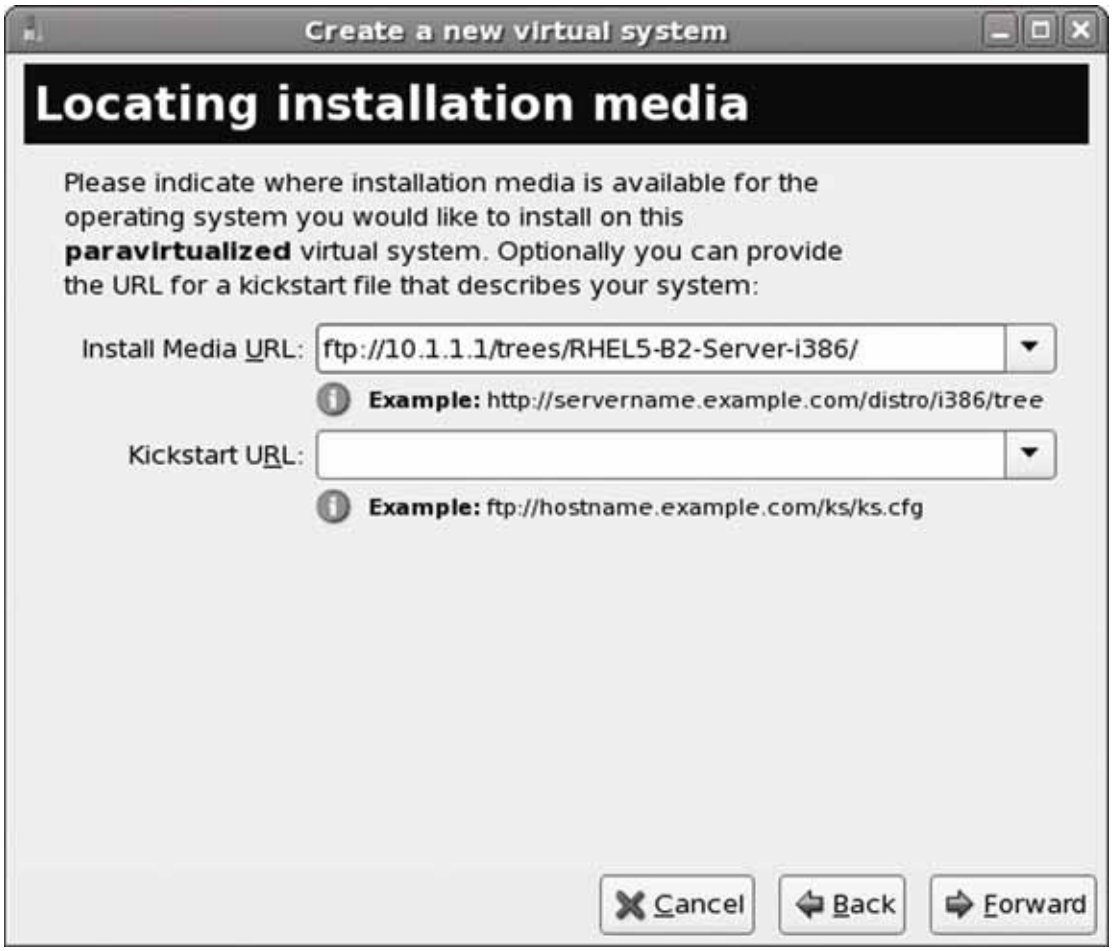
   Choose the virtualization type and click the **Next** button.

6. The **Locating installation media** prompt asks for the installation media for the type of installation you selected. This screen is dependent on what was selected in the previous step.

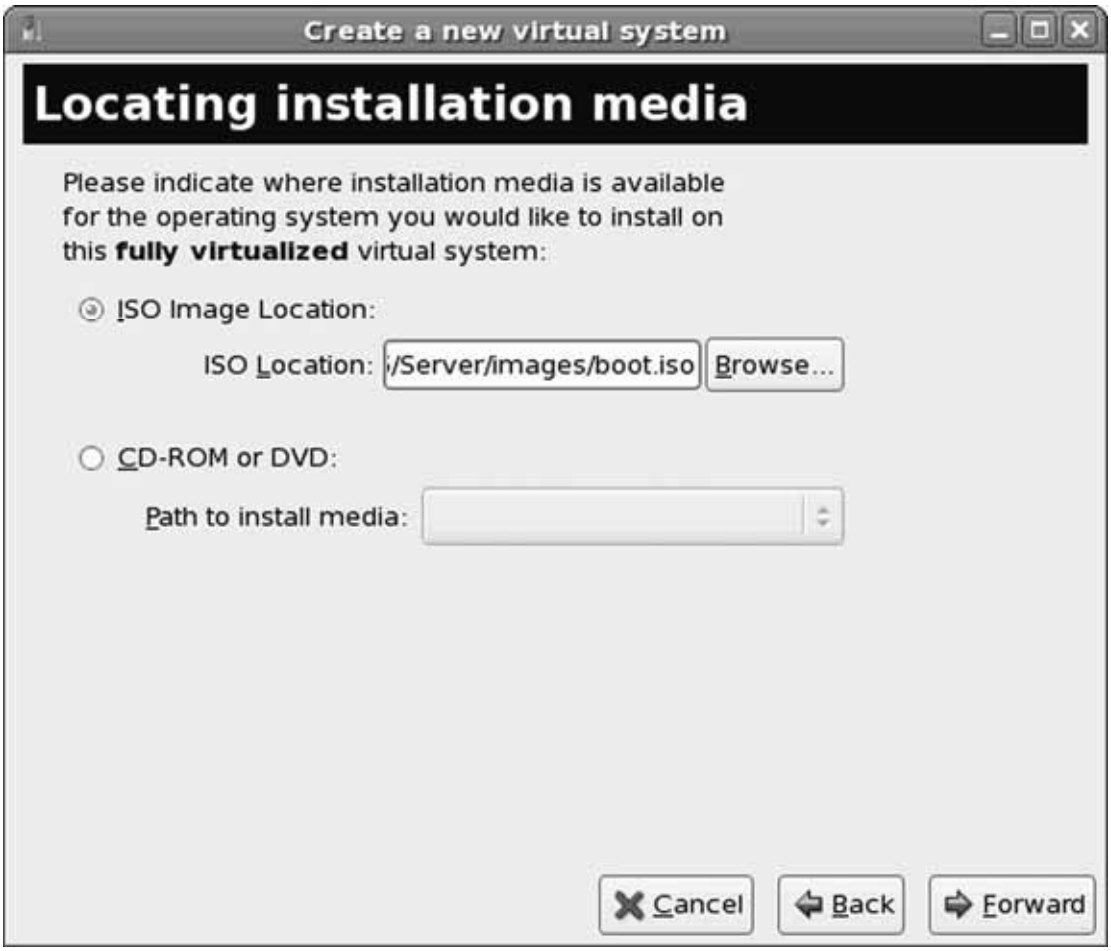
   1. The para-virtualized installation requires an installation tree accessible using one of the following network protocols: `HTTP`, `FTP` or `NFS`. The installation media URL must contain a Fedora installation tree. This tree is hosted using `NFS`, `FTP` or `HTTP`. The network services and files can be hosted using network services on the host or another mirror.

      Using a CD-ROM or DVD image (tagged as an `.iso` file), mount the CD-ROM image and host the mounted files with one of the mentioned protocols.

      Alternatively, copy the installation tree from a Fedora mirror.



2. A fully virtualized guest installation require bootable installation DVDs, CD-ROMs or images of bootable installation DVDs or CD-ROMs (with the .iso or .img file type) locally. Windows installations use DVD, CD-ROM or .iso file. Many Linux and UNIX-like operating systems use an .iso file to install a base system before finishing the installation with a network based installation tree.



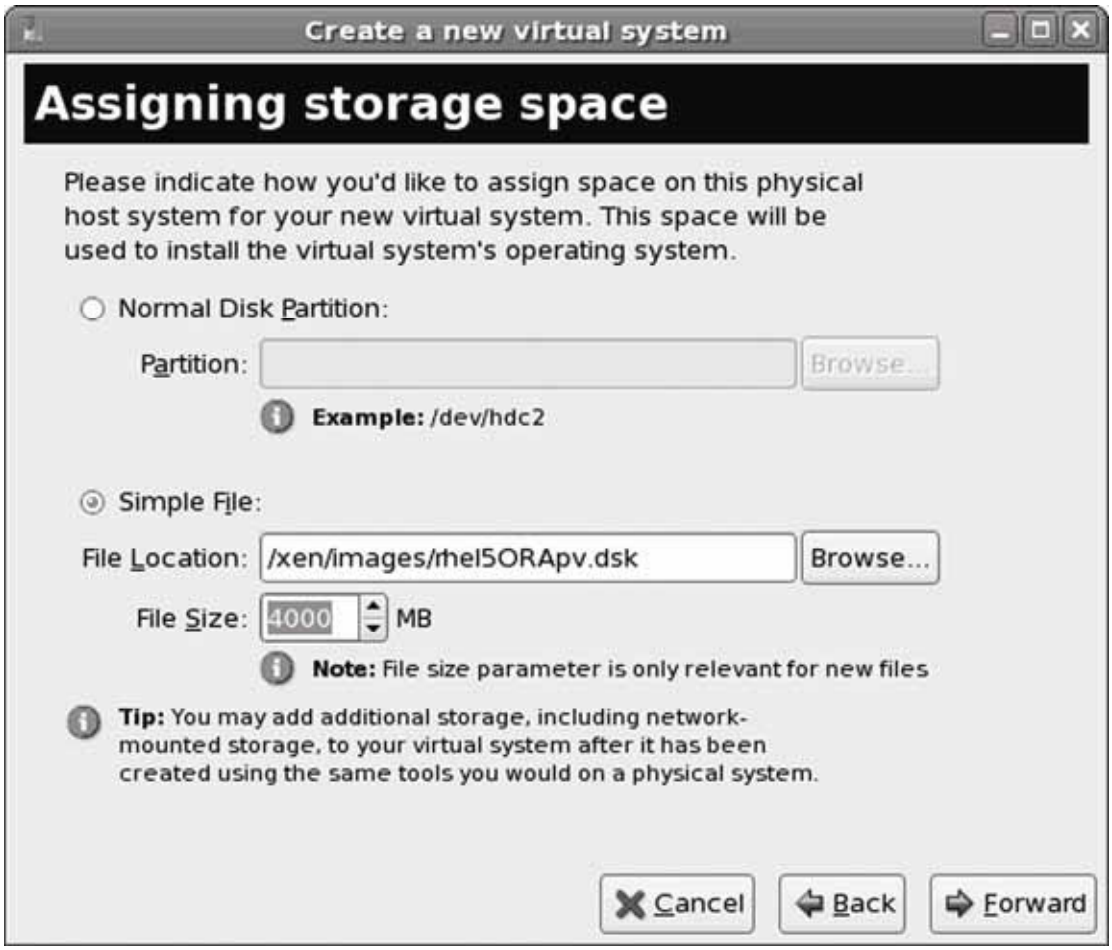
After selecting the appropriate installation media, click the **Forward** button.

7. The **Assigning storage space** window displays. Choose a disk partition, LUN or create a file based image for the guest storage.

   The convention for file based images in Fedora is that all file based guest images are in the `/var/lib/xen/images/` directory. Other directory locations for file based images are prohibited by SELinux. If you run SELinux in enforcing mode, refer to *Section 7.1, "SELinux and virtualization"* for more information on installing guests.

   Your guest storage image should be larger than the size of the installation, any additional packages and applications, and the size of the guests swap file. The installation process will choose the size of the guest's swap file based on size of the RAM allocated to the guest.

   Allocate extra space if the guest needs additional space for applications or other data. For example, web servers require additional space for log files.



Choose the appropriate size for the guest on your selected storage type and click the **Forward** button.

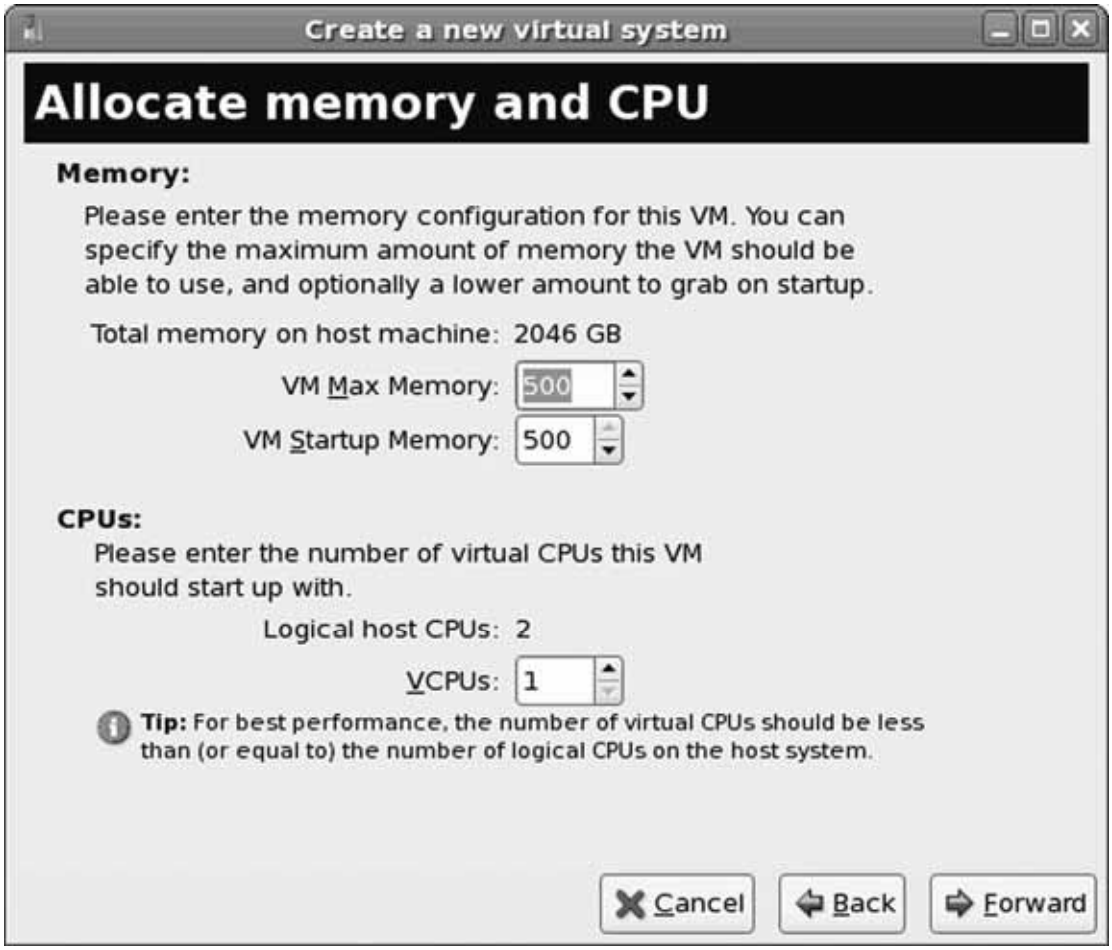
Note

It is recommend that you use the default directory for virtual machine images, `/var/lib/xen/images/`. If you are using a different location (such as `/xen/images/` in this example) make sure it is added to your SELinux policy and relabeled before you continue with the installation (later in the document you will find information on how to modify your SELinux policy).

8. The Allocate memory and CPU window displays. Choose appropriate values for the virtualized CPUs and RAM allocation. These values affect the host's and guest's performance.

   Guests require sufficient physical memory (RAM) to run efficiently and effectively. Choose a memory value which suits your guest operating system and application requirements. Most operating system require at least 512MB of RAM to work responsively. Remember, guests use physical RAM. Running too many guests or leaving insufficient memory for the host system results in significant usage of virtual memory. Virtual memory is significantly slower causing degraded system performance and responsiveness. Ensure to allocate sufficient memory for all guests and the host to operate effectively.

   Assign sufficient virtual CPUs for the virtualized guest. If the guest runs a multithreaded application assign the number of virtualized CPUs it requires to run most efficiently. Do not assign more virtual CPUs than there are physical processors (or hyper-threads) available on the host system. It is possible to over allocate virtual processors, however, over allocating has a significant, negative affect on guest and host performance due to processor context switching overheads.


Create a new virtual system
Allocate memory and CPU
Memory:
Please enter the memory configuration for this VM. You can specify the maximum amount of memory the VM should be able to use, and optionally a lower amount to grab on startup.
Total memory on host machine: 2046 GB
VM Max Memory: 500
VM Startup Memory: 500
CPUs:
Please enter the number of virtual CPUs this VM should start up with.
Logical host CPUs: 2
VCPUs: 1
Tip: For best performance, the number of virtual CPUs should be less than (or equal to) the number of logical CPUs on the host system.
Cancel
Back
Forward

9. The ready to begin installation window presents a summary of all configuration information you entered. Review the information presented and use the **Back** button to make changes, if necessary. Once you are satisfied click the **Finish** button and to start the installation process.

A VNC window opens showing the start of the guest operating system installation process.

This concludes the general process for creating guests with `virt-manager`. *Chapter 3, Guest operating system installation procedures* contains step-by-step instructions to installing a variety of common operating systems.

2.3. Installing guests with PXE

This section covers the steps required to install guests with PXE. PXE guest installation requires a shared network device, also known as a network bridge. The procedures below cover creating a bridge and the steps required to utilize it the bridge for a PXE installation.

1. **Create a new bridge**

   1. Create a new network script file in the `/etc/sysconfig/network-scripts/` directory. This example creates a file named `ifcfg-installation` which makes a bridge named *`installation`*

      ```
      # cd /etc/sysconfig/network-scripts/
      # vim ifcfg-installation
      DEVICE=installation
      TYPE=Bridge
      BOOTPROTO=dhcp
      ONBOOT=yes
      ```

Warning

The line, *`TYPE=Bridge`*, is case-sensitive. It must have uppercase 'B' and lower case 'ridge'.

   2.

      ```
      Start the new bridge.
      # ifup installation
      ```

   3. There are no interfaces added to the new bridge yet. Use the `brctl show` command to view details about network bridges on the system.

      ```
      # brctl show
      bridge name     bridge id               STP enabled     interfaces
      installation    8000.000000000000       no
      virbr0          8000.000000000000       yes
      ```

      The `virbr0` bridge is the default bridge used by `libvirt` for Network Address Translation (NAT) on the default Ethernet device.

2. **Add an interface to the new bridge**

   Edit the configuration file for the interface. Add the `BRIDGE` parameter to the configuration file with the name of the bridge created in the previous steps.

   ```
   # Intel Corporation Gigabit Network Connection
   DEVICE=eth1
   BRIDGE=installation
   BOOTPROTO=dhcp
   HWADDR=00:13:20:F7:6E:8E
   ONBOOT=yes
   ```

After editing the configuration file, restart networking or reboot.

```
# service network restart
```

Verify the interface is attached with the `brctl show` command:

```
# brctl show
bridge name     bridge id               STP enabled     interfaces
installation    8000.001320f76e8e       no              eth1
virbr0          8000.000000000000       yes
```

3. **Security configuration**

   Configure `iptables` to allow all traffic to be forwarded across the bridge.

```
# iptables -I FORWARD -m physdev --physdev-is-bridged -j ACCEPT
# service iptables save
# service iptables restart
```

Disable iptables on bridges

Alternatively, prevent bridged traffic from being processed by `iptables` rules. In `/etc/sysctl.conf` append the following lines:

```
net.bridge.bridge-nf-call-ip6tables = 0
net.bridge.bridge-nf-call-iptables = 0
net.bridge.bridge-nf-call-arptables = 0
```

Reload the kernel parameters configured with `sysctl`

```
# sysctl -p /etc/sysctl.conf
```

4. **Restart libvirt before the installation**

   Restart the `libvirt` daemon.

```
# service libvirtd reload
```

The bridge is configured, you can now begin an installation.

PXE installation with virt-install

For `virt-install` append the `--network=bridge:BRIDGENAME` installation parameter where installation is the name of your bridge. For PXE installations use the *`--pxe`* parameter.

```
# virt-install --accelerate --hvm --connect qemu:///system \
    --network=bridge:installation --pxe\
    --name EL10 --ram=756 \
    --vcpus=4
    --os-type=linux --os-variant=rhel5
    --file=/var/lib/libvirt/images/EL10.img \
```

Example 2.3. PXE installation with virt-install

PXE installation with virt-manager

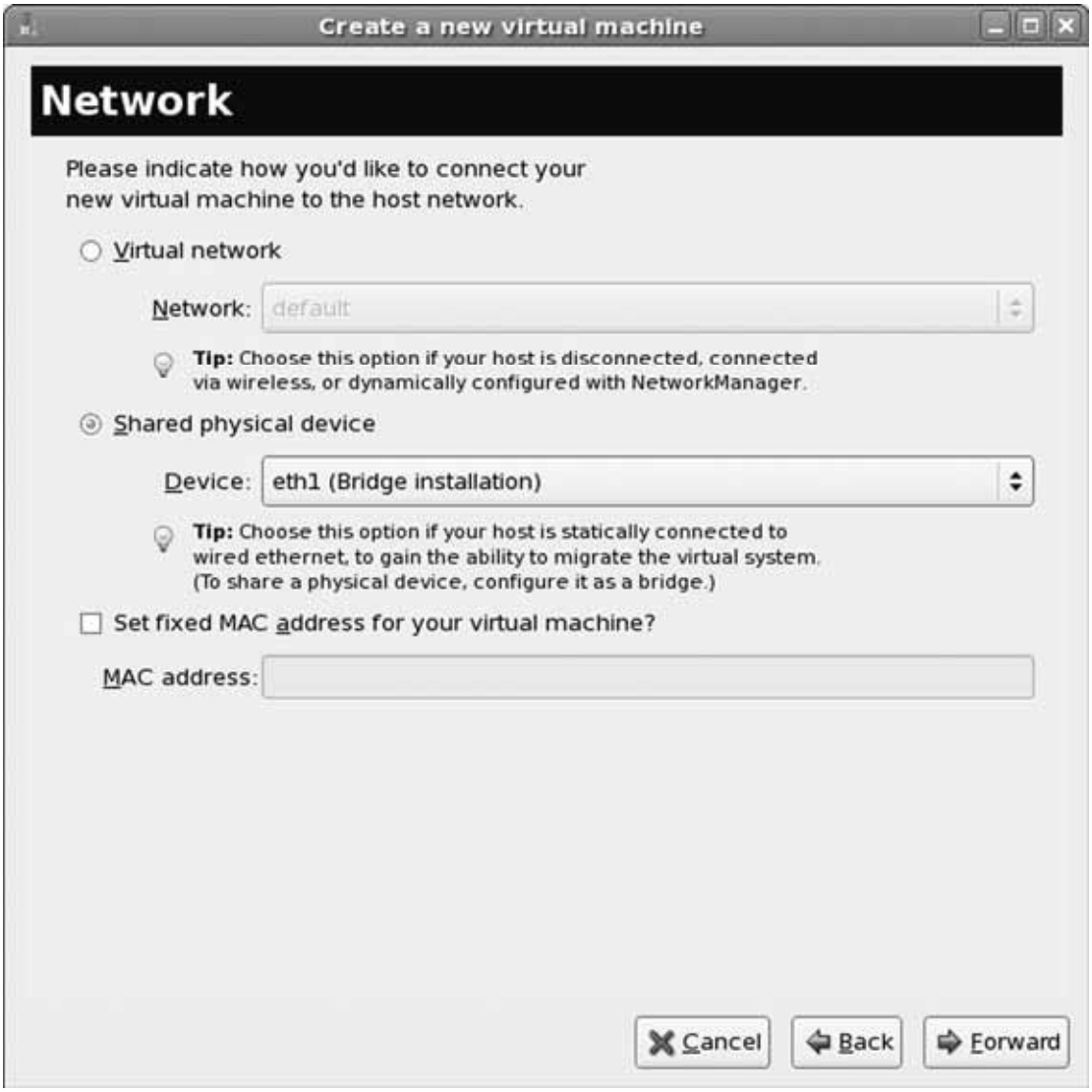
The steps below are the steps that vary from the standard virt-manager installation procedures. For the standard installations refer to *Chapter 3, Guest operating system installation procedures*.

1. **Select PXE**

   Select PXE as the installation method.

2. **Select the bridge**

   Select **Shared physical device** and select the bridge created in the previous procedure.



3. **Start the installation**

   The installation is ready to start.

A DHCP request is sent and if a valid PXE server is found the guest installation processes will start.

Appendix A.
Additional resources

To learn more about virtualization and Linux, refer to the following resources.

A.1. Online resources

- *http://www.cl.cam.ac.uk/research/srg/netos/xen/* The project website of the *Xen*™ para-virtualization machine manager from which the Fedora kernel-xen package is derived. The site maintains the upstream xen project binaries and source code and also contains information, architecture overviews, documentation, and related links regarding xen and its associated technologies.
- The Xen Community website: *http://www.xen.org/*
- *http://www.libvirt.org/* is the official website for the libvirt virtualization API.
- *http://virt-manager.et.redhat.com/* is the project website for the Virtual Machine Manager (virt-manager), the graphical application for managing virtual machines.
- Open Virtualization Center: *http://www.openvirtualization.com*
- Fedora Documentation: *http://docs.fedoraproject.org*
- Virtualization technologies overview: *http://virt.kernelnewbies.org*
- Red Hat Emerging Technologies group: *http://et.redhat.com*

A.2. Installed documentation

- /usr/share/doc/xen-<*version-number*>/ is the directory which contains information about the Xen para-virtualization hypervisor and associated management tools, including various example configurations, hardware-specific information, and the current Xen upstream user documentation.
- man virsh and /usr/share/doc/libvirt-<*version-number*> — Contains sub commands and options for the virsh virtual machine management utility as well as comprehensive information about the libvirt virtualization library API.
- /usr/share/doc/gnome-applet-vm-<*version-number*> — Documentation for the GNOME graphical panel applet that monitors and manages locally-running virtual machines.

- /usr/share/doc/libvirt-python-<*version-number*> — Provides details on the Python bindings for the libvirt library. The libvirt-python package allows python developers to create programs that interface with the libvirt virtualization management library.
- /usr/share/doc/python-virtinst-<*version-number*> — Provides documentation on the virt-install command that helps in starting installations of Fedora and Linux related distributions inside of virtual machines.
- /usr/share/doc/virt-manager-<*version-number*> — Provides documentation on the Virtual Machine Manager, which provides a graphical tool for administering virtual machines.

Appendix B.

Revision History

Revision History	Data	Author
Revision 12.1.3 Split from Red Hat Enterprise Linux 5.4 Virtualization Guide version 5.4-61.	Mon Oct 12 2009	Christopher Curran

Colophon

This manual was written in the DocBook XML v4.3 format.

This book is based on the work of Jan Mark Holzer and Chris Curran.

Other writing credits go to:

- Don Dutile contributed technical editing for the para-virtualized drivers section.
- Barry Donahue contributed technical editing for the para-virtualized drivers section.
- Rick Ring contributed technical editing for the Virtual Machine Manager Section.
- Michael Kearey contributed technical editing for the sections on using XML configuration files with virsh and virtualized floppy drives.
- Marco Grigull contributed technical editing for the software compatibility and performance section.
- Eugene Teo contributed technical editing for the Managing Guests with virsh section.

Publican, the publishing tool which produced this book, was written by Jeffrey Fearn.

The Red Hat Localization Team consists of the following people:

East Asian Languages

- Simplified Chinese
  - Leah Wei Liu
- Traditional Chinese
  - Chester Cheng
  - Terry Chuang
- Japanese
  - Junko Ito
- Korean
  - Eun-ju Kim

Latin Languages

- French
  - Sam Friedmann
- German
  - Hedda Peters
- Italian
  - Francesco Valente
- Brazilian Portuguese
  - Glaucia de Freitas
  - Leticia de Lima
- Spanish
  - Angela Garcia
  - Gladys Guerrero
- Russian
  - Yuliya Poyarkova

Glossary

This glossary is intended to define the terms used in this Installation Guide.

Bare-metal

The term bare-metal refers to the underlying physical architecture of a computer. Running an operating system on bare-metal is another way of referring to running an unmodified version of the operating system on the physical hardware. Examples of operating systems running on bare metal are *dom0* or a normally installed operating system.

dom0

Also known as the *Host* or host operating system.

dom0 refers to the host instance of Linux running the Hypervisor which facilitates virtualization of guest operating systems. Dom0 runs on and manages the physical hardware and resource allocation for itself and the guest operating systems.

Domains

domU and *Domains* are both domains. Domains run on the *Hypervisor*. The term domains has a similar meaning to *Virtual machines* and the two are technically interchangeable. A domain is a Virtual Machine.

domU

domU refers to the guest operating system which run on the host system (*Domains*).

Full virtualization

Xen and KVM can use full virtualization. Full virtualization uses hardware features of the processor to provide total abstraction of the underlying physical system (*Bare-metal*) and create a new virtual system in which the guest operating systems can run. No modifications are needed in the guest operating system. The guest operating system and any applications on the guest are not aware of the virtualized environment and run normally. Para-virtualization requires a modified version of the Linux operating system.

Fully virtualized

See *Full virtualization*.

Guest system

Also known as guests, virtual machines or *domU*.

Hardware Virtual Machine

See *Full virtualization*

Hypervisor

The hypervisor is the software layer that abstracts the hardware from the operating system permitting multiple operating systems to run on the same hardware. The hypervisor runs on a host operating system allowing other virtualized operating systems to run on the host's hardware.

Host

The host operating system, also known as *dom0*.

The host operating system environment runs the virtualization software for *Fully virtualized* and *Para-virtualized* guest systems.

I/O

Short for input/output (pronounced "eye-oh"). The term I/O describes any program, operation or device that transfers data to or from a computer and to or from a peripheral device. Every transfer is an output from one device and an input into another. Devices such as keyboards and mouses are input-only devices while devices such as printers are output-only. A writable CD-ROM is both an input and an output device.

Kernel-based Virtual Machine

KVM (Kernel-based Virtual Machine) is a *Full virtualization* solution for Linux on AMD64 and Intel 64 hardware. VM is a Linux kernel module built for the standard Linux kernel. KVM can run multiple, unmodified virtualized guest Windows and Linux operating systems. KVM is a hypervisor which uses the libvirt virtualization tools (virt-manager and virsh).

KVM is a set of Linux kernel modules which manage devices, memory and management APIs for the Hypervisor module itself. Virtualized guests are run as Linux processes and threads which are controlled by these modules.

LUN

A Logical Unit Number (LUN) is a number assigned to a logical unit (a SCSI protocol entity).

Migration

Migration is name for the process of moving a virtualized guest from one host to another. Migration can be conducted offline (where the guest is suspended and then moved) or live (where a guest is moved without suspending). Xen fully virtualized guests, Xen para-virtualized guest and KVM fully virtualized guests can all be migrated.

Migration is a key feature of virtualization as software is completely separated from hardware. Migration is useful for:

- Load balancing - guests can be moved to hosts with lower usage when a host becomes overloaded.
- Hardware failover - when hardware devices on the host start to fail, guests can be safely relocated so the host can be powered down and repaired.
- Energy saving - guests can be redistributed to other hosts and host systems powered off to save energy and cut costs in low usage periods.
- Geographic migration - guests can be moved to another location for lower latency or in serious circumstances.

Shared, networked storage is used for storing guest images. Without shared storage migration is not possible.

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

The time an offline migration takes depends network bandwidth and latency. A guest with 2GB of memory should take several seconds on a 1 Gbit Ethernet link.

A live migration keeps the guest running on the source host and begins moving the memory without stopping the guest. All modified memory pages are tracked and sent to the destination after the image is sent. The memory is updated with the changed pages. The process continues until it reaches some heuristic; either it successfully copied all the pages over, or the source is changing too fast and the destination host cannot make progress. If the heuristic is met the guest is briefly paused on the source host and the registers and buffers are sent. The registers are loaded on the new host and the guest is then resumed on the destination host. If the guest cannot be merged (which happens when guests are under extreme loads) the guest is paused and then an offline migration is started instead.

The time an offline migration takes depends network bandwidth and latency as well as activity on the guest. If the guest is using significant I/O or CPU the migration will take much longer.

MAC Addresses

The Media Access Control Address is the hardware address for a Network Interface Controller. In the context of virtualization MAC addresses must be generated for virtual network interfaces with each MAC on your local domain being unique.

Para-virtualization

Para-virtualization uses a special kernel, sometimes referred to as the Xen kernel or the kernel-xen package. Para-virtualized guest kernels are run concurrently on the host while using the host's libraries and devices. A para-virtualized installation can have complete access to all devices on the system which can be limited with security settings (SELinux and file controls). Para-virtualization is faster than full virtualization. Para-virtualization can effectively be used for load balancing, provisioning, security and consolidation advantages.

As of Fedora 9 a special kernel will no longer be needed. Once this patch is accepted into the main Linux tree all Linux kernels after that version will have para-virtualization enabled or available.

Para-virtualized

See *Para-virtualization*,

Para-virtualized drivers

Para-virtualized drivers are device drivers that operate on fully virtualized Linux guests. These drivers greatly increase performance of network and block device I/O for fully virtualized guests.

Security Enhanced Linux

Short for Security Enhanced Linux, SELinux uses Linux Security Modules (LSM) in the Linux kernel to provide a range of minimum privilege required security policies.

Universally Unique Identifier

A Universally Unique Identifier (UUID) is a standardized numbering method for devices, systems and certain software objects in distributed computing environments. Types of UUIDs in virtualization include: ext2 and ext3 file system identifiers, RAID device identifiers, iSCSI and LUN device identifiers, MAC addresses and virtual machine identifiers.

Virtualization

Virtualization is a board computing term for running software, usually operating systems, concurrently and isolated from other programs on one system. Most existing implementations of virtualization use a hypervisor, a software layer on top

of an operating system, to abstract hardware. The hypervisor allows multiple operating systems to run on the same physical system by giving the guest operating system virtualized hardware. There are various methods for virtualizing operating systems:

- Hardware-assisted virtualization is the technique used for full virtualization with Xen and KVM (definition: *Full virtualization*)
- Para-virtualization is a technique used by Xen to run Linux guests (definition: *Para-virtualization*)
- Software virtualization or emulation. Software virtualization uses binary translation and other emulation techniques to run unmodified operating systems. Software virtualization is significantly slower than hardware-assisted virtualization or para-virtualization.

Virtualized CPU

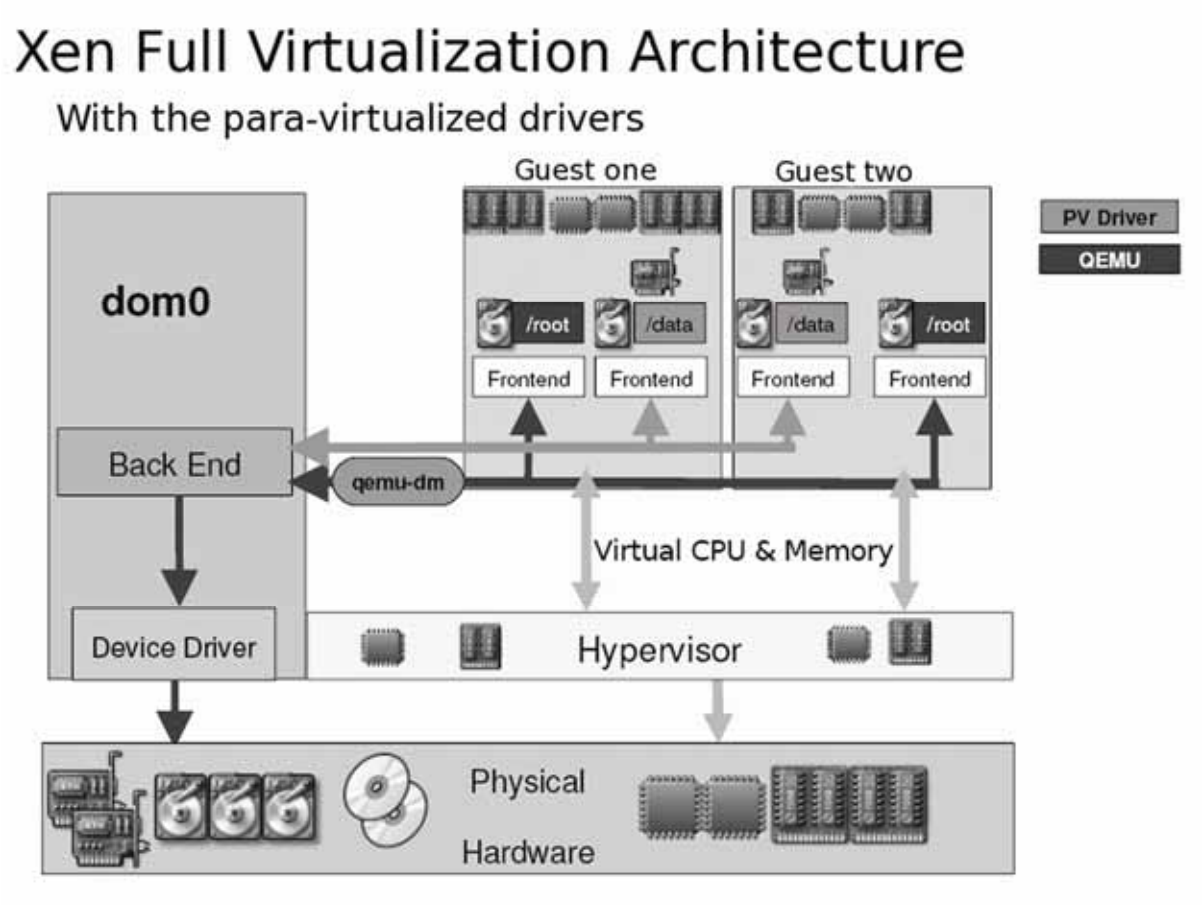
A system has a number of virtual CPUs (VCPUs) relative to the number of physical processor cores. The number of virtual CPUs is finite and represents the total number of virtual CPUs that can be assigned to guest virtual machines.

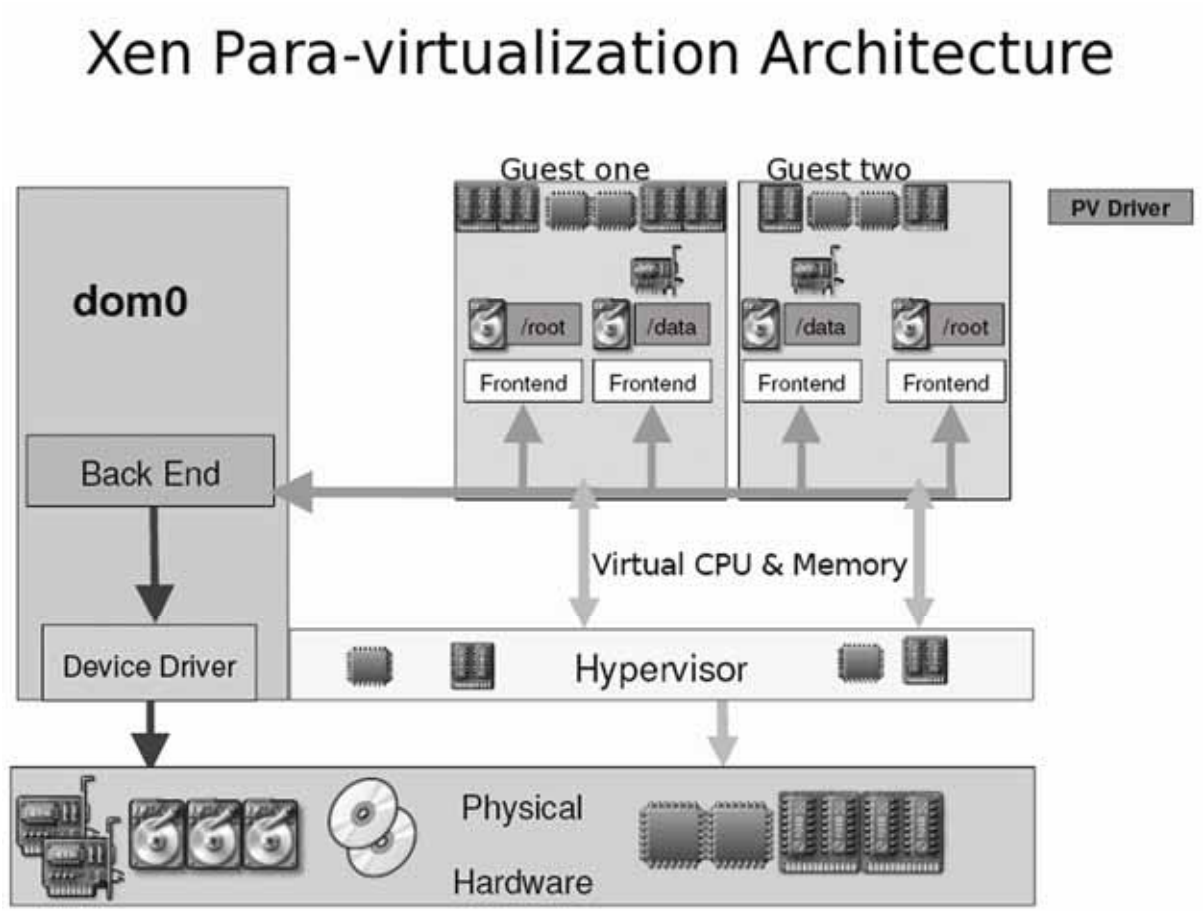
Virtual machines

A virtual machine is a software implementation of a physical machine or programming language (for example the Java Runtime Environment or LISP). Virtual machines in the context of virtualization are operating systems running on virtualized hardware.

Xen

Fedora supports the Xen hypervisor and the KVM hypervisor (refer to Kernel-based Virtual Machine). Both hypervisors have different architectures and development approaches. The Xen hypervisor runs underneath a Linux operating system which acts as a host managing system resources and virtualization APIs. The host is sometimes referred to as as *dom0* or Domain0.


Xen Full Virtualization Architecture
With the para-virtualized drivers
Guest one
Guest two
PV Driver
QEMU
dom0
/root
/data
/data
/root
Frontend
Frontend
Frontend
Frontend
Back End
qemu-dm
Virtual CPU & Memory
Device Driver
Hypervisor
Physical
Hardware


Xen Para-virtualization Architecture
Guest one
Guest two
PV Driver
dom0
/root
/data
/data
/root
Frontend
Frontend
Frontend
Frontend
Back End
Virtual CPU & Memory
Device Driver
Hypervisor
Physical
Hardware

Fedora 12
Official Documentation Collection

Title	Author	Edition	ISBN- 10	ISBN- 13
Fedora 12 Installation Guide	Fedora Documentation Project	paperback	1-59682-179-5	978-1-59682-179-8
		eBook (pdf)	1-59682-184-1	978-1-59682-184-2
Fedora 12 User Guide	Fedora Documentation Project	paperback	1-59682-180-9	978-1-59682-180-4
		eBook (pdf)	1-59682-185-X	978-1-59682-185-9
Fedora 12 Security Guide	Fedora Documentation Project	paperback	1-59682-181-7	978-1-59682-181-1
		eBook (pdf)	1-59682-186-8	978-1-59682-186-6
Fedora 12 SE Linux User Guide	Fedora Documentation Project	paperback	1-59682-182-5	978-1-59682-182-8
		eBook (pdf)	1-59682-187-6	978-1-59682-187-3
Fedora 12 Virtualization Guide	Fedora Documentation Project	paperback	1-59682-183-3	978-1-59682-183-5
		eBook (pdf)	1-59682-188-4	978-1-59682-188-0
http://www.linbrary.com				