



Learn by doing: less theory, more results

VirtualBox 3.1

Deploy and manage a cost-effective virtual environment using VirtualBox

Beginner's Guide

Alfonso V. Romero

[PACKT]
PUBLISHING

www.allitebooks.com

VirtualBox 3.1: Beginner's Guide

Deploy and manage a cost-effective virtual environment using VirtualBox

Alfonso V. Romero



BIRMINGHAM - MUMBAI

VirtualBox 3.1: Beginner's Guide

Copyright © 2010 Packt Publishing

All rights reserved. No part of this book may be reproduced, stored in a retrieval system, or transmitted in any form or by any means, without the prior written permission of the publisher, except in the case of brief quotations embedded in critical articles or reviews.

Every effort has been made in the preparation of this book to ensure the accuracy of the information presented. However, the information contained in this book is sold without warranty, either express or implied. Neither the author, Packt Publishing, nor its dealers or distributors will be held liable for any damages caused or alleged to be caused directly or indirectly by this book.

Packt Publishing has endeavored to provide trademark information about all the companies and products mentioned in this book by the appropriate use of capitals. However, Packt Publishing cannot guarantee the accuracy of this information.

First published: April 2010

Production Reference: 1090410

Published by Packt Publishing Ltd.
32 Lincoln Road
Olton
Birmingham, B27 6PA, UK.

ISBN 978-1-847199-14-0

www.packtpub.com

Cover Image by Faiz Fattohi (faizfattohi@gmail.com)

Credits

Author

Alfonso V. Romero

Editorial Team Leader

Akshara Aware

Reviewer

James Selvakumar Samraj Eliyezar

Project Team Leader

Lata Basantani

Acquisition Editor

Sarah Cullington

Project Coordinator

Shubhanjan Chatterjee

Development Editor

Reshma Sundaresan

Proofreader

Joel T. Johnson

Technical Editor

Akash Johari

Production Coordinator

Adline Swetha Jesuthas

Indexer

Rekha Nair

Cover Work

Adline Swetha Jesuthas

About the Author

Alfonso V. Romero has been working with Linux and open source software since 1999, when he started operating his first web server (Apache) from a PC at home. Since then he's been working for several clients in Mexico as a Java, C++, and web applications developer. He also works for Pearson Education in Mexico as a Computer Books Freelance Translator and Consultant. When he's not experimenting with new trends in open source and .NET applications, he enjoys spending some quality time with his beautiful wife, three kids, and three dogs, or playing his old electric Ibanez guitar.

He's also a big fan of Stephen King, and one of his maximum aspirations is to write a fiction novel, but his passion for computers and information technology keeps him busy as a technical writer.

To God, without whom I'd never be where I am now...

To Adelina my wife, the love of my life. I cannot thank you enough for your patience, love, encouragement, and support that kept me going no matter what happened along the way... I love you very, very much! My heart will always be yours!

To my two daughters Adelina and Arlae, and to my son Alfonso Jr., because every time I got writer's block they always managed to cheer me up with a smile, a kiss, or a hug... I love you infinitely, guys! Finally I'm going to be able to spend quality time with you in the park!

To my mother Estela, my father Alfonso, my brother Richard, and my sister Lucy, for their unconditional support, and for showing me that patience, perseverance, and stubbornness will always help you achieve whatever you're up to.

To Sarah, Pallabi, Shubhanjan, Reshma, and all the staff at Packt Publishing for being such a great support team.

To the VirtualBox team, for developing such a great product!

About the Reviewer

James Selvakumar Samraj Eliyezar is a Software Engineer who graduated from National Engineering College, Kovilpatti, India. He has more than five years of experience in developing applications for banking and financial industry. He develops primarily in Java but loves dynamic languages such as Groovy and Python. Ubuntu Linux is his favorite operating system. In his free time, he likes to blog, tweet, and read.

I would love to thank my mother Elizabeth, wife Serene, and my little son Nithil Jeremiah who are the constant source of motivation and encouragement. I would also like to thank my boss Davethu Jacob for giving me the freedom to explore and innovate.

Table of Contents

Preface	1
Chapter 1: Getting to Work with VirtualBox	7
Running multiple virtual machines	7
Installing VirtualBox on Windows	9
Time for action – downloading and installing VirtualBox on Windows	9
Installing VirtualBox on Linux	15
Time for action – downloading and Installing VirtualBox on Linux	15
Testing VirtualBox	28
Time for action – creating and testing a Damn Small Linux virtual machine	28
Summary	42
Chapter 2: Creating Your First Virtual Machine: Ubuntu Linux	43
Getting started	44
Downloading the Ubuntu Linux Live CD	44
Time for action – downloading the Ubuntu Desktop Live CD	44
Creating your Ubuntu Linux VM	45
Time for action – creating a virtual machine	46
Configuring basic settings for your Ubuntu Linux VM	50
Time for action – basic configuration for your VM	51
Installing Ubuntu Linux on your VM	53
Time for action – installing Ubuntu Desktop on your VM	54
Running your Ubuntu Linux VM	62
Time for action – running Ubuntu Linux	62
Web browsing with Mozilla Firefox	64
Time for action – web browsing in your Ubuntu VM	64
Using OpenOffice.org in your virtual machine	66
Time for action – using OpenOffice.org	66
Shutting down your virtual machine	71
Time for action – shutting down your VM	71

Summary	75
Chapter 3: Creating Your Second Virtual Machine: Windows 7	77
Creating your Windows VM	78
Time for action – creating a virtual machine	78
Booting your Windows 7 installation disk	84
Time for action – booting your Windows 7 installation disk through the First Run Wizard	84
Installing Windows 7 on your VM	86
Time for action – installing Windows XP on your VM	86
Making sound work on your Windows 7 VM	92
Time for action – enabling audio on your Windows 7 virtual machine	92
Removing the installation media from your Windows 7 VM	94
Time for action – removing installation media from your VM	94
Web browsing with Internet Explorer	95
Time for action – web browsing in your Windows 7 VM	96
Using Microsoft Office in your virtual machine	98
Time for action – using OpenOffice.org	98
Shutting down your virtual machine	102
Time for action – shutting down your VM	102
Summary	105
Chapter 4: Installing Guest Additions and Advanced Settings	107
Introducing Guest Additions	108
Installing Guest Additions for Windows	109
Time for action – installing Guest Additions on a Windows XP virtual machine	109
Installing Guest Additions for Linux	112
Time for action – installing Guest Additions on Linux Ubuntu	112
Installing Guest Additions for OpenSolaris	115
Using the fullscreen feature	117
Time for action – using the fullscreen and windowed modes	118
Sharing folders between your host and guest PCs	120
Time for action – sharing folders between a Windows XP host and a Ubuntu guest	120
Activating the Seamless Windows feature	128
Time for action – activating Seamless Windows with Windows and Linux	128
Allowing 3D Hardware Acceleration in your virtual machines	131
Time for action – using Compiz on your Ubuntu VM	132
Summary	138

Chapter 5: Storing Data in VirtualBox	139
Using Virtual Disks in VirtualBox	140
Using an additional VDI hard drive	141
Time for action – adding a secondary virtual drive to your VM	141
Using a VHD hard drive	151
Time for action – adding a VHD virtual drive to your VM	151
Creating multiple virtual machines by cloning	155
Time for action – cloning an Ubuntu Linux hard disk image	156
Expanding hard disk images on the fly	160
Time for action – creating a fixed-size hard drive image	161
Choosing your disk controller type: IDE, SATA, or SCSI	163
Time for action – using a SATA disk controller on a VM	164
Using IDE and SATA drives on a VM	166
Time for action – using IDE and SATA drives	166
Summary	170
Chapter 6: Networking with Virtual Machines	173
Connecting to the default NAT mode	174
Exploring default network adapter types	174
Time for action – viewing the default network adapter types in your virtual machines	175
Testing the NAT mode	178
Time for action – accessing the NAT mode in your VM	179
Using port-forwarding with the NAT mode	184
Time for action – enabling port-forwarding in NAT mode	184
Testing a server operating system in the bridged networking mode	187
Accessing your VM's web server from your host PC	187
Time for action – changing your virtual machine to bridged networking mode	188
Accessing your VM's web server from another VM	191
Time for action – accessing your VM's web sever from another VM	191
Using the 'Not Attached' mode	194
Time for action – isolating a VM with the 'Not Attached' mode	194
Disconnecting your virtual machine from the network without shutting it down	197
Time for action – connecting/disconnecting your VM from the network	197
Using the Internal Networking mode	199
Time for action – communicating between VMs only	199
Using the Host-Only Networking mode	206
Time for action – communicating between VMs and your host PC only	206
Summary	210

Chapter 7: Using Virtual Appliances	213
Setting up preconfigured virtual machines in a flash	213
Importing a virtual appliance	214
Time for action – using the TurnKey Wordpress virtual appliance	215
Exporting a virtual appliance	221
Time for action – exporting your customized Wordpress virtual appliance	221
Working with virtual appliances	228
Using virtual images from VirtualBox® Images	228
Time for action – using a PuppyLinux VM in VirtualBox	228
Using virtual appliances from BitNami	233
Time for action – using the BitNami Drupal virtual appliance	233
Using the Turnkey Linux File Server appliance	239
Time for action – using the Turnkey Linux File Server appliance	239
Summary	249
Chapter 8: Managing your Virtual Machines from a Remote Computer	251
Managing virtual machines from alternative front-ends	252
Using the VBoxManage frontend	252
Time for action – using VBoxManage to start a virtual machine	253
Controlling your virtual machines through VBoxManage	257
Time for action – pausing, resuming, and saving your virtual machine's state	257
Using the VBoxSDL simplified interface	260
Time for action – using VBoxSDL to start a virtual machine	260
Setting up your very own VirtualBox headless server	262
Setting up Ubuntu Server 8.04 LTS	262
Time for action – download and install Ubuntu Server 8.04 LTS	263
Accessing your headless server from a remote PC	268
Time for action – using PuTTY to access your Ubuntu server remotely	268
Installing VirtualBox on your Ubuntu server through apt-get	270
Time for action – installing VirtualBox through apt-get on your Ubuntu server	271
Creating, managing, and running your first remote virtual machine on the Ubuntu headless server	275
Enabling FTP to upload guest images to your headless server	276
Time for action – enabling proftpd on your Ubuntu headless server	276
Uploading an ISO guest image to your Ubuntu server	277
Time for action – uploading a guest ISO image to your headless server	278
Creating a virtual machine in your Ubuntu headless server	281
Time for action – creating a virtual machine with VBoxManage	281
Using a Remote Desktop client and running your remote VM	283
Time for action – using an RDP viewer and starting your VM	284
Enabling sound on your remote virtual machines	290

Time for action – enabling audio on your remote virtual machine	290
Using shared folders on your remote virtual machine	292
Time for action – creating and accessing a shared folder on your Ubuntu headless server	292
Setting up your own remote virtual LAMP server	295
Time for action – running your very own remote virtual LAMP server	295
Summary	300
Appendix A: Using Snapshots	303
Reverting changes to your virtual machine	303
Time for action – saving the state of your Ubuntu virtual machine by taking a snapshot	304
Creating alternate realities in your virtual machines	308
Time for action – using branching snapshots in your VMs	308
Summary	315
Appendix B: Pop Quiz Answers	317
Chapter 1	317
Doing the thing	317
Chapter 2	317
Creating virtual machines	317
Configuring basic settings on your VMs	318
Using the auto capture and host key features	318
Running your Ubuntu Linux VM	318
Your first VM	318
Chapter 3	319
Creating virtual machines	319
Using the auto capture and host key features	319
Running your Windows 7 VM	319
Your first VM	319
Chapter 4	320
Guest Additions	320
Using the fullscreen feature	320
Using the folder sharing feature	320
Chapter 5	321
Using virtual disks in VirtualBox	321
Creating additional virtual disk images	321
Virtual storage	321
Using different storage controller types	321
Storing data in VirtualBox	321
Chapter 6	322
Working with the default network adapter types	322

Table of Contents

Using the bridged networking mode	322
Using the internal networking mode	322
Virtual networking	322
Chapter 7	323
Importing virtual appliances	323
Exporting virtual appliances	323
Working with virtual appliances	323
Virtual appliances	323
Chapter 8	324
Using the VBoxManage interface	324
Setting up your own VirtualBox headless server	324
Enabling FTP and uploading ISO images on your headless server	324
Remote virtual machines and alternative front-ends	324
Appendix A	325
Using the folder sharing feature	325
Creating alternate realities for your VMs	325
Index	327

Preface

The furore around virtualization is taking the technology world by storm and is a must for efficient utilization of network server capacity, storage administration, energy, and capital. VirtualBox is free which brings down your upfront costs for an agile data center. VirtualBox will transform your IT infrastructure into a lean data center on a Windows XP/Vista/7, Windows 2003/2008 Server, Linux, Macintosh, or OpenSolaris platform. Although VirtualBox has grown by leaps and bounds, there is not enough documentation to guide you through its features and implementation.

This hands-on guide gives you a thorough introduction to this award-winning virtualization product. It will help you implement the right virtual environment for you. Additionally, this book will help you set up an environment that will work for your system. You will learn to architect and deploy your first virtual machine without being overwhelmed by technical details.

This practical book unveils the robust capabilities and easy-to-use graphical interface of VirtualBox to help you effectively administer and use virtual machines in a home or office environment. You begin by creating your first virtual machine on a Windows/Linux guest operating system and installing Guest Additions. The book then goes on to discuss the various formats that VirtualBox supports and how it interacts with other formats. The comprehensive instructions will help you to work with all the networking modes offered by VirtualBox. Virtual appliances will be explained in detail—how they help to reduce installation time for virtual machines and run them from VirtualBox.

By the end of this book, you will be able to run your own headless VirtualBox server to create, manage, and run virtual machines in that server from a remote PC.

What this book covers

Chapter 1, *Getting to Work with VirtualBox*, gives you a practical introduction to VirtualBox, and how you can use this excellent software to run several virtual machines inside a Windows/Linux host.

Chapter 2, *Creating Your First Virtual Machine: Ubuntu Linux*, teaches you about the basic settings needed to create and run a virtual machine using the Ubuntu operating system. You'll also learn to test some basic functions like web browsing and using the OpenOffice suite of applications.

Chapter 3, *Creating Your Second Virtual Machine: Windows 7*, teaches you how to install Windows 7 on a virtual machine, along with the basic settings needed. You'll also learn to test some basic functions like web browsing and using the Microsoft Office 2007 trial edition.

Chapter 4, *Installing Guest Additions and Advanced Settings*, teaches you how to install the Guest Additions to take real advantage of your virtual machines. You'll also learn how to share folders between your host PC and your guest virtual machines, how to use seamless windows to integrate your host and guest desktops, and how to activate hardware 3D acceleration on your virtual machines.

Chapter 5, *Storing Data in VirtualBox*, teaches you everything about using virtual hard disks, and how you can use different disk formats such as VDI, VMDK, and VHD seamlessly in your virtual machines. You'll also learn the differences between 'cloning' and 'copying' virtual hard disks, between fixed and dynamically expanding hard drive images, and how to choose between the different hard disk controllers used by VirtualBox.

Chapter 6, *Networking with Virtual Machines*, teaches you all about the five networking modes available in VirtualBox and how to choose the most appropriate mode for your virtual machines, depending on your needs. You'll learn how to connect a virtual machine to the Internet and a local LAN via the default NAT mode, and how to choose the best networking mode when using a virtual machine as a web server, among other things.

Chapter 7, *Using Virtual Appliances*, teaches you about using virtual appliances and how they can help make your life easier when using virtual machines and VirtualBox. You'll also learn how to import and export virtual machines using the OVF standard.

Chapter 8, *Managing Your Virtual Machines From a Remote Computer*, teaches you about the alternative front ends available in VirtualBox, and how you can use them to run your virtual machines based on your specific needs. You will also learn how to set up your very own Ubuntu headless server and use it to run a remote virtual machine from a Windows XP desktop PC.

Appendix A, *Using Snapshots*, teaches you how to save the state of your virtual machines through the snapshots feature. You'll also learn to create alternate realities in your virtual machines, where you can follow two or more completely independent time paths.

What you need for this book

To run VirtualBox, you need a host PC with a 32- or 64-bit Intel or AMD CPU 1.5 GHz or faster, at least 512 MB of RAM and 30 MB of disk space. However, you must take into account the fact that each virtual machine will require additional disk space, and the more memory you can get your hands on, the better your experience with VirtualBox will be, especially when dealing with new guest operating systems like Windows 7. So in short, any recent Intel or AMD processor will do, but get all the RAM and hard disk space you can!

You can use a Windows XP/Vista/7, Mac OS X, OpenSolaris, or Linux host operating system. I recommend using an Ubuntu Linux or Windows 7 host. For the guest operating systems, you can use practically every Windows and Linux OS available. For more information you can visit http://www.virtualbox.org/wiki/Guest_OSes, where you can get a list of all the guest operating systems supported by VirtualBox.

Who this book is for

If you are a System Administrator who needs to set up a virtual machine and want to use an open source tool to do it, this book will prove invaluable. No prior knowledge of VirtualBox is required, but you should have experience with general system administration.

Conventions

In this book, you will find a number of styles of text that distinguish between different kinds of information. Here are some examples of these styles, and an explanation of their meaning.

Code words in text are shown as follows: “Open a terminal window and type
`sudo apt-get install dkms`”

A block of code is set as follows:

```
mkdir myshared VBoxManage sharedfolder add "UbuntuVB" --name  
    "myshared" --hostpath "/home/aromero/myshared"
```

Any command-line input or output is written as follows:

```
"C:\Program Files\Sun\VirtualBox\VBoxManage" setextradata "UbuntuVB"  
    "VBoxInternal/Devices/pcnet/0/LUN#0/Config/webserver/Protocol" "TCP"
```

New terms and **important words** are shown in bold. Words that you see on the screen, in menus or dialog boxes for example, appear in the text like this: “Scroll down until you locate the **New password** section”.



Warnings or important notes appear in a box like this.



Tips and tricks appear like this.

Reader feedback

Feedback from our readers is always welcome. Let us know what you think about this book—what you liked or may have disliked. Reader feedback is important for us to develop titles that you really get the most out of.

To send us general feedback, simply send an e-mail to feedback@packtpub.com, and mention the book title via the subject of your message.

If there is a book that you need and would like to see us publish, please send us a note in the **SUGGEST A TITLE** form on www.packtpub.com or e-mail suggest@packtpub.com.

If there is a topic that you have expertise in and you are interested in either writing or contributing to a book on, see our author guide on www.packtpub.com/authors.

Customer support

Now that you are the proud owner of a Packt book, we have a number of things to help you to get the most from your purchase.



Downloading the example code for the book

Visit http://www.packtpub.com/files/code/9140_Code.zip to directly download the example code.

The downloadable files contain instructions on how to use them.

Errata

Although we have taken every care to ensure the accuracy of our content, mistakes do happen. If you find a mistake in one of our books—maybe a mistake in the text or the code—we would be grateful if you would report this to us. By doing so, you can save other readers from frustration and help us improve subsequent versions of this book. If you find any errata, please report them by visiting <http://www.packtpub.com/support>, selecting your book, clicking on the **let us know** link, and entering the details of your errata. Once your errata are verified, your submission will be accepted and the errata will be uploaded on our website, or added to any list of existing errata, under the Errata section of that title. Any existing errata can be viewed by selecting your title from <http://www.packtpub.com/support>.

Piracy

Piracy of copyright material on the Internet is an ongoing problem across all media. At Packt, we take the protection of our copyright and licenses very seriously. If you come across any illegal copies of our works, in any form, on the Internet, please provide us with the location address or website name immediately so that we can pursue a remedy.

Please contact us at copyright@packtpub.com with a link to the suspected pirated material.

We appreciate your help in protecting our authors, and our ability to bring you valuable content.

Questions

You can contact us at questions@packtpub.com if you are having a problem with any aspect of the book, and we will do our best to address it.

1

Getting to Work with VirtualBox

Hi, and welcome to this book about VirtualBox, the best virtualization software I've ever used! Instead of boring you with a chapter full of theory to explain all the "knicks and knacks" about VirtualBox, I'm going to give you a tiny introduction to virtualization and then jump straight ahead into the installation process, ok?

In this chapter you will:

- ◆ Learn about virtualization and how you can take advantage of this technology
- ◆ Learn to install VirtualBox on Windows XP
- ◆ Learn to install VirtualBox on Ubuntu Linux

And now, let the show begin...

Running multiple virtual machines

The first time I heard about VirtualBox was when I was looking for something to help me make tutorials for Windows users trying to learn to use Linux. I needed something to help me run Ubuntu Linux inside my main Windows XP PC to write my tutorials in MS Word, capture all the Linux screenshots, and paste them directly into the MS Word document. I've worked with this excellent and powerful virtualization software since version 2.0, and I definitely give two thumbs up to the guys in the development team because version 3 has come a really long way, as we'll see throughout the exercises in this book, especially in areas related to the 3D Acceleration and USB support features.

For those of you new to **virtualization**, let me put it this way: With VirtualBox, you can use a Windows, Linux, Mac OS X, or Solaris PC to run one or more 'virtual' PCs inside it. Each **virtual machine** is called a **guest**, and your main PC is called a **host**. The bottom line is that you can run one 'host' operating system and several 'guest' operating systems inside that 'host'!

For example, if you have a Windows PC with 4 GB of RAM and a 320 GB hard disk, you can use VirtualBox to run three virtual machines side by side on the same computer running Windows. The following table shows how you can distribute your physical resources among the three virtual machines and your Windows PC:

Machine	Resources	Description
Windows XP (host)	RAM: 1 GB Hard Disk: 80 GB	This is your physical machine.
Linux Virtual Machine (guest 1)	RAM: 1 GB Hard Disk: 80 GB	This is the first virtual machine. You can use practically any Linux distro available!
Open Solaris Virtual Machine (guest 2)	RAM: 1 GB Hard Disk: 80 GB	This is the free, open source version of the Solaris operating system.
Windows 7 Virtual Machine (guest 3)	RAM: 1 GB	This is a new Windows release.

Naturally, you don't need to use all your VMs at the same time! Most of the time, you'll end up having a lot of virtual machines on your host PC and running only one or two of them in parallel. And don't worry about your computer's resources. Having an 80 GB virtual hard disk doesn't mean that your virtual machine is going to take up all that storage space immediately. Thanks to VirtualBox, you can have dynamically expanding storage, which means the initial size of your virtual machine's hard disk will be very small, and it will increase its size to take up more space as needed.

That means you can have more than four virtual machines with 80 GB hard drives on the Windows XP host computer, in the above example, because their virtual hard drives won't take up their full size immediately. However, it's better to keep in mind your physical hard disk space limitations when creating your virtual machines in order to avoid the typical problems that arise when your hard disk is full!

But that's enough chatter for now! You'll learn more about virtualization and VirtualBox with all the exercises I've prepared throughout the book, so let's get going!

Installing VirtualBox on Windows

Now let's get our hands dirty with the first exercise in this chapter! If you're going to install VirtualBox on Linux, feel free to skip this section. I'm going to use Windows XP in this exercise because it's currently the most widely used version of Windows. But the procedure is similar in Windows Vista or Windows 7, so you can use whichever one suits your needs the best.

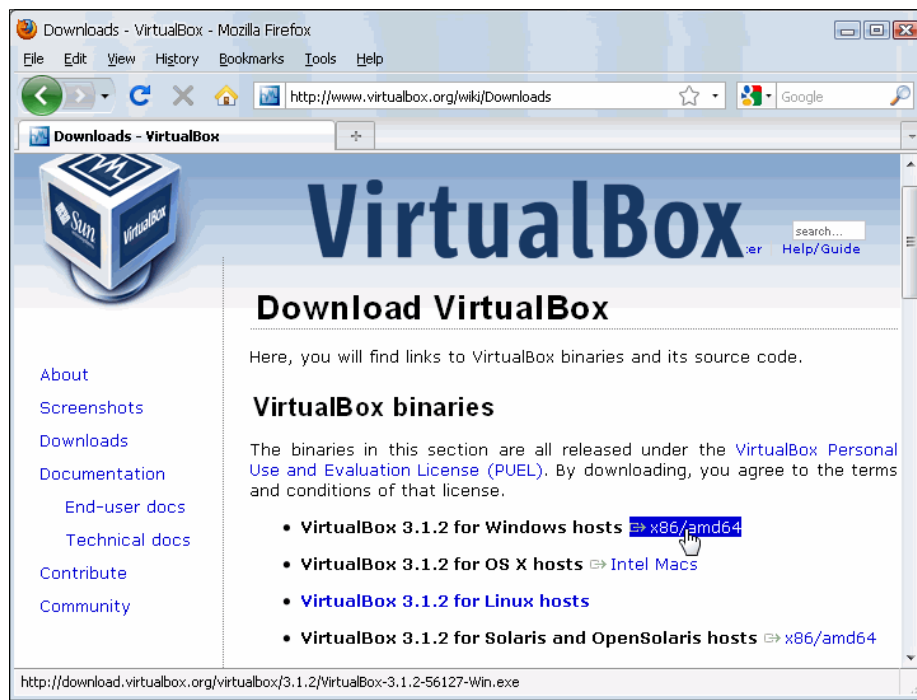
Time for action – downloading and installing VirtualBox on Windows

In this exercise, you'll see how easy it is to download VirtualBox and install it in a Windows XP system.

1. Open your Web browser, and type `http://www.virtualbox.org` in the address bar. The VirtualBox main page will appear:



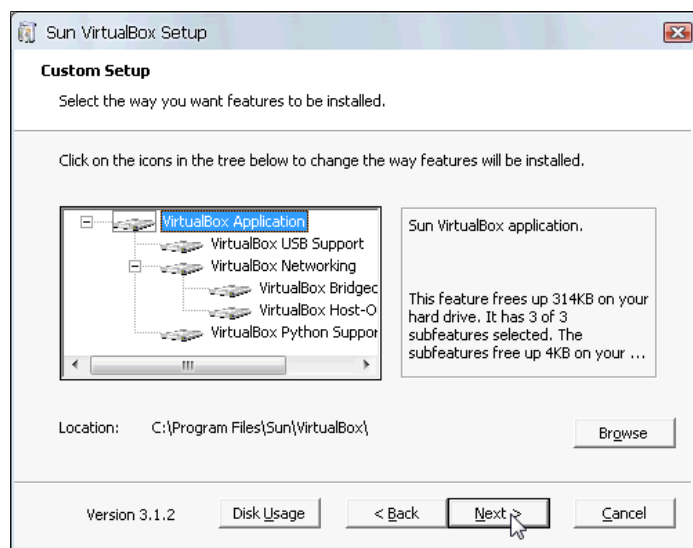
2. Click on the **Downloads** link located in the left sidebar. The **Download VirtualBox** page will show up:



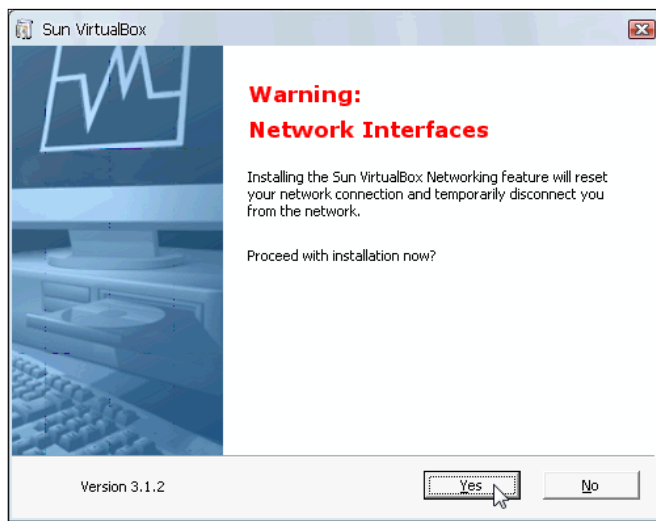
3. Now click on the **x86/amd64** link at the right of the **VirtualBox 3.1.X for Windows hosts** line. A dialog box will pop up, asking if you want to open or save the VirtualBox installation file. Click on **Save** to start downloading this file.
4. Once the downloading process ends, double-click on the downloaded file (it should be something like **VirtualBox-3.X.X-Win.exe**), and wait for the **Do you want to open this file?** warning dialog box to appear. Click on **Run**, and the **Sun xVM VirtualBox Setup** dialog will appear:



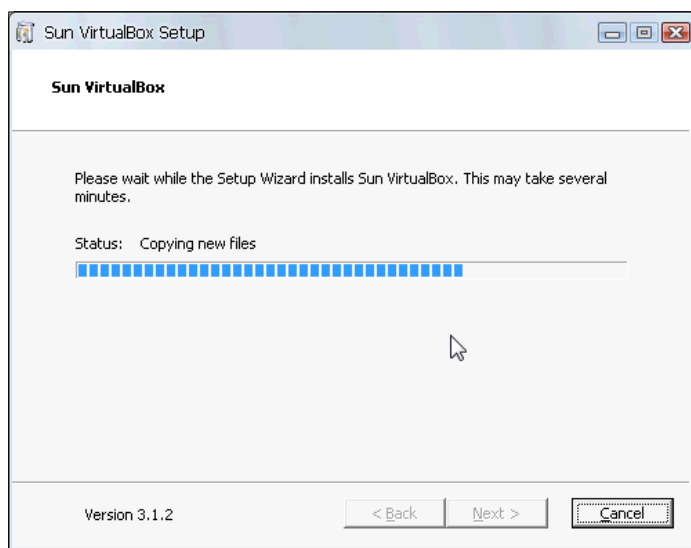
5. Click on the **Next** button to start installing VirtualBox on your PC. The next dialog will ask you to read the **VirtualBox Personal Use and Evaluation License (PUEL)**. If you agree with it, select the **I accept the terms in the License Agreement** radio button, and click on **Next** to continue.
6. The next dialog, **Custom Setup**, will show you the VirtualBox features that will be installed as well as the installation directory. Leave the default options, and click on **Next** to continue:



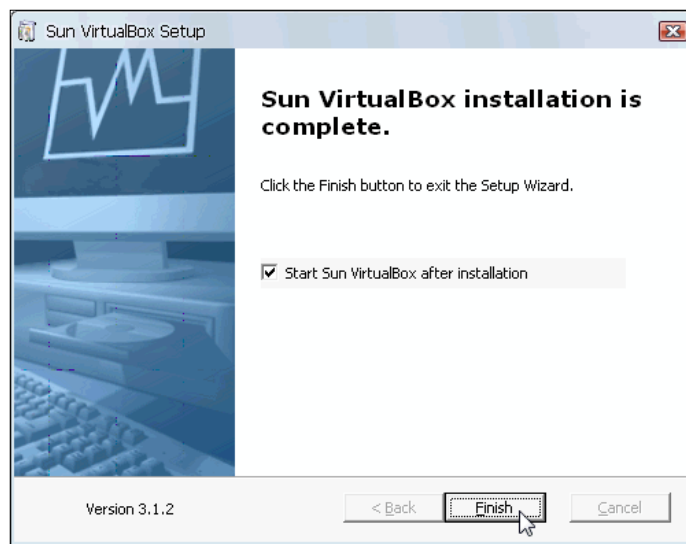
7. On the next dialog, you need to choose if you want to create a shortcut to run VirtualBox from the desktop and/or from the Quick Task Bar. You can leave both options selected, and click on **Next** to continue.
8. The next dialog will warn you about temporarily disconnecting you from the local area network because the setup program will install the **Sun xVM VirtualBox Networking** feature:



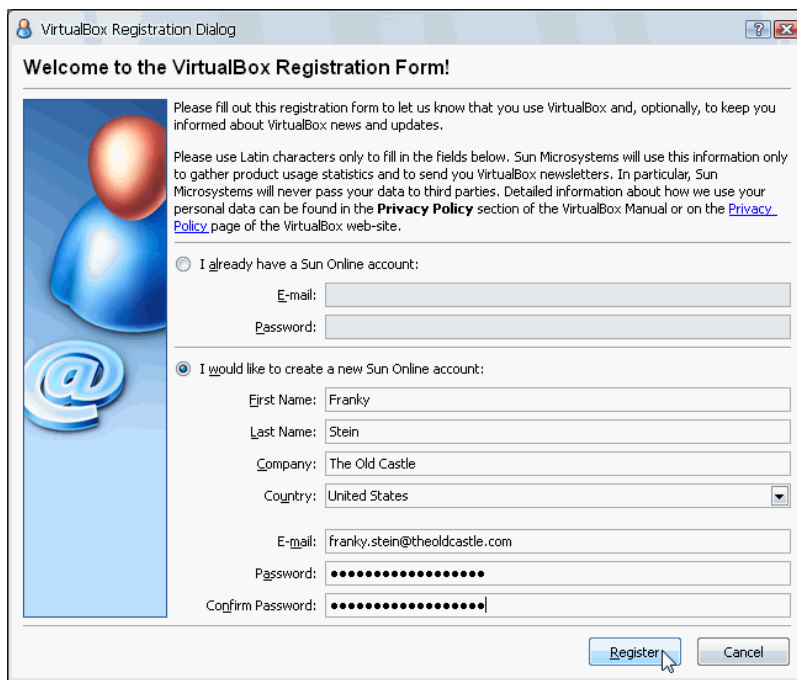
9. Click on **Yes**, and then on **Install** to start installing VirtualBox on your computer:



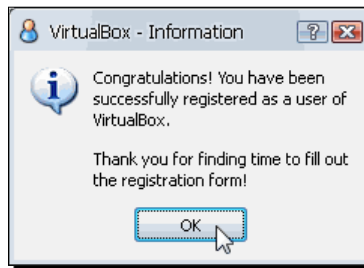
10. Once the installation process finishes, you'll see the following dialog:



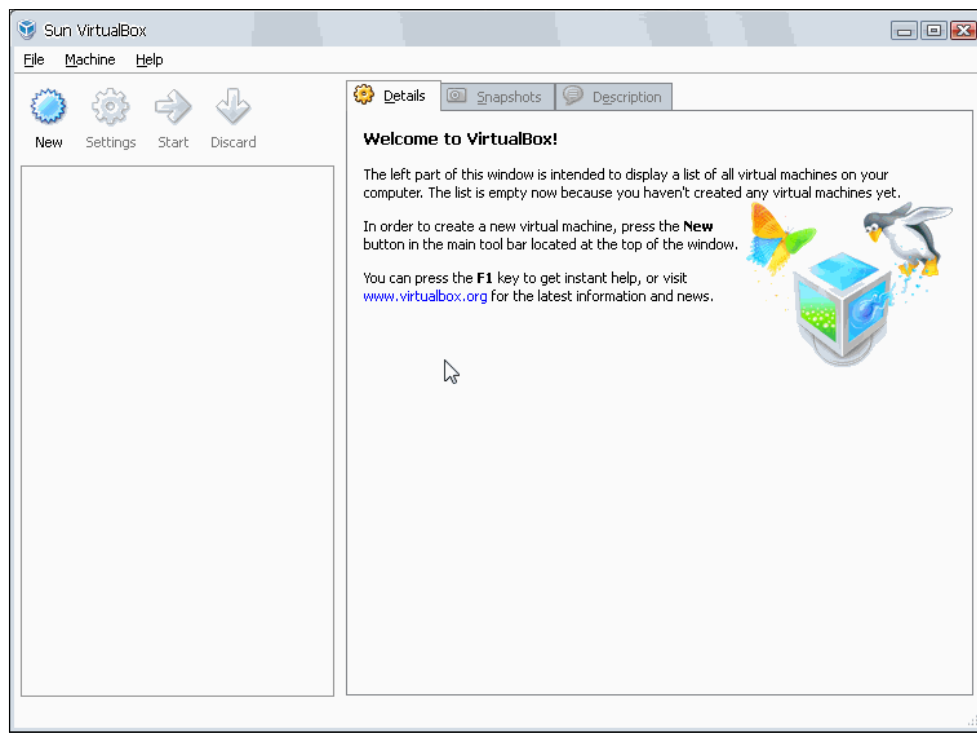
11. Click on **Finish** to exit the setup wizard and open the VirtualBox main window. The **VirtualBox Registration Dialog** will appear next:

The image shows the 'VirtualBox Registration Dialog' window. The title bar says 'VirtualBox Registration Dialog'. The main text reads 'Welcome to the VirtualBox Registration Form!'. Below this, there is a paragraph of text: 'Please fill out this registration form to let us know that you use VirtualBox and, optionally, to keep you informed about VirtualBox news and updates.' Another paragraph follows: 'Please use Latin characters only to fill in the fields below. Sun Microsystems will use this information only to gather product usage statistics and to send you VirtualBox newsletters. In particular, Sun Microsystems will never pass your data to third parties. Detailed information about how we use your personal data can be found in the [Privacy Policy](#) section of the VirtualBox Manual or on the [Privacy Policy](#) page of the VirtualBox web-site.' There are two radio buttons: 'I already have a Sun Online account:' and 'I would like to create a new Sun Online account:'. The second option is selected. Below the first option are fields for 'E-mail:' and 'Password:'. Below the second option are fields for 'First Name:', 'Last Name:', 'Company:', 'Country:', 'E-mail:', 'Password:', and 'Confirm Password:'. The 'First Name' field contains 'Franky', 'Last Name' contains 'Stein', 'Company' contains 'The Old Castle', and 'Country' is set to 'United States'. The 'E-mail' field contains 'franky.stein@theoldcastle.com'. The 'Password' and 'Confirm Password' fields are filled with dots. At the bottom right, there are 'Register' and 'Cancel' buttons. A mouse cursor is pointing at the 'Register' button.

- 12.** If you already have a Sun account, just fill in your email and password, and click on the **Register** button to register your copy of VirtualBox. If this is the first time you're using a Sun product, select the **I would like to register creating a new Sun Online account** radio button, and fill in the required fields. Then click on **Register**. Once you register with Sun, you'll see the following success dialog:



- 13.** Now click on the **OK** button to go to the VirtualBox main screen:



- 14.** You can close VirtualBox now.

What just happened?

See how easy it was to download and install VirtualBox on Windows? And wait until you start using it! You'll start wondering why someone didn't come up with VirtualBox ten years ago! Seriously speaking, however, you can use VirtualBox on the following Windows platforms:

- ◆ Windows XP, all service packs (32-bit)
- ◆ Windows Server 2003 (32-bit)
- ◆ Windows Vista (32-bit and 64-bit).
- ◆ Windows Server 2008 (32-bit and 64-bit)
- ◆ Windows 7 (32-bit and 64-bit)

And the installation process is very similar, if not identical, on most Windows versions. Just be sure to have enough RAM for your physical 'host' PC and one or more 'guest' virtual machines. Oh, and you have to be careful with hard disk space also! But you don't need to worry about that right now; we'll cover all those things throughout the exercises in this book.

Have a go hero – experiment with VirtualBox on Windows

If you have the ability to do so, it would be a great idea to try VirtualBox on other Windows versions, especially on Windows 7, the most recent Windows release. You can also start browsing around VirtualBox's user interface and see if you can create a virtual machine by yourself!

Installing VirtualBox on Linux

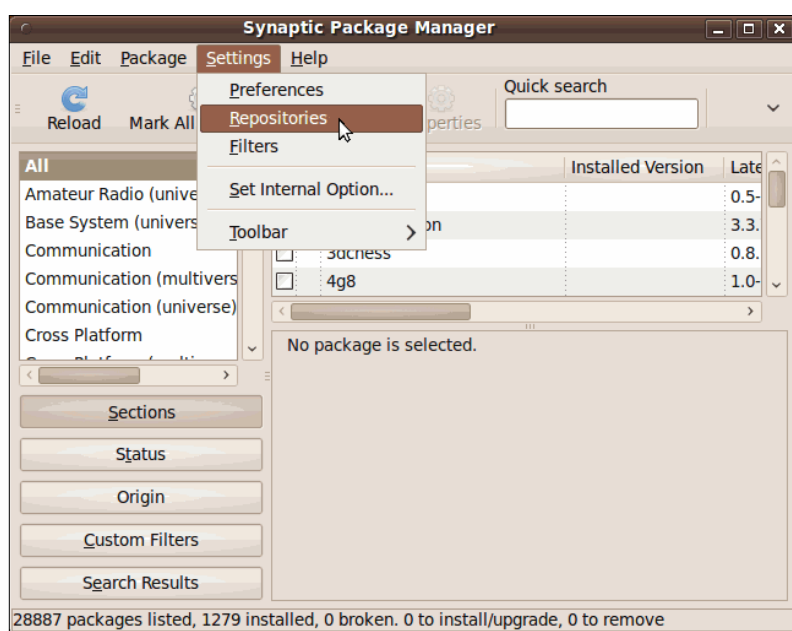
If you read the previous section about installing VirtualBox on Windows, then this will be your second hands-on exercise, but if you skipped the Windows section, let's go and get our hands dirty with VirtualBox and Ubuntu Linux!

Time for action – downloading and installing VirtualBox on Linux

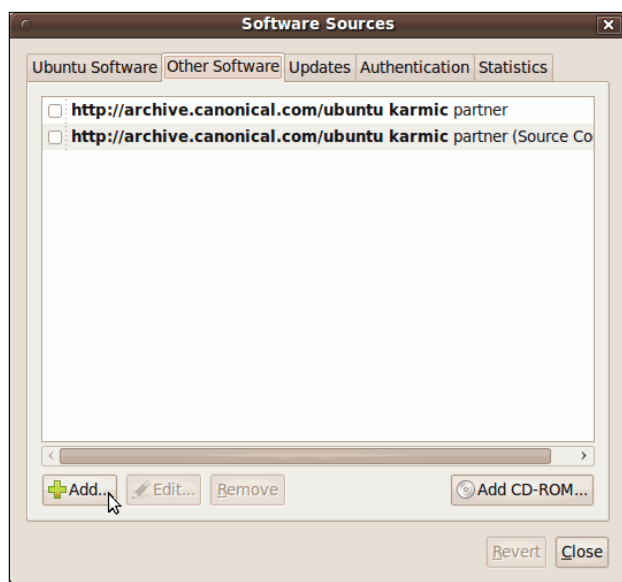
Ok, for this exercise you'll need a copy of Ubuntu Linux already installed on your PC. I chose Ubuntu because it's one of the friendliest Linux distributions available, as you will see in a moment.

- 1.** Before installing VirtualBox, you'll need to install two additional packages on your Ubuntu system. Open a terminal window (**Applications | Accessories | Terminal**), and type `sudo apt-get update`, followed by *Enter*. If Ubuntu asks for your administrative password, type it, and hit *Enter* to continue.
- 2.** Once the package list is updated, type `sudo apt-get install dkms`, and hit *Enter*; then type *Y* and hit *Enter* to install the DKMS package.

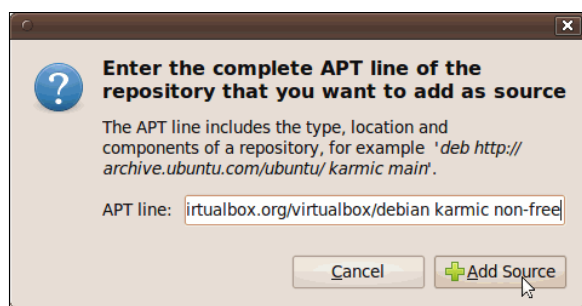
3. The other package needed before you can install VirtualBox is `build-essential`. This package contains all the compiling tools VirtualBox needs to build the kernel module. Type `sudo apt-get install build-essential`, and hit *Enter*. Then type `Y`, and hit *Enter* again to continue. Wait for the `$` prompt to show up again, type `exit`, and hit *Enter* to close the terminal window. Now you can proceed to install VirtualBox.
4. Open the Synaptic Package Manager (**System | Administration | Synaptic Package Manager**), and select the **Settings | Repositories** option in the menu bar (if Ubuntu asks for your administrative password, type it, and press *Enter* to continue):



5. The **Software Sources** dialog will appear. Click on the **Other** (on earlier Ubuntu versions the name of this tab is **Third-Party Software**) and then on the **Add+** button:

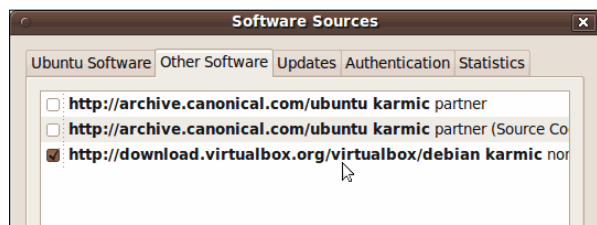


6. Another dialog box will show up. Now type `deb http://download.virtualbox.org/virtualbox/debian karmic non-free` on the **APT line** field, and click on the **Add Source** button:

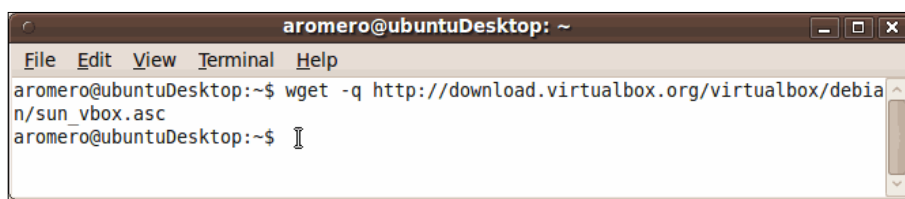


If you're not using Ubuntu 9.10 Karmic Koala, then you'll need to change the APT line in the previous step. For example, if you're using Ubuntu 9.04 Jaunty Jackalope, replace the `karmic` part with `jaunty`. On the http://www.virtualbox.org/wiki/Linux_Downloads webpage, you'll find more information about installing VirtualBox on several Linux distributions and the APT line required for each Ubuntu distribution available.

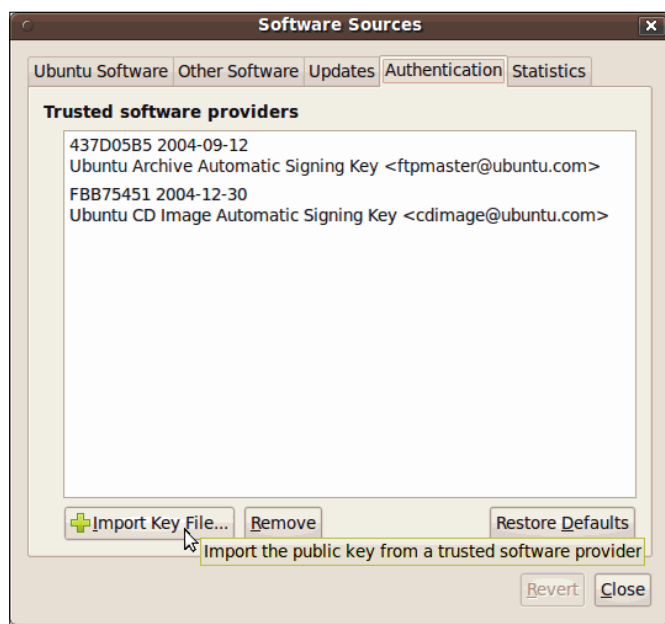
7. The third-party software source for VirtualBox will now show up on the list:



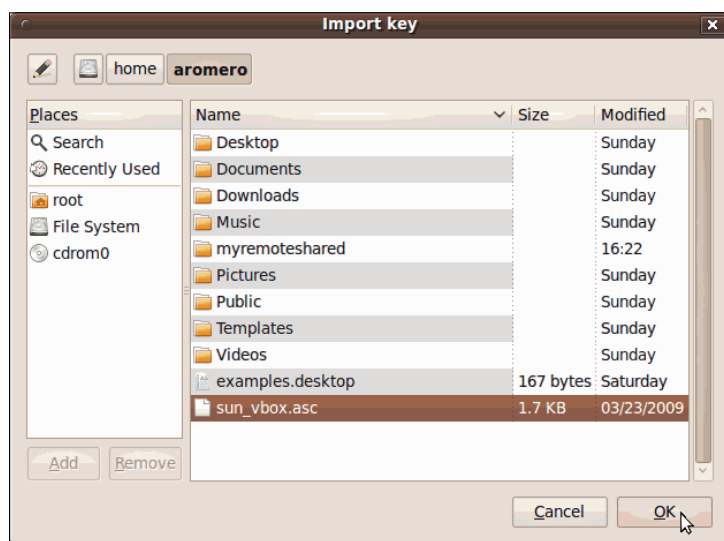
8. Now open a terminal window (**Applications | Accessories | Terminal**), and type `wget -q http://download.virtualbox.org/virtualbox/debian/sun_vbox.asc` to download the Sun public key:



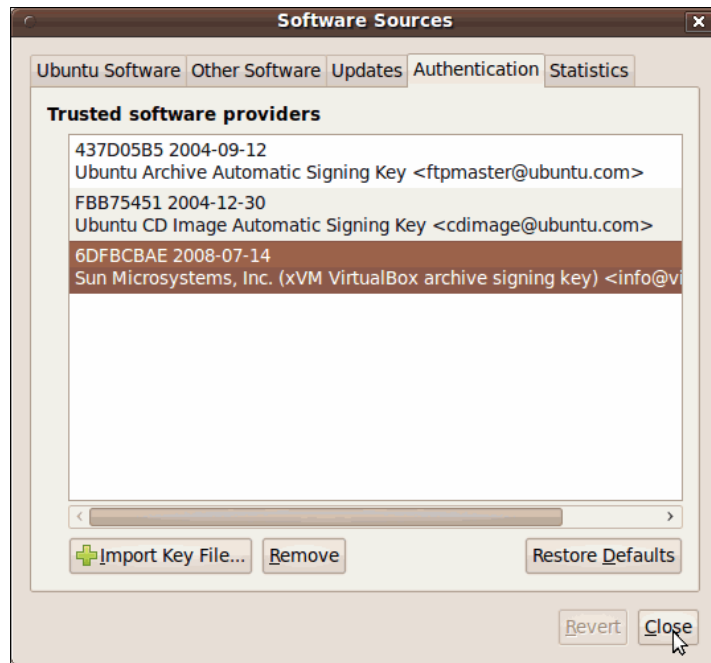
9. Go back to the Synaptic Manager, select the **Authentication** tab, and click on the **Import Key File** button:



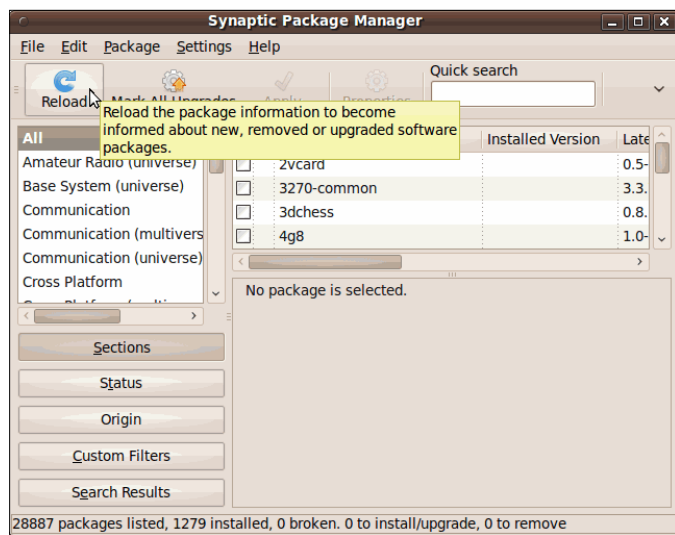
- 10.** The **Import Key** dialog will appear next. Select the `sun_vbox.asc` file you just downloaded, and click on the **OK** button to continue:



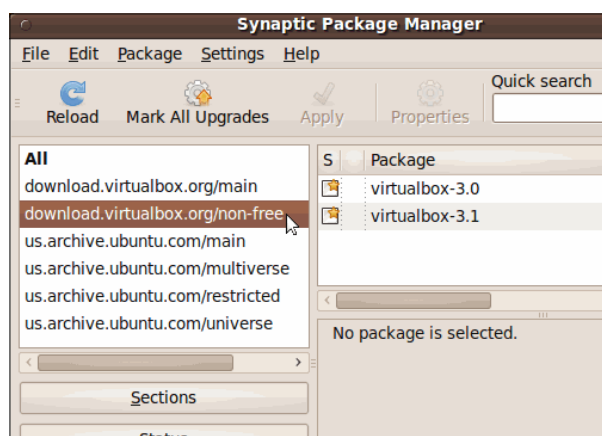
- 11.** The Sun public key for VirtualBox should now appear on the list:



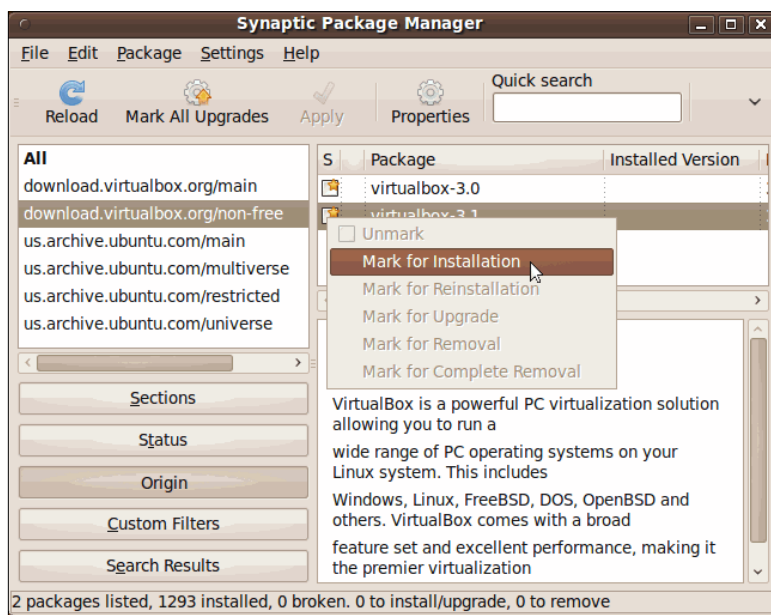
12. You can now delete the Sun public key file you downloaded earlier. Click on the **Close** button to return to the Synaptic Package Manager. If the **Repositories Changed** dialog shows up, select the **Never show this message again** checkbox, and click on **Close** to continue.
13. Now click on the Synaptic Package Manager's Reload button to update your package sources with the most recent VirtualBox version:



14. Once the Synaptic Package Manager finishes updating the package sources list, click on the **Origin** button located at the lower-left part of the window and select the `download.virtualbox.org/non-free` repository from the window above this button:

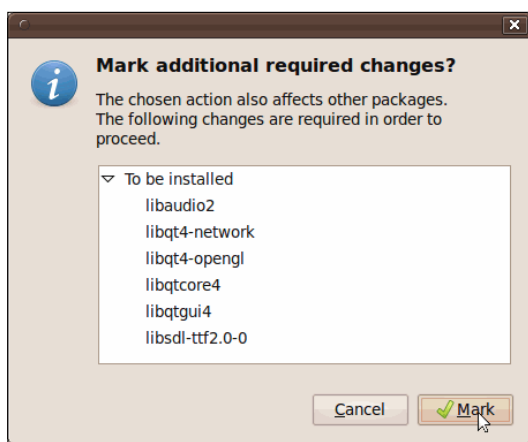


- 15.** Click on the most recent `virtualbox-3.X` package checkbox in the right window, and select the **Mark for Installation** option:

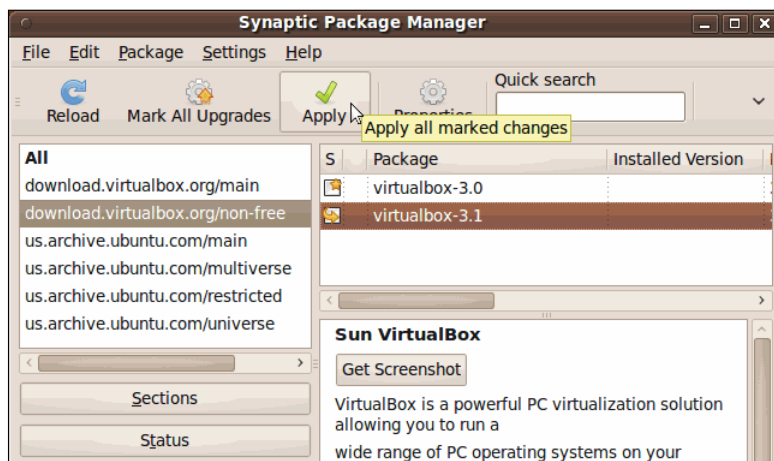


When upgrading to a newer VirtualBox version, you must first completely remove the older version. Then you'll be able to install the newest version without any hassles.

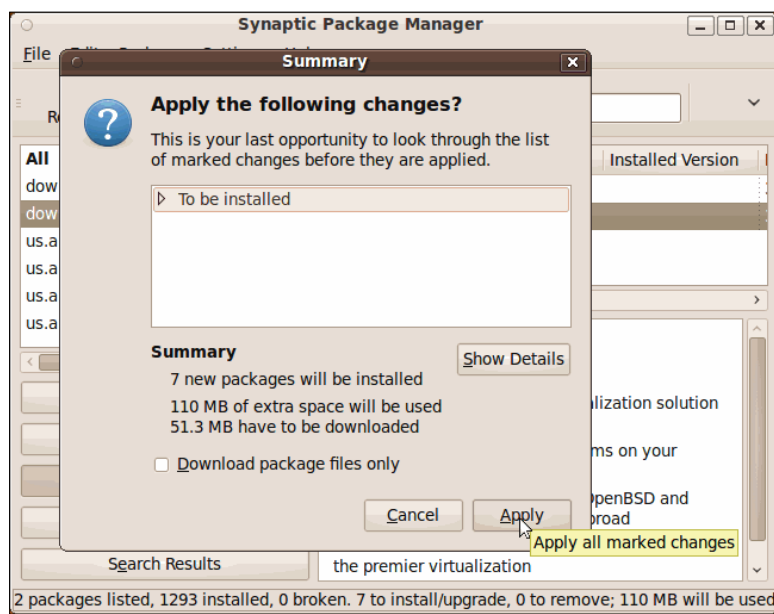
- 16.** The **Mark additional required changes?** dialog box will appear next. Click on the **Mark** button to mark all the additional packages required to install VirtualBox:



17. Now click on the **Apply** button in the Synaptic Package Manager:

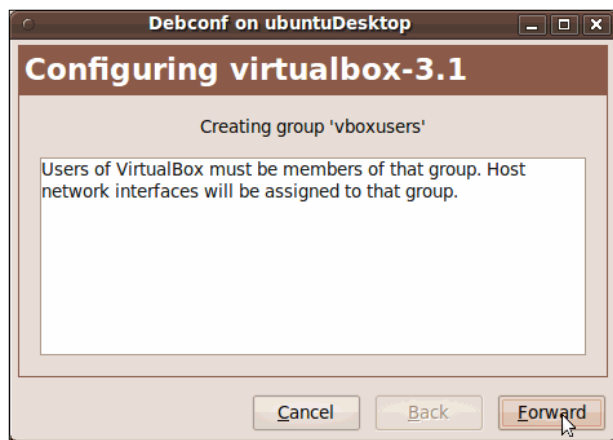


18. The **Apply the following changes?** dialog box will appear next. Make sure the **Download package files only** option is deselected, and click on the **Apply** button to start installing the required packages, along with VirtualBox:

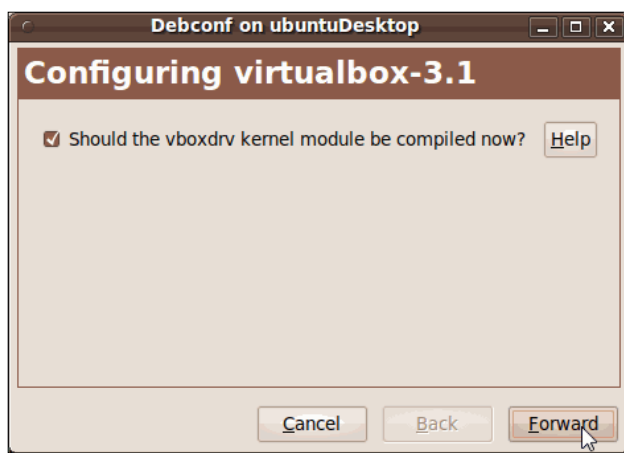


19. The Synaptic Package Manager will start downloading the required packages and, when finished, it will install them along with VirtualBox.

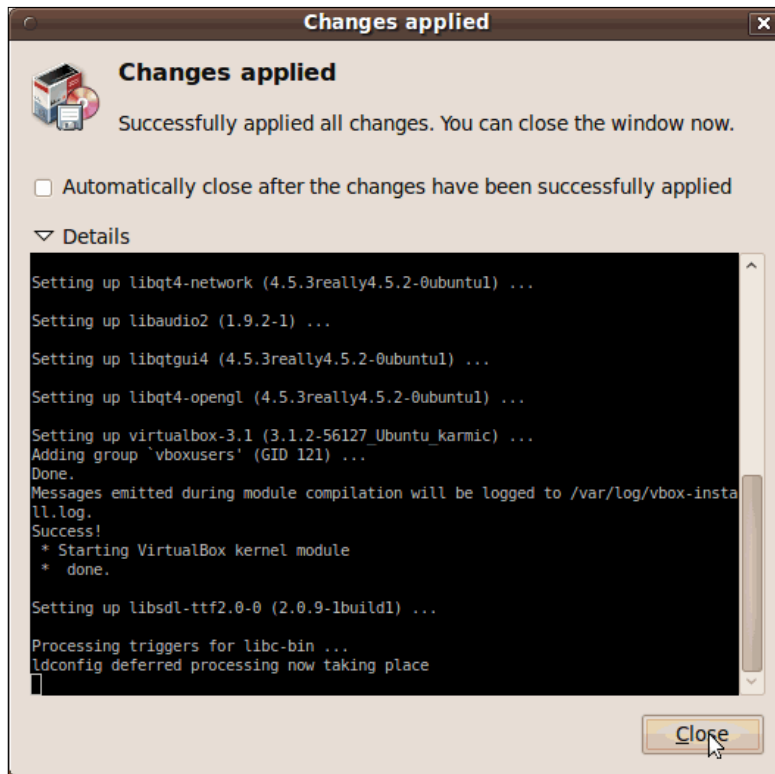
20. Eventually, the **Configuring virtualbox-3.X** dialog will appear:



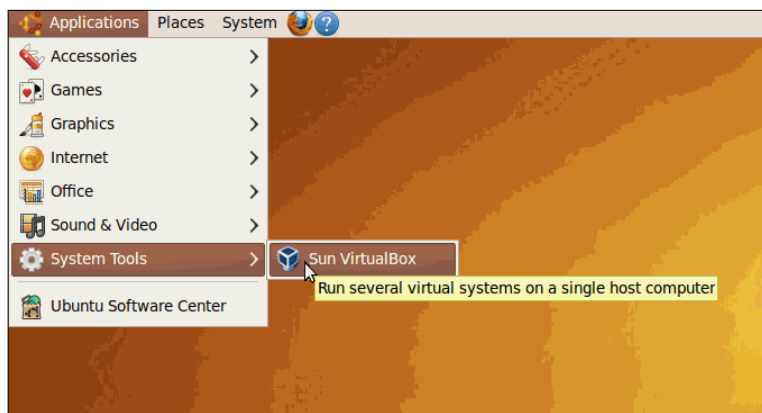
21. Click on **Forward** to continue configuring VirtualBox. The configuration wizard will ask if you want to compile the `vboxdrv` kernel module. Make sure that option is selected, and click on **Forward** to continue:



- 22.** Wait until **Debconf** compiles the kernel module and finishes all the configuration procedure for VirtualBox. After that, the following dialog will appear:



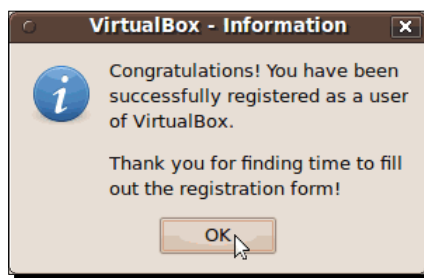
- 23.** This means VirtualBox installed successfully. Click on the **Close** button to continue and close the Synaptic Package Manager.
- 24.** Open a terminal window, type `sudo usermod -a -G vboxusers yourusername`, and hit *Enter* to add your username to the `vboxusers` group. Don't forget to replace *yourusername* with your own username. Then type `exit`, and hit *Enter* again to close the terminal window.
- 25.** Logout from Ubuntu, and then log back in. Now you can access VirtualBox through the Ubuntu menu bar (**Applications | System Tools | Sun VirtualBox**):



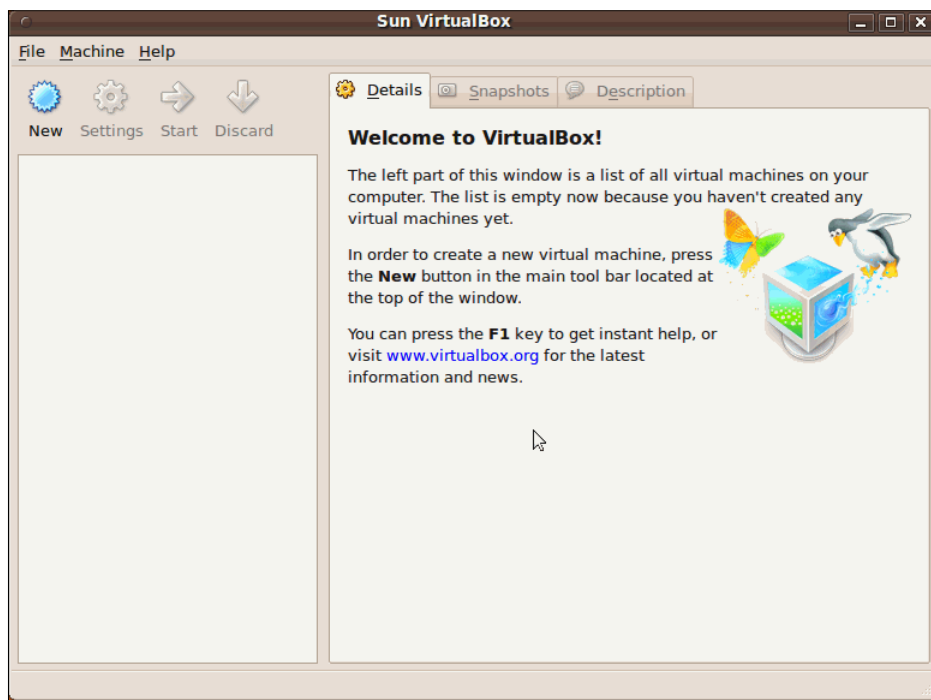
- 26.** After opening VirtualBox for the first time, you'll see the **VirtualBox License** dialog. Scroll down to the end of the agreement, and then click on the **I Agree** button to continue.
- 27.** Next you'll see the **VirtualBox Registration Dialog**:

A screenshot of the 'VirtualBox Registration Dialog' window. The window has a title bar with 'VirtualBox Registration Dialog' and a close button. The main content area is titled 'Welcome to the VirtualBox Registration Form!'. It contains a blue sidebar with a red balloon and an '@' symbol. The main text area says: 'Please fill out this registration form to let us know that you use VirtualBox and, optionally, to keep you informed about VirtualBox news and updates. Please use Latin characters only to fill in the fields below. Sun Microsystems will use this information only to gather product usage statistics and to send you VirtualBox newsletters. In particular, Sun Microsystems will never pass your data to third parties. Detailed information about how we use your personal data can be found in the Privacy Policy section of the VirtualBox Manual or on the Privacy Policy page of the VirtualBox web-site.' There are two radio buttons: 'I already have a Sun Online account:' and 'I would like to create a new Sun Online account:'. The second option is selected. Below the first option are fields for 'E-mail:' and 'Password:'. Below the second option are fields for 'First Name:', 'Last Name:', 'Company:', 'Country:' (a dropdown menu showing 'United States'), 'E-mail:', 'Password:', and 'Confirm Password:'. At the bottom right are 'Register' and 'Cancel' buttons. A mouse cursor is pointing at the 'Register' button.

- 28.** If you already have a Sun account, just fill in your email and password, and click on the **Register** button to register your copy of VirtualBox. If this is the first time you're using a Sun product, select the **I would like to register creating a new Sun Online account** radio button, and fill in the required fields. Then click on **Register**. Once you register with Sun, you'll see a success dialog:



- 29.** Now click on the **OK** button to go to the VirtualBox main screen:



- 30.** You can close VirtualBox now.

What just happened?

Ok, ok... Installing VirtualBox on Linux is a little more elaborated process, but hey, I was behind you all the time, right? On Linux there are some additional things to consider when you're installing a software application like VirtualBox. I decided to show you how to use the Synaptic Package Manager because this wonderful piece of open source software takes care of all the complicated things that always come up when installing something in Linux.

On the first three steps of the previous exercise, you upgraded the Ubuntu packages list and then installed a couple of packages required by VirtualBox on a Linux installation. These packages prepare your system for building the external modules VirtualBox needs to operate virtual machines. The `DKMS` (Dynamic Kernel Module Support) package allows your system to build these modules dynamically, so you don't have to worry in case you upgrade your Ubuntu system. The `build-essential` package contains the `gcc` compiler and some other tools required to compile and build the kernel modules. The Synaptic Package Manager uses repositories scattered throughout Internet. To install VirtualBox, we had to add a specific repository called `download.virtualbox.org/main` because the standard Ubuntu installation only includes the `virtualbox-ose` package, the **OSE (Open Source)** version of VirtualBox. In this book we're going to work with the **PUEL (Personal Use and Evaluation License)** version of VirtualBox because it's free for personal/academic use or for product evaluation.

Basically, the PUEL version of VirtualBox has some features that aren't included in the OSE version:

- ◆ **USB support:** You can insert USB devices on your host PC and use them in your virtual machines.
- ◆ **RDP (Remote Display Protocol) server:** The PUEL version of VirtualBox includes a complete RDP server, so you can connect to remote virtual machines from any RDP compatible client.
- ◆ **USB over RDP:** This means a virtual machine can access the USB devices connected to the remote computer on which the RDP client is running.

Finally, we also have to add the Sun public key because Synaptic won't be able to access the VirtualBox PUEL repository otherwise. After that, you just need to select the desired VirtualBox version, and Synaptic will take care of the rest for you.

Once you've installed VirtualBox on Linux, the process for creating and using virtual machines will be the same on any platform.

Have a go hero – experimenting with VirtualBox on Linux

If you have the ability and resources, would be a great idea to try VirtualBox on other Linux distros, but you need to make an extra effort because not every distribution uses the Synaptic Package Manager. But the VirtualBox website has some basic information on how to install VirtualBox on several Linux distros such as Debian, Fedora, Red Hat, or SuSE. The website provides several specific packages for some of the most popular Linux distributions and one general package for the rest of the Linux distros available. And remember you can always send me an email in case you get stuck!

Oh, and you can also start to browse around VirtualBox's user interface. Go on and experiment all you can and try to create a virtual machine by yourself!

Testing VirtualBox

Ok, you have VirtualBox installed and ready to go. Are we going to wait until Chapter 2 to start using this wonderful piece of software? Naah! Let's take it for a little test drive...

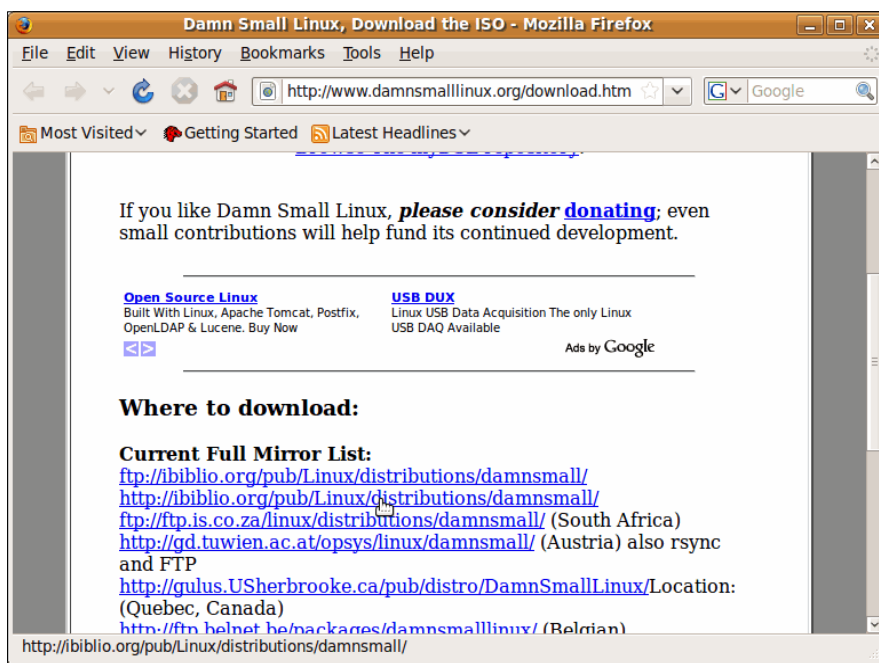
Time for action – creating and testing a Damn Small Linux virtual machine

In this next exercise, you'll download **Damn Small Linux**, one of the smallest Linux distros currently available, and then I'll show you how to create a quick virtual machine on your brand new VirtualBox software in order to test it!

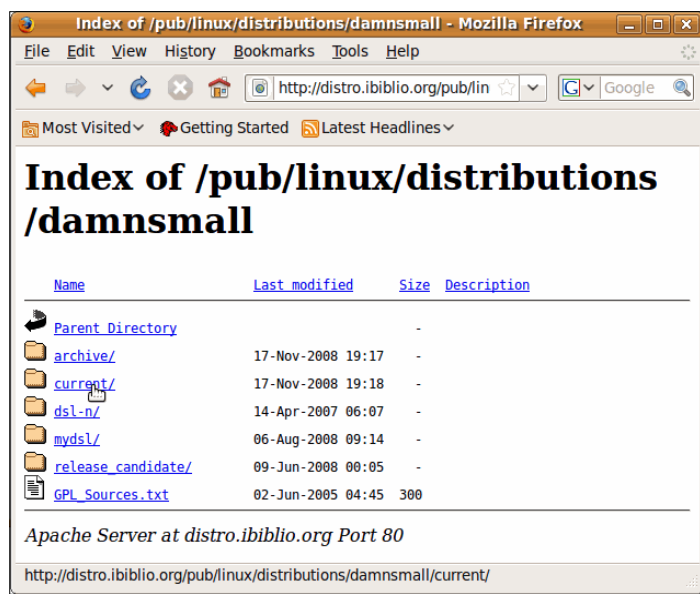
1. Open your web browser, and type `http://www.damnsmalllinux.org`. The **What is DSL?** page will show up:



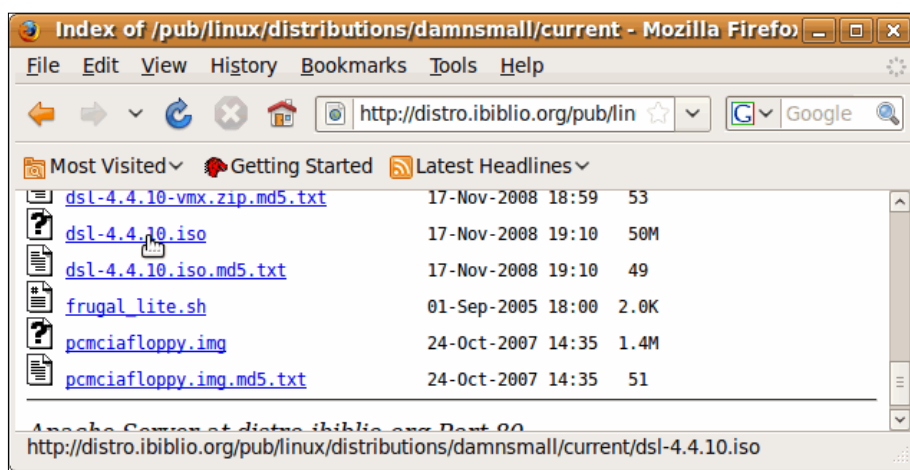
2. Scroll down the page until you locate the **Download** link, or go to the `http://www.damnsmalllinux.org/download.html` page directly, and scroll down to the **Current Full Mirror List** section:



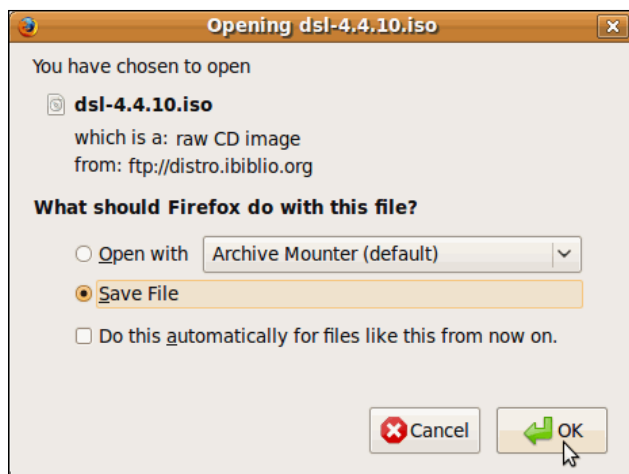
3. Click on an HTTP link (in this case, I chose the second link, <http://ibiblio.org/pub/Linux/distributions/damnsmall/>) and then on the **current** link of the next page:



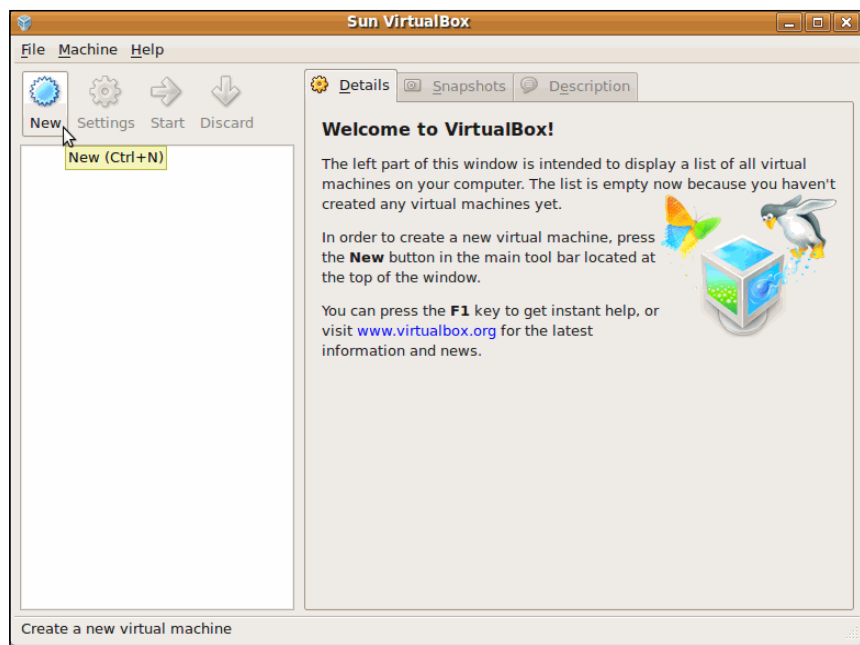
4. On the next page, scroll down until you locate the **dsl-4.4.10.iso**, and click on it to start downloading the DSL ISO file:



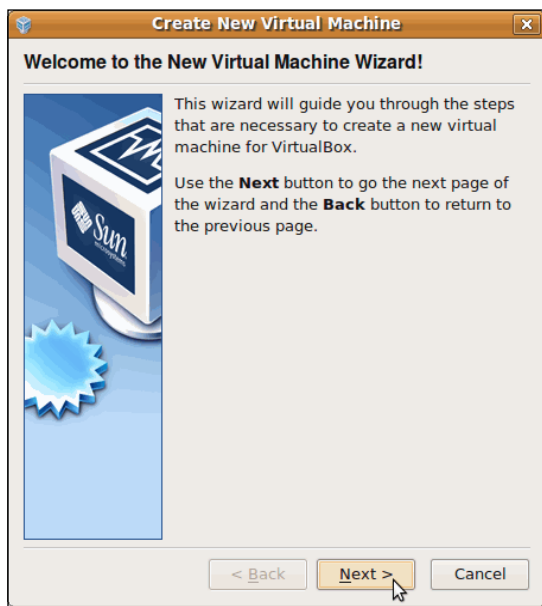
5. A dialog box will pop up, asking if you want to open or save the DSL ISO file. Click on **Save** to start downloading this file:



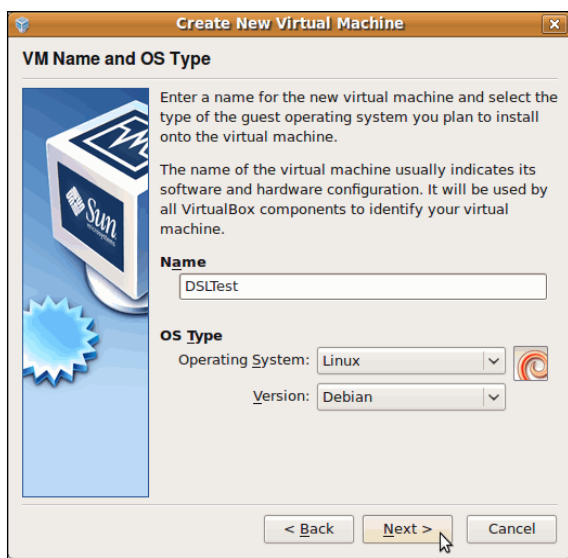
6. Once the downloading process ends, open VirtualBox, and click on the **New** button to create a new virtual machine:



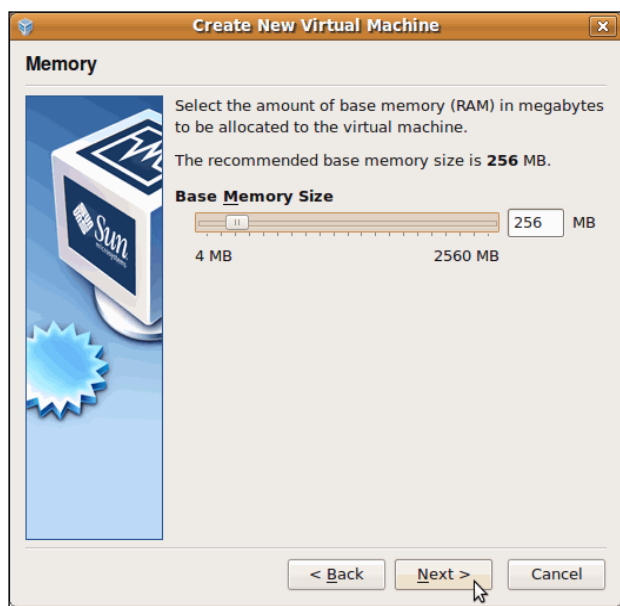
7. The **Welcome to the New Virtual Machine Wizard** dialog will show up. Click on **Next** to continue:



8. The **VM Name and OS Type** dialog will appear next. Type **DSLTest** in the **Name** field, select **Linux** as the **Operating System** and **Debian** as the **Version**, then click on **Next** to continue:



9. Leave the default 256 MB value as the **Base Memory Size** in the **Memory** dialog, and click on **Next** to continue:



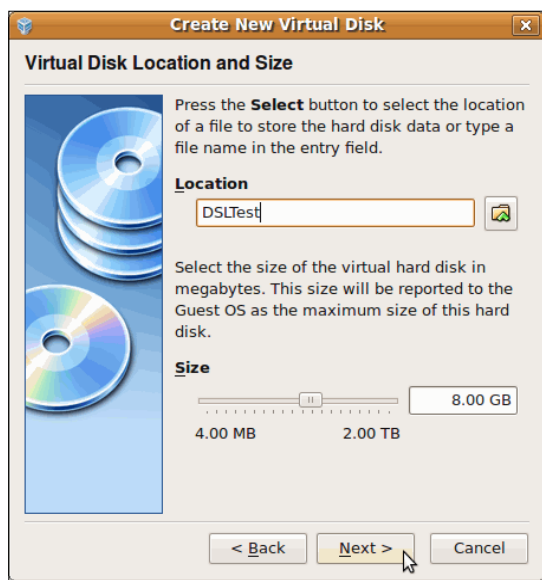
10. Leave the default **Create new hard disk** option on the **Virtual Hard Disk** dialog. Make sure the **Boot Hard Disk (Primary Master)** option is selected, and click on **Next** to continue:



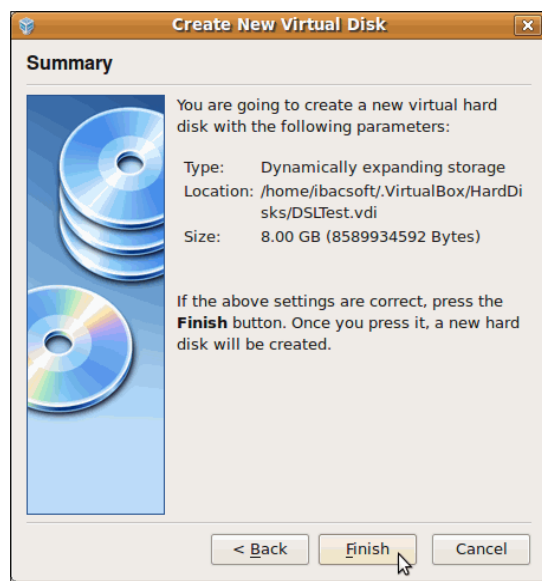
- 11.** Now the **Welcome to the Create New Virtual Disk Wizard!** dialog box will appear. Click on **Next** to continue, and leave the **Dynamically expanding storage** option selected as the **Storage Type** in the **Hard Disk Storage Type** dialog:



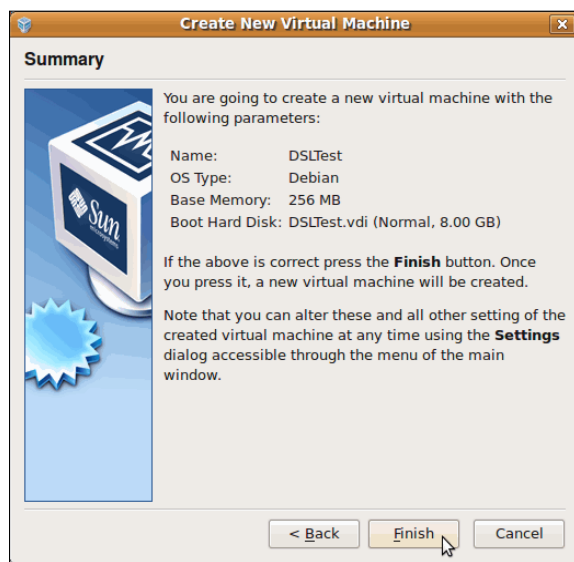
- 12.** In the next dialog (**Virtual Disk Location and Size**), leave the `DSLTest` and 8.00 GB default options for **Location** and **Size**, respectively, and click on **Next** to continue:



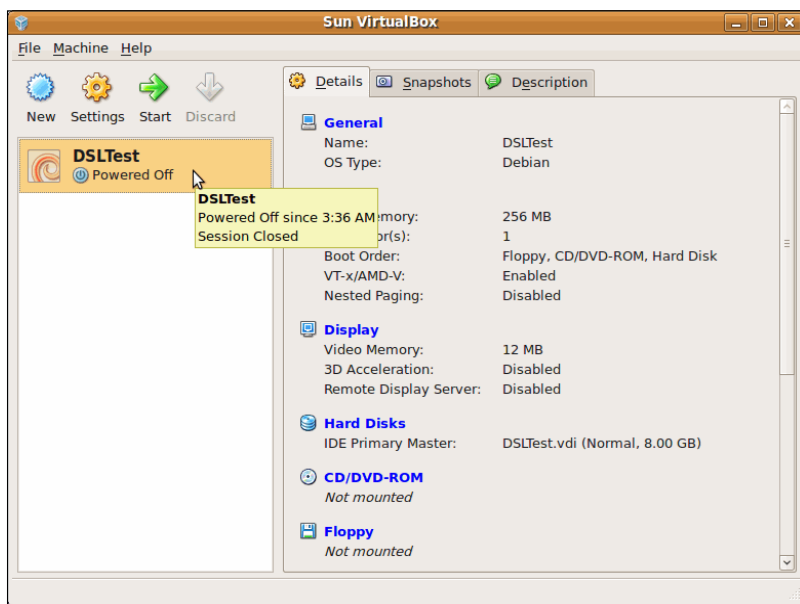
- 13.** Now the wizard will show you a **Summary** of all the parameters for the virtual disk of your virtual machine. Click on **Finish** to continue:



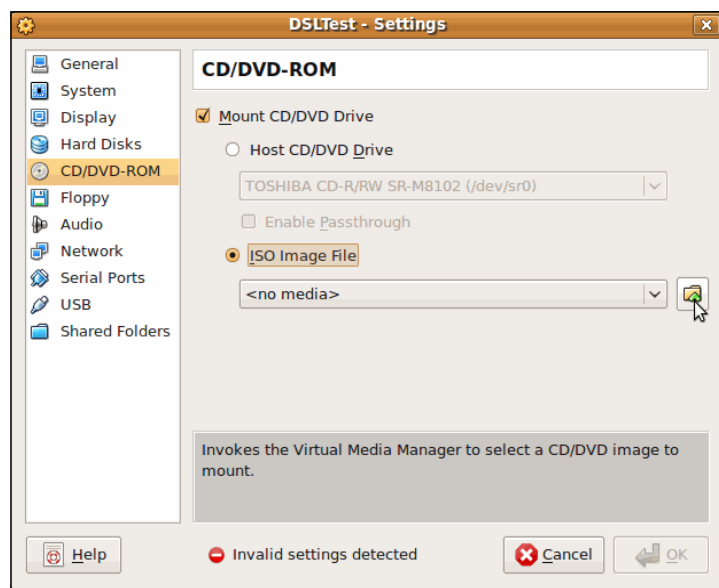
- 14.** The last dialog will show you a **Summary** of all the parameters selected for your new virtual machine. Click on **Finish** to continue:



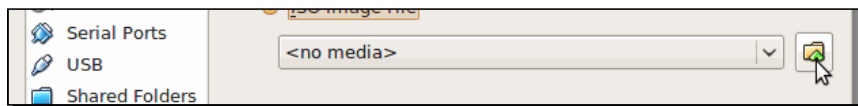
15. After creating the virtual machine, it will show up in VirtualBox's main screen:



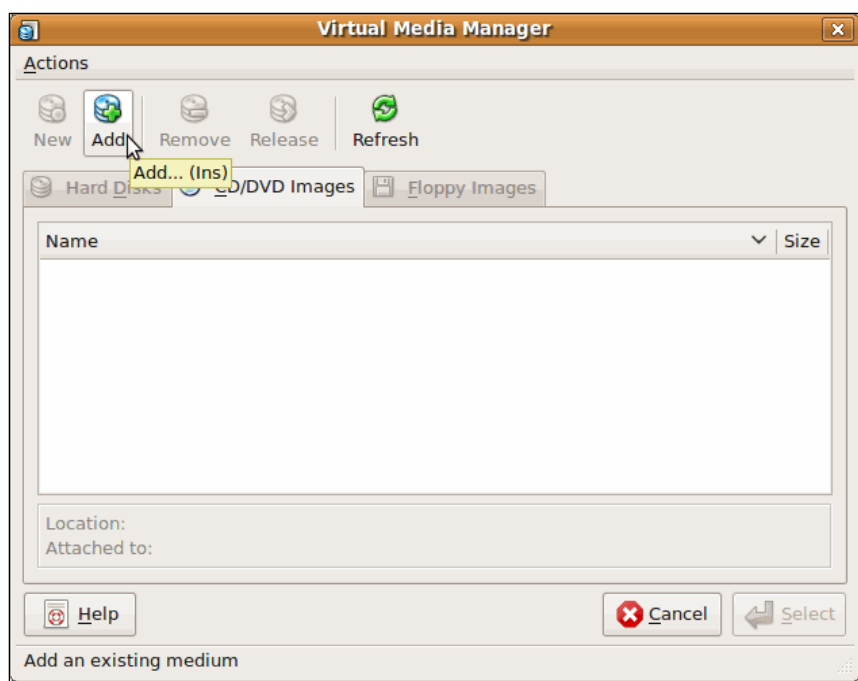
16. Now click on the **Settings** button to open the **DSLTest - Settings** dialog box. Select the **CD/DVD-ROM** category from the list on the left side, click on the **Mount CD/DVD Drive** checkbox, and then select the **ISO Image File** option:



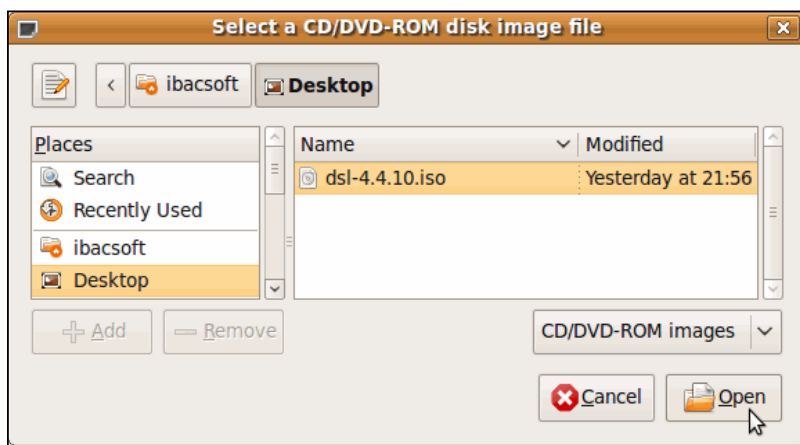
17. Next, click on the **Invoke Virtual Media Manager** button:



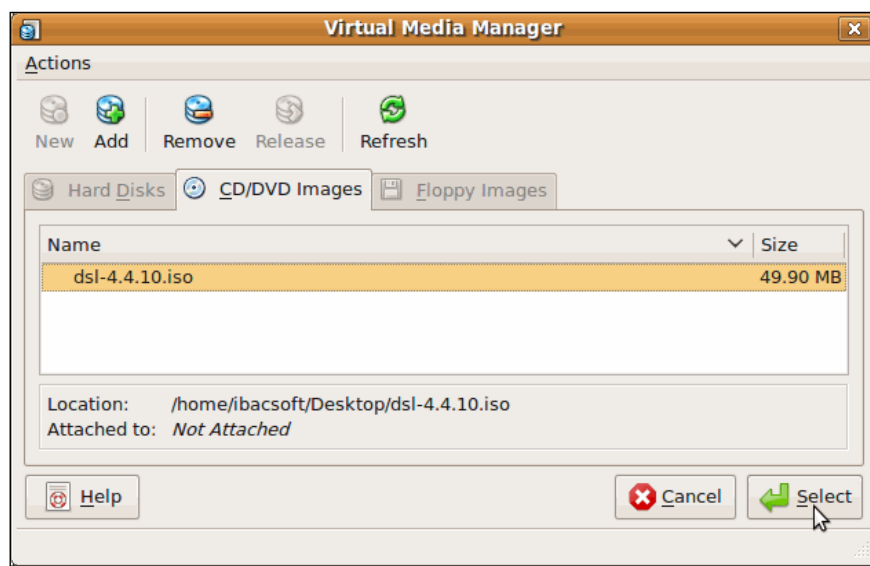
18. The **Virtual Media Manager** dialog will appear. Click on the **Add** button to continue:



- 19.** The **Select a CD/DVD-ROM disk image file** dialog box will show up next. Navigate to the directory where you downloaded the `dsl*.iso` image, and click on the **Open** button to continue:

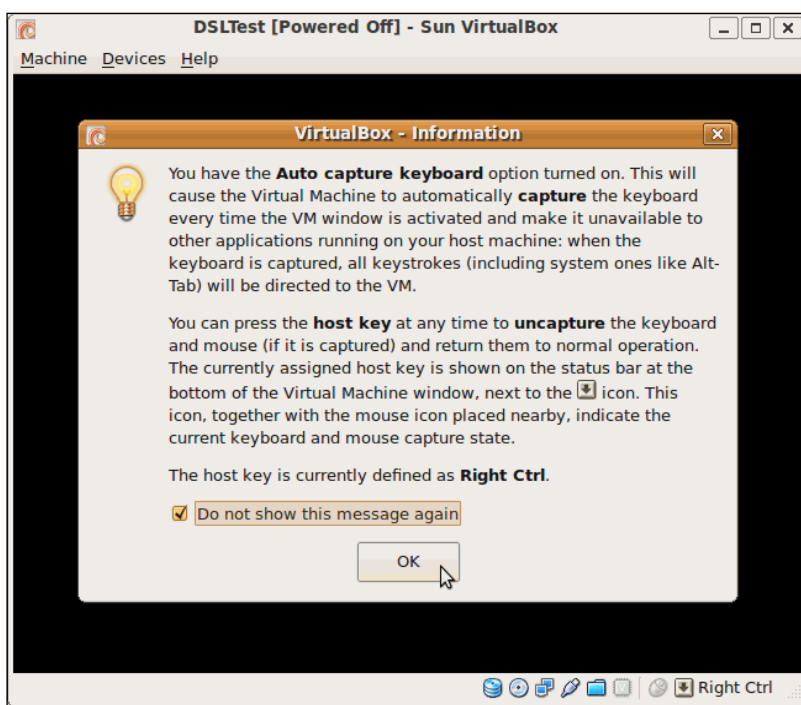


- 20.** Now the DSL ISO image will appear in the **CD/DVD Images** list of the Virtual Media Manager:



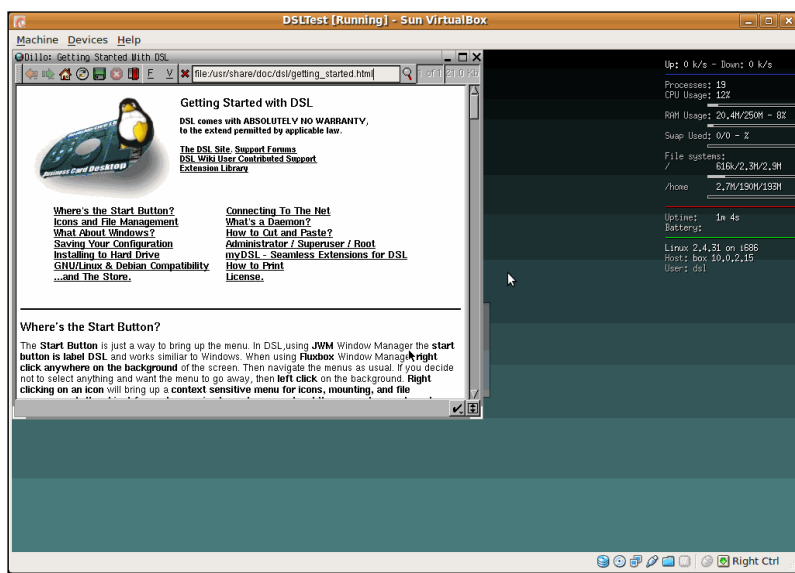
- 21.** Click on **Select** to return to the **DSLTest – Settings** dialog box, and then click on **OK** to return to VirtualBox's main screen.

22. Click on the **Start** button to start your virtual machine. The **VirtualBox – Information** dialog box will show up to inform you that you need to use the *Right-Ctrl* key to alternate between using the keyboard in your virtual machine and your host PC. Select the **Do not show this message again** checkbox, and click on **OK** to continue:



23. The DSL virtual machine will start to boot. Wait until a **VirtualBox – Information** dialog box appears to tell you that the virtual machine is using **16 bit** color instead of **32 bit**. Select the **Do not show this message again** checkbox, and click on **OK** to continue.
24. Click inside the virtual machine's screen with your mouse, and another **VirtualBox – Information** dialog will show up to inform you that you need to use the *Right-Ctrl* key to alternate between using the mouse in your virtual machine and your host PC. Select the **Do not show this message again** checkbox, and click on **Capture** to continue.

25. Wait for a few minutes for DSL to start booting up. When it is finished, the following screen will show up:



26. Now you have created your first virtual machine!

What just happened?

Whew! I must admit that this was a lengthy exercise! Thanks to VirtualBox's user-friendly interface, you've just learned how to create this test virtual machine with one of Linux's smallest distros available: **Damn Small Linux (DSL)** for short). In fact, it's so small, it doesn't even need a hard disk to work properly! I hope you had fun with this exercise, and don't worry about all the settings involved in creating a virtual machine; we'll have enough exercises to deal with them throughout the rest of the book!

For now, just concentrate on the virtual machine creation process as a whole: Click on the **New** button, select the operating system you want to use, assign a chunk of memory and hard disk space to your new virtual machine, and tell it where to boot an ISO image with the operating system of your choice, as if it were just another ordinary PC in your desk!

Have a go hero – creating more virtual machines

Now that you've created a test virtual machine, go on and download other Linux distributions, and try to create other virtual machines; experiment with different settings for RAM and hard disk space.



Just remember to never assign more than half of your physical RAM to a virtual machine because your main PC could collapse!

We'll talk more about this later. For now, just keep experimenting with all the VirtualBox features you can! And if you have any questions, feel free to email the Packt team at questions@packtpub.com!

Pop quiz – doing the thing

1. What would be the best definition of a virtual machine?
 - a. A physical PC connected to a LAN.
 - b. A 'guest' machine running inside another 'host' machine.
 - c. A computer with lots of RAM and hard disk space.
2. You need to use MS Word to write a document, but the only available PC at the moment is running Linux. The best thing you can do is:
 - a. Cry and pull your hair out in despair.
 - b. Install VirtualBox on your Linux PC, create a Windows XP virtual machine, and install MS Word.
 - c. Go out and buy a new Windows Vista PC with MS Word installed.
3. Virtualization means...
 - a. Sharing one physical PC between two or more virtual machines to maximize resource usage.
 - b. To surf the web looking for new applications.
 - c. Running two or more operating systems in one PC.

Summary

This chapter was a hands-on introduction to VirtualBox, the best virtualization software out there!

Specifically, we covered:

- ◆ What virtualization is, and how we can use VirtualBox to run several virtual machines inside a physical host
- ◆ How to install VirtualBox on Windows environments
- ◆ How to install VirtualBox on Linux environments
- ◆ How to create, configure, and run a test virtual machine

Now that you have VirtualBox installed and running, you're ready to delve into the VirtualBox world! In the following chapter, I'll show you how to create, configure, and run your own Ubuntu Linux virtual machine in any host operating system supported by VirtualBox!

2

Creating Your First Virtual Machine: Ubuntu Linux

The first time I used VirtualBox was to write a book about using the Apache Roller blog server in Linux, and because I needed to use my beloved MS Word software for the writing part, I had to struggle between a Windows XP and a Linux PC. When I installed VirtualBox on my Windows XP machine and tried a copy of Ubuntu Linux in a virtual machine, I was amazed! I could even use my Windows screen capture software to capture all the Linux screenshots for my book!

That's why I decided to use Ubuntu Linux for this chapter's exercises. In my humble opinion, it's the most friendly and fascinating Linux distribution for beginners.

In this chapter you shall:

- ◆ Create your first virtual machine in VirtualBox, using Ubuntu Linux
- ◆ Learn about your virtual machine's basic configuration
- ◆ Download and install Ubuntu Linux on your virtual machine
- ◆ Learn how to start and stop your virtual machine

So let's get on with it...

Getting started

In this chapter, you'll need to download the Ubuntu Linux Desktop Live CD. It's a pretty big download (700 MB approximately), so I'd recommend you to start downloading it as soon as possible. With a 4 Mbps Internet connection, you'd need approximately 1 hour to download a copy of Ubuntu Linux Desktop.

That's why I decided to change the order of some sections in this chapter. After all, action is what we're looking for, right?

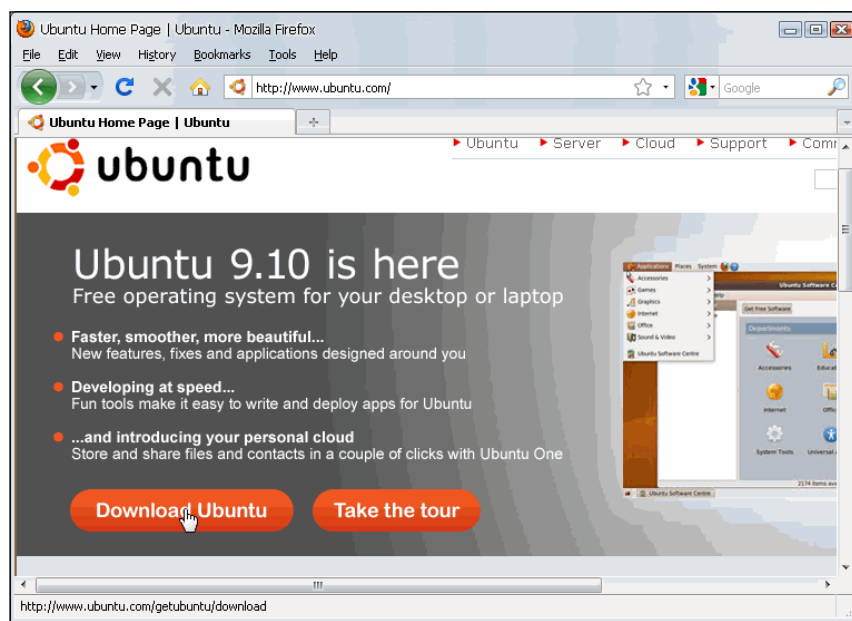
Downloading the Ubuntu Linux Live CD

After finishing the exercise in this section, you can jump straight ahead into the next section while waiting for your Ubuntu Live CD to download. That way, you'll have to wait for less time, and your virtual machine will be ready for some action!

Time for action – downloading the Ubuntu Desktop Live CD

In the following exercise I'll show you how to download the Ubuntu Linux Desktop Edition from the official Ubuntu website.

1. Open your web browser, and type `http://www.ubuntu.com` in the address bar. The **Ubuntu Home Page** will appear. Click on the Ubuntu Desktop **Download** link to continue:



2. The **Download Ubuntu** page will show up next. Ubuntu's most recent version will be selected by default (**Ubuntu Desktop 9.10**, at the time of this writing). Select a location near you, and click on the **Begin download** button to continue.



The Ubuntu Download page lets you download the 32-bit version automatically. If you want the 64-bit version of Ubuntu 9.10, you'll need to click on the **Alternative download options** link below the **Download locations** list box. The exercises in this chapter use the 32-bit version.

3. You'll be taken to the download page. After a few seconds, your download will start automatically. Select the **Save File** option in your browser, and click on **OK** to continue.
4. Now you just have to wait until the download process finishes.

What just happened?

I think the exercise pretty much explains itself, so I just want to add that you can also order a free Ubuntu CD or buy one from Ubuntu's website. Just click on the **Get Ubuntu** link at the front page and follow the instructions. I ordered mine when writing this chapter to see how long it takes to arrive at my front door. I hope it arrives before I finish the book!

Have a go hero – doing more with the thing

You can try downloading other Ubuntu versions, like Ubuntu 8.10 Hardy Heron. Just click on the **Alternative download options** link below the **Download locations** list box, and explore the other options available for download.

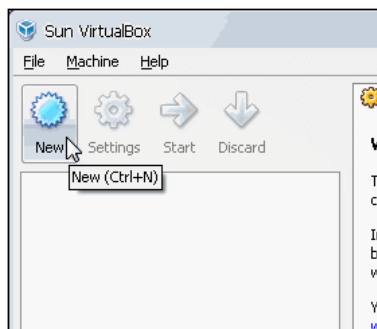
Creating your Ubuntu Linux VM

Now that you installed VirtualBox, it's time to learn how to work with it. I've used other virtualization products such as VMware besides VirtualBox, and in my humble opinion, the user interface in VirtualBox is a delight to work with.

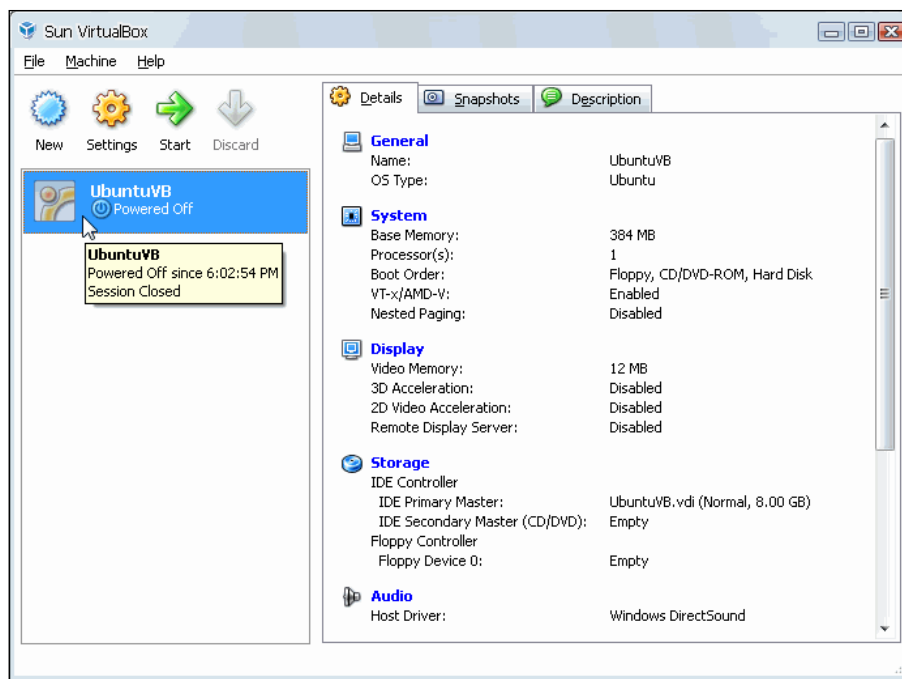
Time for action – creating a virtual machine

At last you have the chance to use Windows and Linux side by side! This is one of the best features VirtualBox has to offer when you want the best of both worlds!

1. Open VirtualBox, and click on the **New** button (or press *Ctrl+N*) to create a new virtual machine:



2. The **Welcome to the New Virtual Machine Wizard!** dialog will show up. Click on **Next** to continue. Type **UbuntuVB** in the **Name** field, select **Linux** as the **Operating System** and **Ubuntu** as the **Version** in the **VM Name and OS Type** dialog.
3. Click on **Next** to continue. You can leave the default 384 MB value in the **Memory** dialog box or choose a greater amount of RAM, depending on your hardware resources.
4. Click on **Next** to continue. Leave the default values in the **Virtual Hard Disk** dialog, and click on **Next** twice to enter the **Create New Virtual Disk** wizard. Leave the default **Dynamically Expanding Storage** option in the **Hard Disk Storage Type** dialog, and click on **Next** to continue.
5. Leave the default values chosen by VirtualBox for your Ubuntu Linux machine in the **Virtual Disk Location and Size** dialog (**UbuntuVB** and **8.00 GB**), and click on **Next** to continue.
6. A **Summary** dialog will appear, showing all the parameters you chose for your new virtual hard disk. Click on **Finish** to exit the **Create New Virtual Hard Disk** wizard. Now another **Summary** dialog will appear to show the parameters you chose for your new virtual machine. Click on **Finish** to exit the wizard and return to the VirtualBox main window:



7. Your new UbuntuVB virtual machine will appear on the VirtualBox main screen, showing all its parameters in the **Details** tab.

What just happened?

Now your Ubuntu Linux virtual machine is ready to run! In this exercise, you created a virtual machine with all the parameters required for a typical Ubuntu Linux distribution. The first parameter to configure was the **base memory** or **RAM**. As you saw in step 3, the recommended size for a typical Ubuntu Linux installation is 384 MB.



Exercise caution when selecting the RAM size for your virtual machine. The golden rule of thumb dictates that, if you have less than 1 GB of RAM, you can't assign more than one half of your physical RAM to a virtual machine because you'll get into trouble! Your **host** PC (the one that runs VirtualBox) will start to behave erratically and could crash!

With 2 GB, for example, you can easily assign 1 GB to your virtual machine and 1 GB to your physical PC without any problems. Or you could even have three virtual machines with 512 MB of RAM each and leave 512 MB for your host PC. The best way to find out the best combination of RAM is to do some experimenting yourself and to know the minimum RAM requirements for your host and guest operating systems.



Don't assume that assigning lots of RAM to your virtual machine will increase its performance. If, for example, you have 2 GB of RAM on your host and you assign 1 GB to an Ubuntu virtual machine, it's very unlikely there will be a bigger performance increase than if you were assigning only 512 MB to the same Ubuntu virtual machine. On the contrary, your host PC will work better if you only assign 512 MB to the Ubuntu virtual machine because it will use some of the extra RAM for disk caching, instead of recurring to the physical hard disk. However, it all depends on the guest operating system you plan to use and what applications you need to run.

If you look closely at the **Base Memory Size** setting in the **Memory** dialog when creating a virtual machine, you'll notice that there are three memory areas below the slider control: the green-colored area indicates the amount of memory range you can safely choose for your virtual machine; the yellow-colored area indicates a dangerous memory range that you can choose, but nobody knows if your virtual machine and your host will be able to run without any problems; and the red-colored area indicates the memory range that your virtual machine can't use. It's wise to stick with the default values when creating a new virtual machine. Later on, if you need to run a memory-intensive application, you can add more RAM through the **Settings** button, as we'll see in the following section.

Another setting to consider (besides memory) when creating a virtual machine is the **virtual hard drive**. Basically, a virtual hard disk is represented as a special file on your host computer's physical hard disk, and it's commonly known as a disk image file. This means that your host computer sees it as a 'large' file on its system, but your virtual machine sees it as a 'real' hard disk connected to it. Since virtual hard drives are completely independent from each other, there is no risk of accidental overwriting, and they can't be larger than the free space available on your real computer's physical hard drive. This is the most common way to handle virtual storage in VirtualBox; later on, we'll see more details about disk image files and the different formats available.

VirtualBox assigns a default value based on the guest operating system you plan to install in your virtual machine. For Ubuntu Linux, the default value is 8 GB. That's enough space to experiment with Ubuntu and learn to use it, but if you really want to do some serious work—desktop publishing or movie production, for example—you should consider assigning your virtual machine more of your hard disk space.

You can even get two hard drives on your physical machine, and assign one for your host system and the other one for your virtual machine! Or you can also create a new virtual hard disk image and add it as if it were a second hard drive! There are a lot of possibilities; we'll explore all we can about virtual hard disks in a later chapter, so don't worry about it for now.

Before going to the next section, I'd like to talk about the two virtual hard disk storage types available in VirtualBox: **dynamically expanding storage** and **fixed-size storage**. The **Hard Disk Storage Type** dialog you saw in step 4 of the previous exercise contains a brief description for both types of storage. At first, the dynamically expanding option might seem more attractive because you don't see the space reduction in your hard drive immediately.

When using dynamically expanding storage, VirtualBox needs to expand the storage size continuously, and that could mean a slight decrease in speed when compared to a fixed-size disk, but most of the time this is unnoticeable. Anyway, when the virtual hard disk is fully expanded, the differences between both types of storage disappear.

Most people agree that a dynamically expanding disk represents a better choice than a fixed one, since it doesn't take up unnecessary space from your host's hard disk until needed. Personally, when experimenting with a new virtual machine, I use the dynamically expanding option, but when doing some real work, I like to set apart my virtual machine's hard disk space from the beginning, so I choose the fixed-size storage option in these cases.

Have a go hero – experimenting with memory and hard disk storage types

When creating a virtual machine, you can specify the amount of RAM to assign to it instead of using the default values suggested by VirtualBox. Create another virtual machine named `UbuntuVB2`, and try to assign the entire RAM available to it. You won't be able to continue until you select a lower value because the **Next** button will be grayed out, which means you're in the red-colored memory range. Now move back the slider until the **Next** button is active again; you'll probably be in the yellow-colored memory range. See if your virtual machine can start with that amount of memory and if you can use both your host PC and your VM without any problems. In case you encounter any difficulties, keep moving back the memory range until all problems disappear.

Once you're done experimenting with the memory setting, use the `UbuntuVB2` virtual machine with the same exact settings as the one you created in the previous exercise, but this time use a fixed-size hard drive. Just take into account that since VirtualBox must prepare all the storage space at once, this process may take a long time depending on the storage size you selected and the performance of your physical hard disk. Now go and try out different storage sizes with both types of disks: dynamically expanding and fixed size.

Pop quiz – creating virtual machines

1. What happens when you create a new virtual hard disk on VirtualBox?
 - a. The virtual hard disk takes up all the storage space on your physical hard disk.
 - b. The virtual hard disk appears like a 'large' file on your real computer's file system.
 - c. The virtual hard disk downloads the guest operating system automatically.
2. Can you have three 10-GB virtual hard disks on a 160-GB empty physical hard drive?
 - a. Yes, because the physical hard drive has more storage space than the three virtual hard disks could take up.
 - b. No, because the physical hard drive can only have one virtual hard disk.
 - c. No, because the physical hard drive has more storage space than the three virtual hard disks could take up.
3. What happens if you have two virtual machines in the same host PC with a 200-GB hard drive and each virtual machine has a 20-GB virtual hard drive?
 - a. The virtual hard drives will overwrite each other.
 - b. The virtual hard drives will be isolated from each other.
 - c. You can only create one virtual hard drive per host PC.
4. You have 1 GB of RAM on your host PC, and you want to create three virtual machines with 512 MB each. What would happen if you run the three VMs at the same time?
 - a. You will only be able to start the first virtual machine because your host PC needs at least half of the total RAM free.
 - b. The three virtual machines will start and run flawlessly.
 - c. You'll only be able to start the first two virtual machines because each one needs 512 MB of RAM.

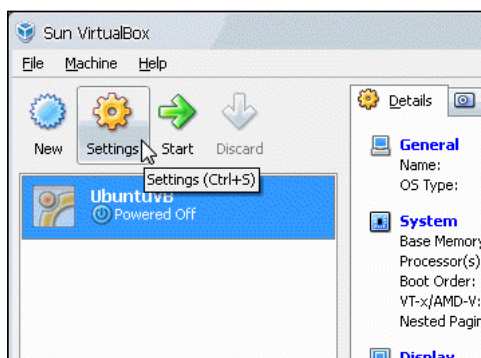
Configuring basic settings for your Ubuntu Linux VM


All right, you created your Ubuntu virtual machine and downloaded a copy of Ubuntu Desktop Live CD. Can you start installing Ubuntu now? I know you'll hate me, but nope, you can't. We need to tell your virtual machine where to boot the Live CD from, as if we were using a real PC. Follow me, and I'll show you the basic configuration settings for your VM, so you can start the Ubuntu installation ASAP!

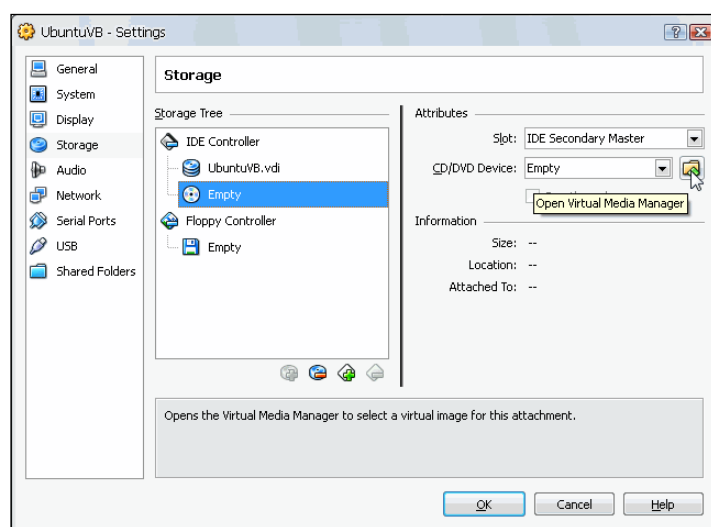
Time for action – basic configuration for your VM

In this exercise you'll learn how to adjust some settings for your virtual machine, so you can install Ubuntu Linux on it.

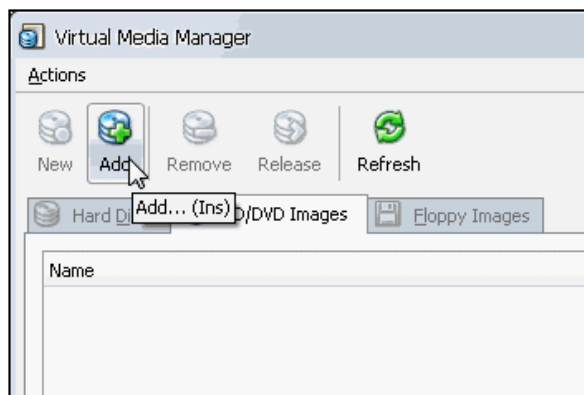
1. Open VirtualBox, select your `UbuntuVB` virtual machine, and click on the **Settings** button:



2. The **UbuntuVB – Settings** dialog will appear, showing all the settings in the **General** tab.
3. Click on the **Storage** category from the list in the left panel. Then select the **Empty** slot located just below the `UbuntuVB.vdi` hard disk image, under the **IDE Controller** element inside the **Storage Tree** panel, and click on the **Invoke Virtual Media Manager** button ():



4. The **Virtual Media Manager** dialog will appear next. Click on the **Add** button to add the **Ubuntu Linux Live CD ISO** image:



5. The **Select a CD/DVD-ROM disk image file dialog** will show up next. Navigate to the directory where you downloaded the Ubuntu Desktop ISO image, select it, and click on the **Open** button to continue.
6. The Ubuntu Desktop ISO image will appear selected in the **CD/DVD Images** tab from the **Virtual Media Manager** dialog. Click on the **Select** button to attach the Ubuntu ISO image to your virtual machine's CD/DVD drive.
7. Next, the Ubuntu ISO image file will appear selected on the **ISO Image File** setting from the **UbuntuVB – Settings** dialog. Click on **OK** to continue.
8. Now you're ready to start your virtual machine and install Ubuntu!

What just happened?

As you can see from the previous exercise, your virtual machine is just like a real PC. You need to 'insert' a bootable CD so that the VM can boot and start installing Ubuntu. In this case, we used an ISO image of Ubuntu Desktop instead of having to insert a real bootable CD. That's one of the benefits of VirtualBox: you can forget about having to burn a CD to test the new Ubuntu version or any other operating system! You just need to download the ISO image and configure your virtual machine to act as if there was a real CD inserted in the CD-ROM drive!

You can also go the traditional way, in case you have a DVD or CD ready to install your Ubuntu Desktop system. Just insert your DVD/CD, click on your virtual machine's **Settings** button, go to the **CD/DVD-ROM** category, and select the **Host CD/DVD drive** option instead of **ISO Image File**.

Whichever option you choose, your virtual machine will start to boot like a real physical PC! In the following section you'll get to see it with your own eyes...

Pop quiz – configuring basic settings on your VMs

1. What do you need to install a guest operating system on a virtual machine?
 - a. A big virtual hard disk image in your physical hard disk.
 - b. An ISO image or a bootable CD of the guest operating system.
 - c. Another real PC to install the new guest operating system on.
2. What is the job of the Virtual Manager in VirtualBox?
 - a. Create a new virtual machine.
 - b. Manage virtual hard disks and guest ISO images.
 - c. Manage networking issues.
3. Can you insert a real CD to install an operating system in your virtual machine?
 - a. No, because you can only use ISO images to install guest operating systems.
 - b. No, because you can't use the physical CD/DVD drive of your host PC with your Virtual machine.
 - c. Yes, because you can use the physical CD/DVD drive of your host PC as a CD/DVD Drive in your virtual machine.

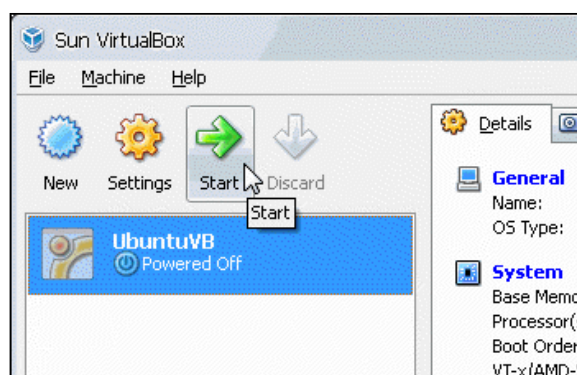
Installing Ubuntu Linux on your VM

Now everything's ready to start installing Ubuntu! At last you're about to see this wonderful piece of software in action...

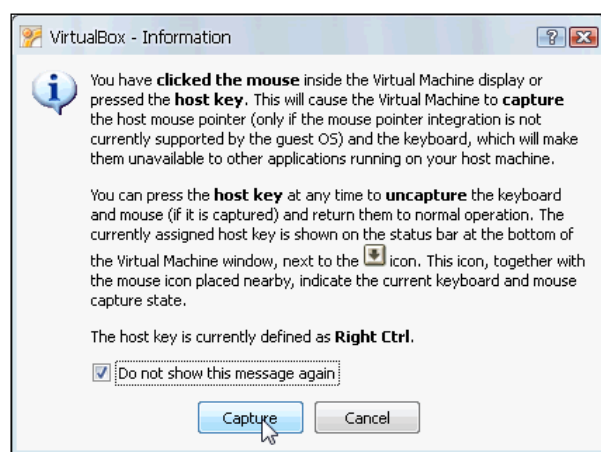
Time for action – installing Ubuntu Desktop on your VM

Ok, it's time to try out our new toy! Get ready for the ride of your life...


1. Click on the **Start** button to start your virtual machine:



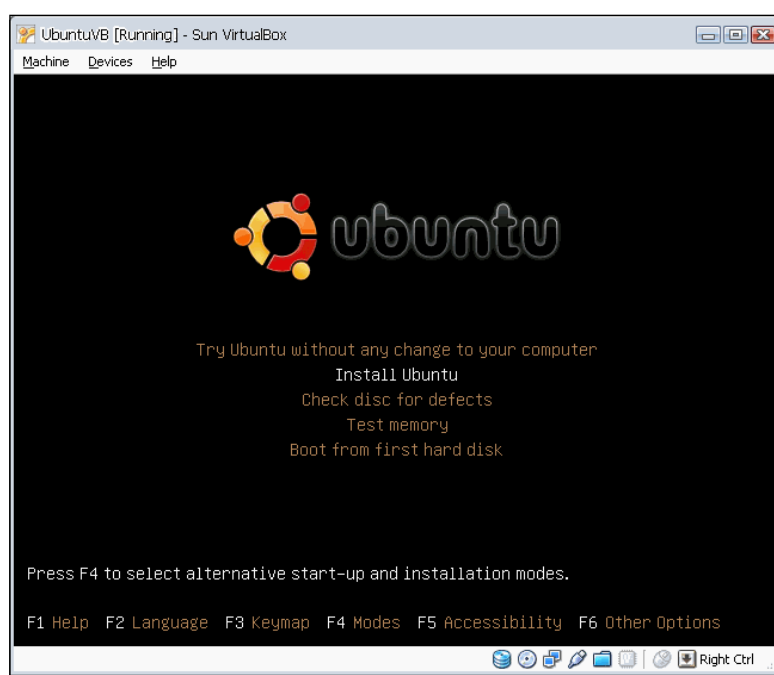
2. An Information dialog will appear to tell you that the **Auto capture keyboard** option is turned on. Enable the **Do not show this message again** option, and click on **OK** to continue.
3. Another Information dialog will show up to tell you about the color depth of your virtual machine. Enable the **Do not show this message again** option, and click on **OK** to continue.
4. The Ubuntu menu screen will show up, along with a pop-up menu where you can select the language of installation. Click anywhere inside the virtual machine screen, and the following information dialog will pop up:



5. Enable the **Do not show this message again**, and click on **Capture** to continue. The mouse pointer will disappear, and you will be able to move along the language menu. Select the **English** option, and press *Enter* to continue.

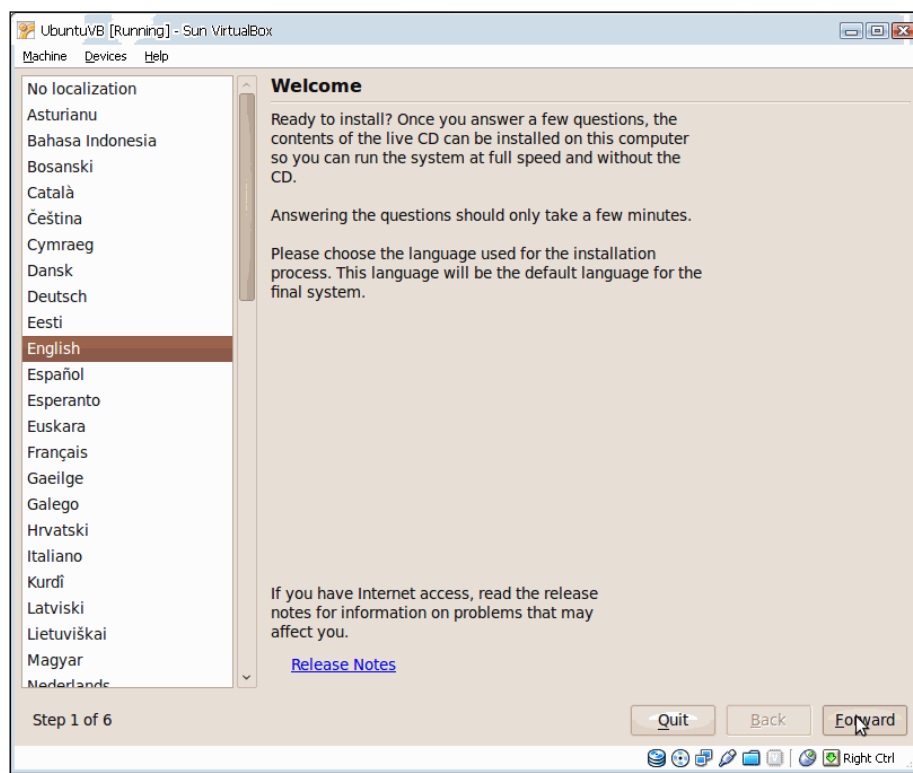
[ Remember you can press *Right-Ctrl*, the default host key, at any time to uncapture the keyboard and mouse so you can move your mouse pointer out of the virtual machine screen.]

6. Now select the **Install Ubuntu** option on the Ubuntu start-up menu, and press *Enter*:



7. Ubuntu will start to boot and, depending on your hardware, it will take some time to boot up completely. A red bar below the Ubuntu logo will appear to show the progress of the booting process (the bar should turn completely yellow when it's finished).

8. At the end of the Ubuntu booting process, the **Welcome** screen will appear, as shown below:



9. Select the language you want to use for the installation process (in this exercise, we'll stick with **English**). Click on **Forward** to continue. Select your **Region** and **City** in the **Where are you?** screen, and click on **Forward** to continue.
10. The **Keyboard layout** screen will appear next. You can leave the default option if you're using a typical QWERTY keyboard or choose an appropriate layout for your keyboard. To be sure the layout you chose works with your keyboard, type something into the test box. Click on **Forward** when you're ready to continue the installation process.
11. The **Prepare disk space** screen will appear next. You can leave the default setting (Use the entire disk) to let Ubuntu prepare the disk for you. Click on **Forward** to continue.

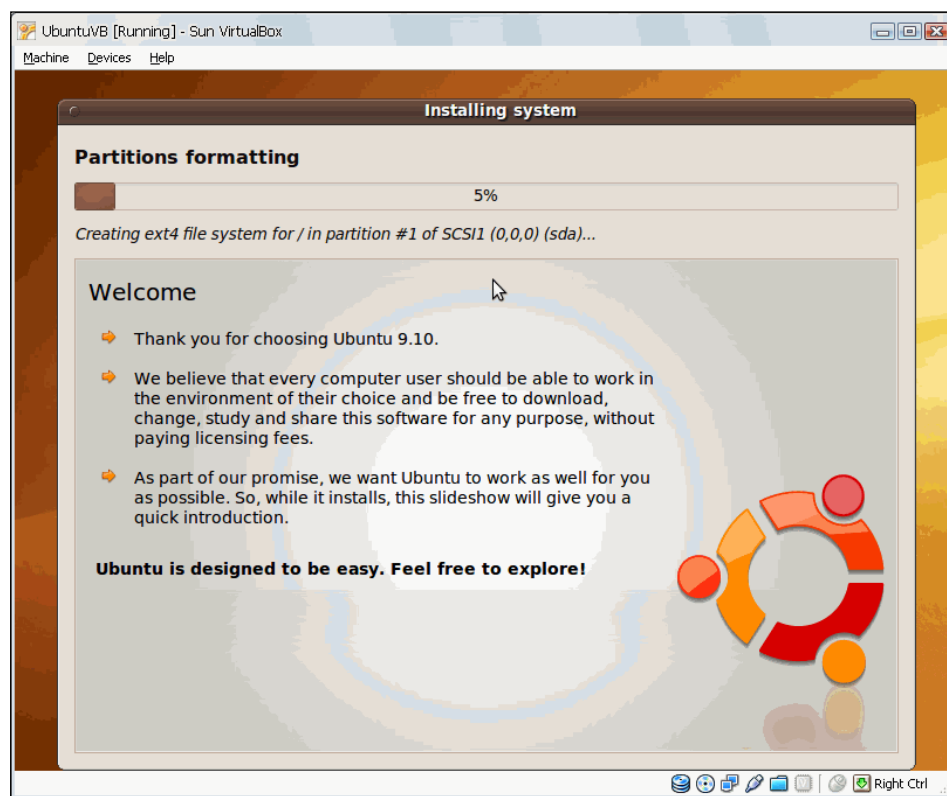


Remember you're using a virtual hard disk and not the physical hard drive of your real PC; it's perfectly ok to use the entire virtual hard disk for your Ubuntu guest operating system; the rest of the files in your real hard drive won't be affected.

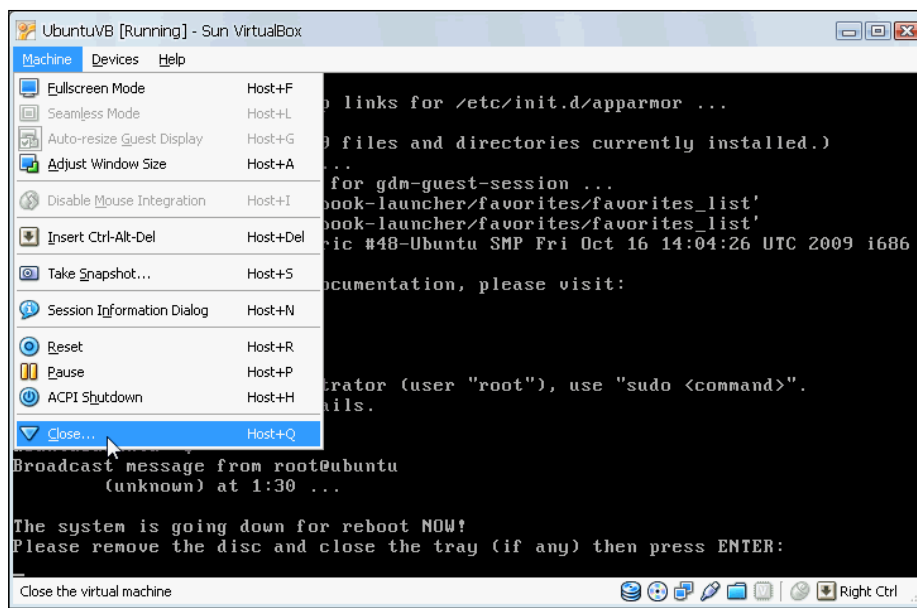
- 12.** On the next screen (**Who are you?**), you need to fill in your name, username, password, and your computer's name. You can also choose to login automatically or require a password to login. Use the following screenshot as a guide, but remember to replace my personal info with yours! Click on **Forward** when you're ready to continue:

- 13.** The **Ready to Install** screen will show up next with a summary of all the settings you chose for the Ubuntu installation. Click on **Install** when you're ready to begin the installation process.

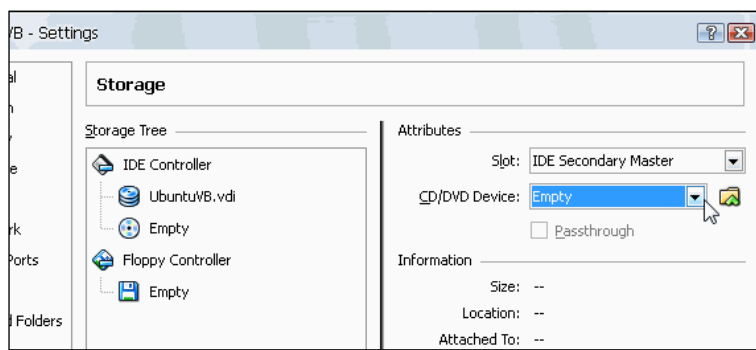
- 14.** Ubuntu will start to install in your virtual machine. An **Installing System** dialog will show you the installation progress:



- 15.** After some time (depending on your hardware speed), the **Installation Complete** dialog will appear. Click on **Restart Now** to exit the installer.
- 16.** Eventually, the Ubuntu logo will show up. In a real PC, you would have to remove the installation disc and press *Enter*. In this case, you need to shutdown the virtual machine and change the CD-DVD Rom drive setting. Hit the *Right-Ctrl* key to *uncapture* the mouse so you can move it to select the **Machine | Close** option from the VirtualBox main menu:



17. The **Close Virtual Machine** dialog will appear next. Select the **Power off the machine** option, and click on **OK** to continue.
18. VirtualBox will shutdown the virtual machine, and you'll return to the main screen. Now click on the **Settings** button to open the **UbuntuVB – Settings** dialog, go to the **Storage** category, select the `ubuntu-9.10-desktop-i386.iso` image in the **Storage Tree** panel; then click on the **CD/DVD Device** list box, and select the **Empty** option to remove the Ubuntu ISO image:



19. Click on **OK** to close the **UbuntuVB – Settings** dialog. Your virtual machine is now ready to start Ubuntu Linux for the first time!

What just happened?

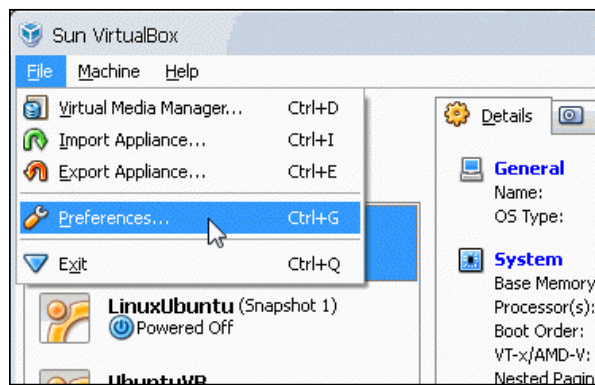
Ok, you mixed up all the ingredients, baked the mix, and now the cake is ready to eat! Sorry about the analogy, but I was hungry at the time I wrote this!

Personally, I think this is one of the coolest things I've seen in the last few years... Installing a completely standalone guest operating system inside a host operating system with just a few clicks... Whew! Now we'll really start to squeeze all the juice out of our hardware! But before jumping to the next exercise, there are a few things I want to talk about.

Every time you start a virtual machine in VirtualBox, it has to share the keyboard and the mouse with your real PC. The **Auto capture keyboard** feature lets your virtual machine 'capture' all the keyboard action automatically every time you activate its window, and it's enabled by default on every new virtual machine. As you saw in step 2, an information dialog shows up when starting a virtual machine for the first time to tell you about this feature.

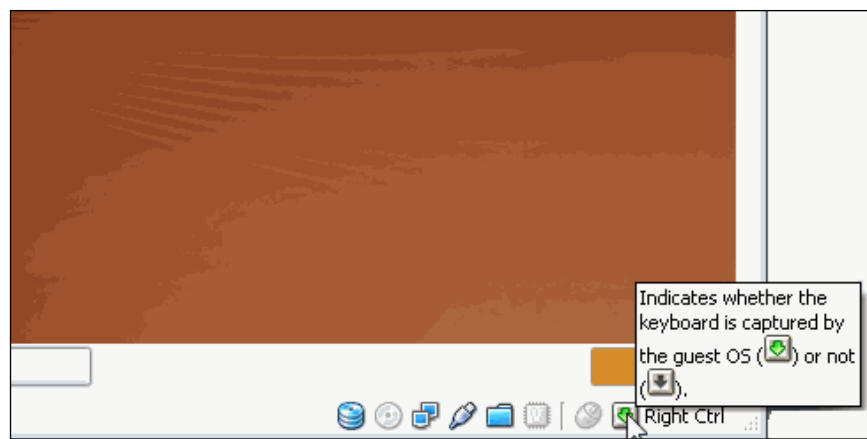
Also, if you click with your mouse inside the virtual machine's window, it will 'capture' all your mouse movements, as the information dialog in step 4 indicates. And how can we 'uncapture' the keyboard and the mouse to use it outside the virtual machine again? VirtualBox uses a special key, called the **Host Key**, for this purpose. By default, the host key is **Right Ctrl**.

To redefine the host key, you need to go to the VirtualBox main screen and select **File | Preferences** in the main menu:



Then you need to select the **Input** category in the **VirtualBox – Settings** dialog, click on the **Host Key** field, and then hit the key you want to use as the new host key. Finally, you just need to click on the **OK** button to apply the changes.

Once you define the key you want to use as the host key, you can use it to 'capture' and 'uncapture' the keyboard and the mouse in your virtual machine. Every VM shows the actual state of the host key at the bottom-right part of its main window:



In short, to use your virtual machine, you need to move your mouse inside its main window and click on it. You can also select the virtual machine's window and hit the host key to 'capture' the keyboard and the mouse. Then, if you want to exit your virtual machine and use something on your host PC, you just hit the host key, and you can move your mouse out of the virtual machine's window, along with your keyboard. Simple enough, right?

The last thing I want to mention is the color depth of your virtual machine's screen. In step 3 of the previous exercise, an information dialog appeared on your virtual machine to tell you about this. Don't worry about this for now; we'll take care of it in Chapter 4 when dealing with the Guest Additions and hardware acceleration. Now let's just try out your new Ubuntu virtual machine!

Pop quiz – using the auto capture and host key features

1. When you start a VM for the first time, the host key lets you do the following:
 - a. Close the virtual machine whenever you want.
 - b. 'Capture' and 'uncapture' your keyboard and mouse to alternate between your virtual machine and your host PC.
 - c. See what your boss is doing on his/her PC.
2. Before opening your virtual machine, you were chatting with a friend on your host PC. Now that your VM is running, you can't go back to your messaging application because the mouse won't move out of the VM's screen! What can you do?
 - a. Call 911 and ask for immediate help!
 - b. Try to open a messaging application inside your virtual machine.
 - c. Hit the host key to 'uncapture' your keyboard and mouse so you can go back to your messaging application.

3. Your boss doesn't want to use the *Right-Ctrl* key as the host key for his virtual machines; he wants to use the *F11* key instead. What can you do to help him?
 - a. Pop out the *F11* and *Right-Ctrl* keys with a paperclip; then interchange their positions, and insert them back again.
 - b. Use the **VirtualBox – Settings** dialog to change the host key.
 - c. Hire a professional programmer to make the change for him.

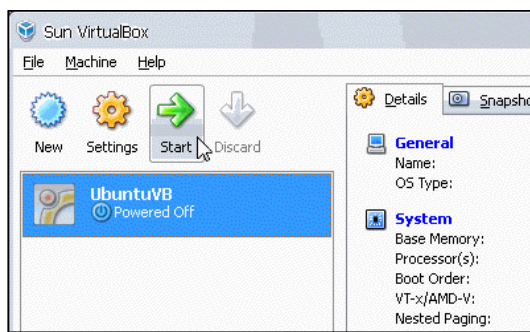
Running your Ubuntu Linux VM

This is going to be the most entertaining section of the chapter: you'll get to play with your brand-new Ubuntu Linux virtual machine! If you haven't used Linux before, I'd definitely recommend that you browse through the Ubuntu documentation at <https://help.ubuntu.com/9.10/index.html>.

Time for action – running Ubuntu Linux

The best way to test your new virtual machine is experimenting, so let's get on with it!

1. Open VirtualBox (in case you closed it after the last section's exercise), select your **UbuntuVB** virtual machine, and click on **Start** to turn it on:



2. Ubuntu will start to boot in your virtual machine. Eventually, the Ubuntu logo will show up along with the progress bar and, after a few seconds (or minutes, depending on your hardware), the Ubuntu login screen will show up. Click inside the virtual machine screen to capture the mouse and keyboard, type the username you assigned in the installation process, and hit *Enter* to continue.
3. Now type the password for your username, and hit *Enter* again. Ubuntu will start to load. When finished, you'll see the Ubuntu GNOME Desktop screen:



4. One of the first things you'll notice is the **Update Manager** dialog. This dialog shows up when your Ubuntu system needs software updates. Click on **Install Updates** to start the updating process. Normally, the Update Manager will ask for your administrator password. Type it, press *Enter*, or click on **OK** and then wait for the Update Manager to finish its job so you can work with your Ubuntu system fully updated.
5. If the Update Manager asks you to restart your Ubuntu system after updating, click on the **Restart Now** button, and wait for your Ubuntu virtual machine to reboot.

What just happened?

Isn't it cool to have a little Ubuntu system running inside your real PC? Just like a pregnant mother feeling her baby's first movements! Well, not as touching, but you get the point, right?

Ubuntu is one of the friendliest Linux distributions available. That's why I decided to use it for this chapter's exercises. Now let's go and test the Internet connection on your new Ubuntu virtual machine!

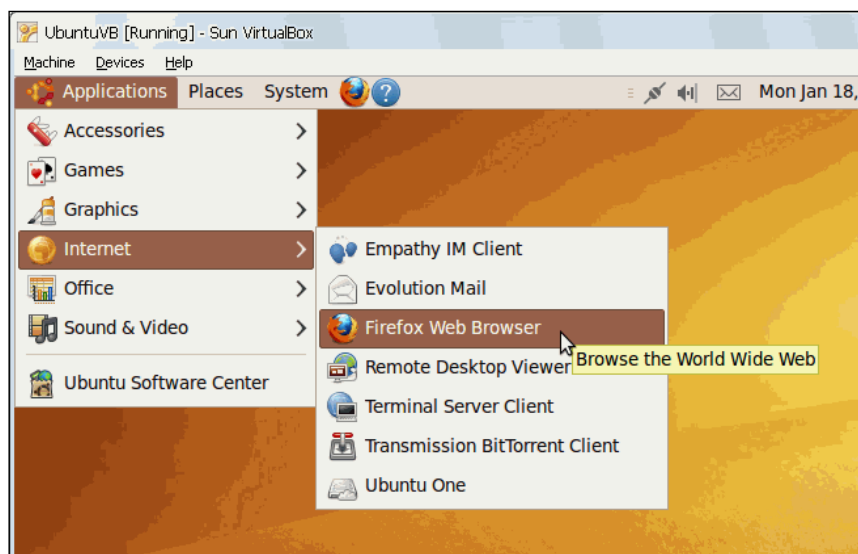
Web browsing with Mozilla Firefox

One of the best things about the Ubuntu Desktop edition is that you can use Mozilla Firefox out of the box. And the Ubuntu Update Manager keeps it updated automatically for you!

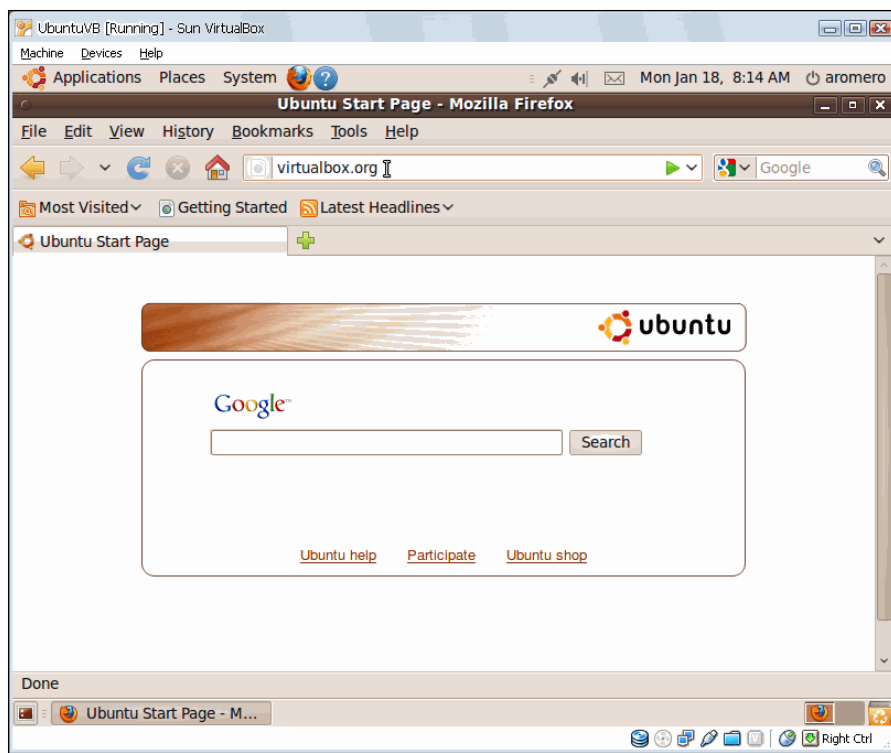
Time for action – web browsing in your Ubuntu VM

You have your virtual machine installed. What's next? Let's surf the web! After all, what could be more important than that?

1. Open the **Applications** menu on your Ubuntu virtual machine, and select **Internet | Firefox Web Browser** from the menu:



2. The Mozilla Firefox window will show the **Ubuntu Start Page**. Type `virtualbox.org` on the address bar and press *Enter*:



2. The VirtualBox homepage should appear as an indication that you have Internet access in your virtual machine. You can close Mozilla Firefox now.



If you cannot connect to Internet from your virtual machine, check your host's network settings. If you can connect from your host, try using another virtual network adapter type in your virtual machine to see if the problem disappears. In Chapter 6, *Networking with Virtual Machines*, you'll find out how to change the network adapter type in your virtual machines.

What just happened?

Well, this exercise is not really hard, right? But this is a cool way to test if your new virtual machine has Internet enabled by default. Later on, we'll talk about the different settings related to virtual network interfaces and VirtualBox. You can also know if your virtual machine can connect to Internet through the Ubuntu Update Manager because it will issue a warning if it cannot access the Ubuntu software sources. For now, it's good to know we can surf the web! Now let's see how you can do some real work inside your Ubuntu VM...

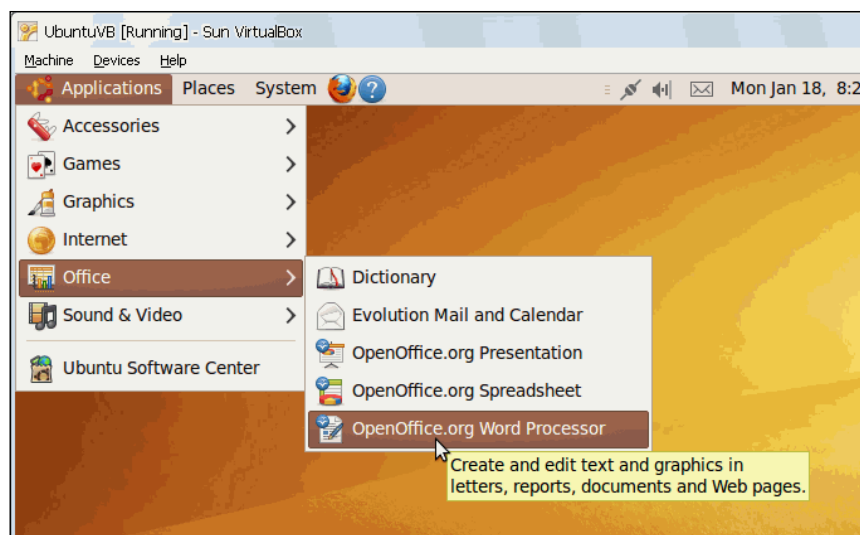
Using OpenOffice.org in your virtual machine

Ok, we have Internet enabled on our Ubuntu virtual machine; what else could we ask for? How about some word processing, a spreadsheet, and some presentations, for starters? I know it's boring, but some of us also use VirtualBox to work!

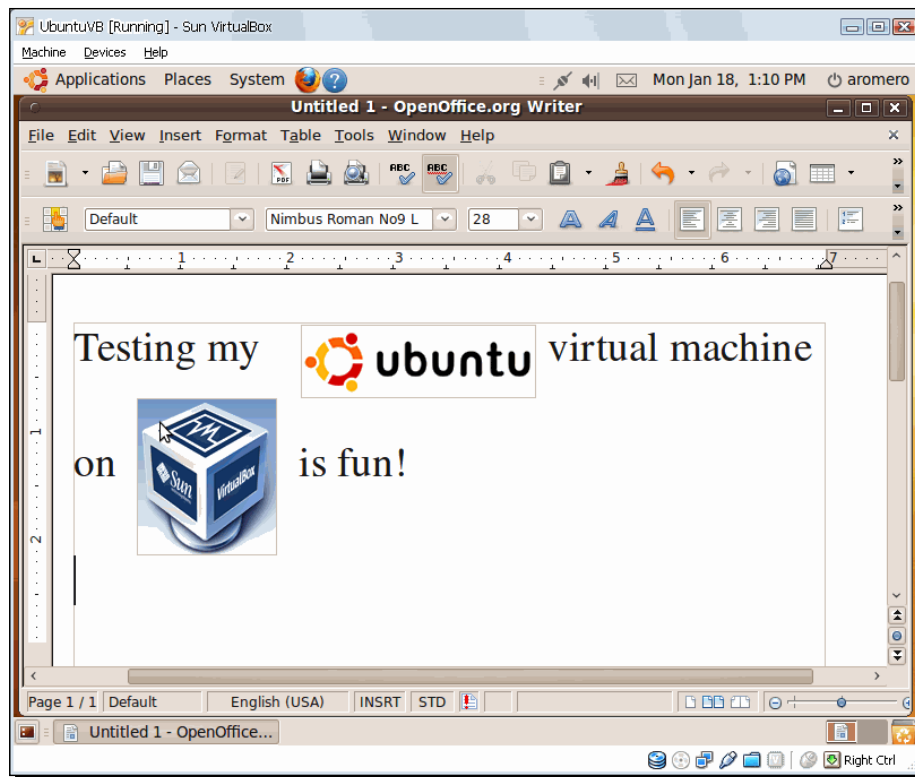
Time for action – using OpenOffice.org

Ubuntu comes with OpenOffice.org, the open source productivity suite that has proven to be an effective alternative to MS Office for Linux users. Now let's try it out on your new Ubuntu virtual machine...

1. Open the **Applications** menu on your Ubuntu virtual machine, and select **Office | OpenOffice.org Word Processor** from the menu:

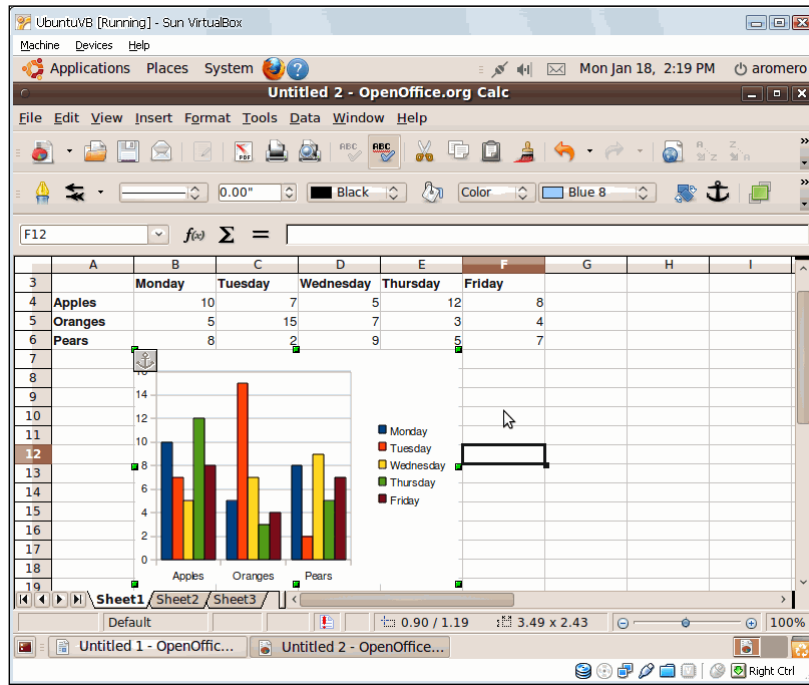


2. The **Untitled 1 – OpenOffice.org Writer** window will appear. You can use OpenOffice Writer as if you were on a real machine:

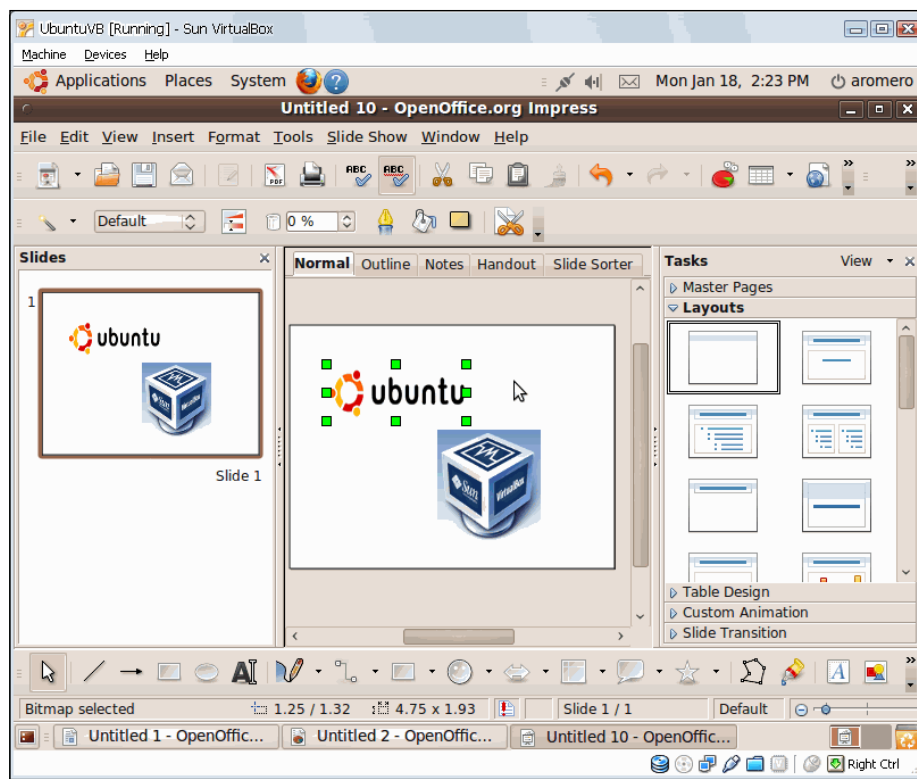


3. Now go to the **Applications** menu again, and this time select the **Office | OpenOffice.org Spreadsheet** option.

4. The **Untitled 2 – OpenOffice.org Calc** window will show up, overlapping the Writer window. You can also work with it as in a real PC:



5. And now, go back to the **Application** menu, and select the **Office | OpenOffice.org Presentation** option.
6. The **Presentation Wizard** screen will show up. Select the **Empty Presentation** option, click on **Next** twice, and then click on **Create** to continue. The **Untitled 3 – OpenOffice.org Impress** window will show up, overlapping the other two windows:



7. Now you can close all the application windows inside your virtual machine.

What just happened?

How about that? A complete office productivity suite inside your main PC! And Internet access too! So, if you always wanted to learn about Linux or any other operating system but were afraid of messing up your main PC, VirtualBox has come to your rescue!

Now let's see how to turn off your virtual machine...

Have a go hero – trying out Ubuntu One: your personal cloud

Now that you have an Ubuntu virtual machine, you would likely benefit from trying out the Ubuntu One service, where you can back up, store, sync, and share your data with other Ubuntu One users. And the best of all, it's free! To open an account, select **Applications | Internet | Ubuntu One**, and follow the instructions on screen.

Have a go hero – sharing information between your VM and your host PC

Use your Ubuntu One account to transfer some files between your virtual machine and your host PC. If you're using Windows, you can work with the Ubuntu One web interface at <http://one.ubuntu.com>.

Have a go hero – instant messaging from your virtual machine

Open the Empathy IM client in your Ubuntu virtual machine (**Applications | Internet | Empathy IM Client**), and try to use your instant messaging account to test if you can see any difference when sending instant messages from your virtual machine and from your host PC. You can also create a new account in MSN, Yahoo, Google Talk, or any other messaging protocol supported by Empathy. To see the whole list of features, you can visit <http://live.gnome.org/Empathy>.

Pop quiz – running your Ubuntu Linux VM

1. Can you use the Internet Explorer web browser on your Ubuntu virtual machine?
 - a. No, but you can use the Mozilla Firefox web browser, or you can create another virtual machine with a Windows guest operating system.
 - b. Yes, because it's the only web browser available nowadays.
 - c. I really don't know.
2. Your boss has heard a lot of wonderful things about the OpenOffice productivity suite, but he isn't sure if the guys from the sales team would be willing to change their beloved MS Office products for a free open-source alternative. What can you do to help him out?
 - a. Install VirtualBox, and create an Ubuntu virtual machine on all the sales team's PCs so that they can try out OpenOffice with copies of the real files without risking their precious data.
 - b. Install OpenOffice on all the sales team's PCs, and let them work with the real data.
 - c. Buy a new computer for each member of the sales team, and install OpenOffice on it.

3. Mr. Jones and his wife need to share the same Windows XP computer due to financial problems, but they don't want to mess up each other's files and applications. Since Mr. Jones is a freelance Java programmer who works with Linux applications, and Mrs. Jones needs MS Word for her daily tasks, could Mr. Jones use an Ubuntu virtual machine to keep all of his work isolated from his wife's files?
- Yes.
 - No.
 - Geez, I don't know!

Shutting down your virtual machine

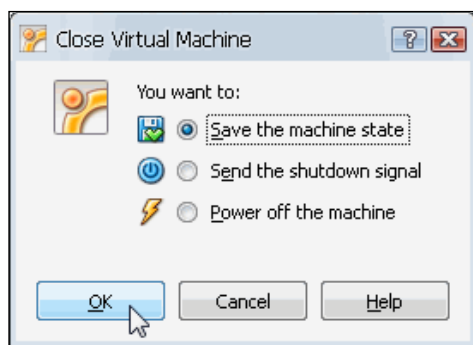
I know you're thinking, "Geez, I can't believe this guy! He's actually going to spend an entire subsection of this chapter just to show us how to shutdown a virtual machine! Aw, come on!"

Now it's my turn: Remember we're talking about a virtual machine here, not a real PC! You need to consider several things before shutting this baby down!

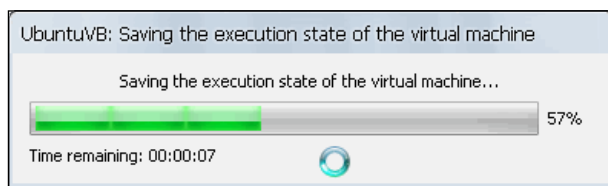
Time for action – shutting down your VM

Now it's time to stop whining and start learning how to shut down your virtual machine...

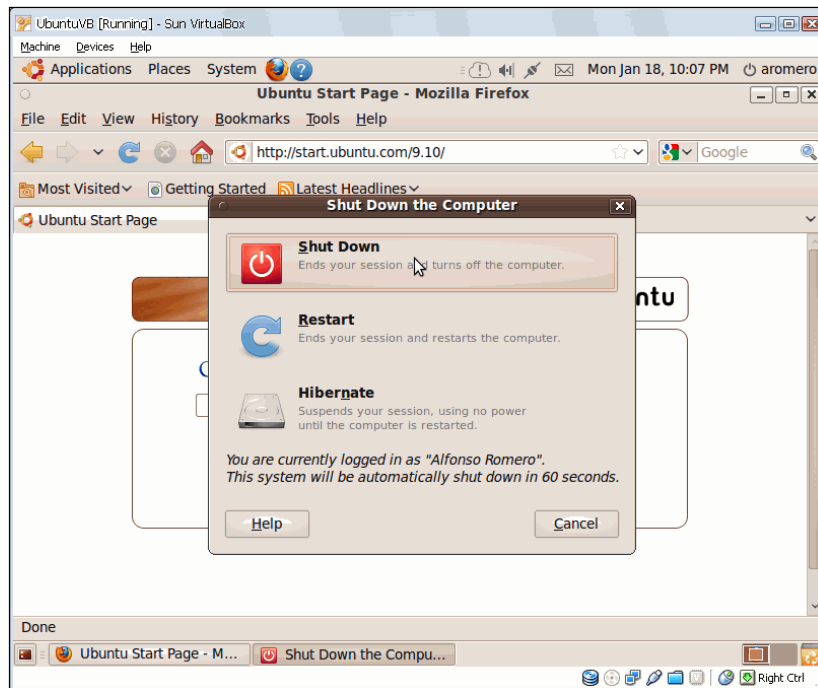
1. Make sure you close all the applications inside your virtual machine, open the Mozilla Firefox Web browser, 'uncapture' your keyboard and mouse so that you can move to the VirtualBox main menu, and select the **Machine | Close** option.
2. The **Close Virtual Machine** dialog will show up with three choices for shutting down your virtual machine. Select the **Save the machine state** option, and click on **OK** to continue:



- VirtualBox will show the following dialog before shutting down the virtual machine:



- Once your virtual machine's state is saved, it will shut down automatically, and you'll be returned to the VirtualBox main screen. Also, your **UbuntuVB** virtual machine status indicator will change from **Powered Off** to **Saved**. If you start it again, it will continue exactly where it was left off, with the Mozilla Firefox web browser window open.
- Start your UbuntuVB virtual machine again, leave the Mozilla Firefox window open, and select the **Machine | Close** menu option from the VirtualBox main menu again to open the **Close Virtual Machine** dialog.
- This time select the **Send the shutdown signal** option, and click on **OK** to continue. This has the same effect as if you had pressed the power off button on a real PC. The **Shutdown the computer** dialog from Ubuntu will show up:



7. Click on the **Shut Down** button to continue. Your Ubuntu virtual machine will shutdown without saving its state. This is the same as using the regular shutdown process in Ubuntu: you select the **Shut Down...** option from the pull-down menu in the upper right corner of your Ubuntu virtual machine. If you later start it again, the Mozilla Firefox window won't open.

What just happened?

Well, it was pretty much just a simple 'shutdown your virtual machine' exercise, don't you agree? In VirtualBox, you have three choices when shutting down your virtual machines. We already saw the first choice –**Save the machine state**–in action. It's kind of like suspending or hibernating a laptop computer. Your virtual machine's state 'freezes' and stays that way until you later start it again. Then, it resumes normal operation as if nothing happened. All the applications you left open will still be there.

The second choice, **Send the shutdown signal**, acts as if you pushed the power button on a real machine. Most modern operating systems will try to use a proper shutdown mechanism.

The third choice, **Power off the machine**, is like pulling the power plug on a real PC, so be careful! You could lose sensitive data if you use it carelessly! I only use this option when a virtual machine crashes and I can't shut it down with the other two choices...



Always try to use the guest operating system interface first to shutdown the virtual machine. If that fails, you can use the **Send the shutdown signal** option instead, but it's better to treat your virtual machine as if it were a real PC, don't forget it.

Remember that the **Send the shutdown signal** option only works if your guest operating system supports ACPI events. For example, this option is useless when your virtual machine is showing up the Ubuntu GRUB loader boot screen.

Pop quiz – your first VM

1. Your boss has been a real pain in the butt lately and all because he wants to know how to avoid the annoying dual-boot process between Windows XP and Linux. He's always testing new applications on Linux but needs to do his reports on MS Excel. What could you do to help him?
 - a. Donate one year of your salary to buy your boss another computer with Linux installed.
 - b. Use VirtualBox to get rid of dual-booting, and show your boss how he can have Windows XP as the host operating system, along with an Ubuntu Linux virtual machine.
 - c. Call the Fonz for help! Heeyyy!

2. If you were a teacher, what would be the best way to teach your students how to use Ubuntu?
 - a. Install Windows XP on the physical PC and Ubuntu on a virtual machine.
 - b. Install Ubuntu on the physical PC and Windows XP on a virtual machine.
 - c. Install Ubuntu on the physical PC and Ubuntu on a virtual machine.
3. What would happen if you assigned 768 MB of RAM to your virtual machine, with only 1 GB of physical RAM in your system?
 - a. The virtual machine would fly as an eagle!
 - b. A doorway to the underworld would open, causing infinite dismay!
 - c. The host PC would definitely crash, and all data could be lost.
4. You're going through difficult financial times, so you resolve to sell your laptop and your son's PC. The only PC left is the one at your studio. What can you do to share that PC with your wife and kid?
 - a. Establish a daily schedule so each of you uses the PC at different times throughout the day.
 - b. Each one of you must boot your PC from a different hard disk.
 - c. Install VirtualBox, and create two Ubuntu virtual machines: one for your wife and the other one for your kid. That way each one of you can have a completely isolated environment.
5. Just five more minutes and you'll be on your way to a dreamy weekend! Suddenly, you hear a cry: "Martinez, come over here!" Oh no... It's your boss! What does he wants now? "I can't turn this damn thing off!" he shouts angrily at you. You take a look at the LCD, and a sarcastic grin almost slips because of your thoughts... Your boss opened two virtual machines at the same time, and now one of them isn't responding, maybe due to low memory. The next step is to
 - a. Start laughing frantically at your boss, and show him who the 'real McCoy' is by pulling the plug on his computer.
 - b. Try to restrain yourself from laughing, and show your boss how to shutdown the faulty virtual machine by using the **Power off the machine** option.
 - c. Grab your boss's head, smash it against the LCD screen, and run away like mad!

Have a go hero – experimenting with a KUbuntu virtual machine

Excellent, my friend. You installed and ran successfully your first Ubuntu virtual machine! Now the world can be yours. But wait! What if I told you that there's another Ubuntu flavor you have not tasted yet? Yeah! Your Ubuntu virtual machine is one of the coolest around, but not the only one, indeed!

There's another Ubuntu-derived distribution around called **KUbuntu**. Why the K? Well, the primary difference between Ubuntu and KUbuntu is that Ubuntu uses the GNOME window manager (<http://www.gnome.org>), and KUbuntu uses the KDE window manager (<http://www.kde.org/>). And which one is better? Well, that's what you need to try out for yourself...

Go and grab a copy of KUbuntu from <http://www.kubuntu.org>, create a virtual machine named KUbuntuVB or something like that, and install KUbuntu on it. If possible, try to run your Ubuntu and KUbuntu virtual machines at the same time to compare them, so you can decide which one is better!

And if you're feeling reckless, you should definitely try out other Linux distributions, for example: SuSE, Red Hat, Slackware, or Fedora. Come on, I know you can do it!

Summary

I hope you enjoyed the exercises in this chapter, especially if you've never used Ubuntu Linux before (or any other Linux distro whatsoever). Here you learned about the basic settings needed to create a virtual machine, installed the Ubuntu operating system, ran the virtual machine, and shut it down appropriately.

Specifically, we covered:

- ◆ How to download the Ubuntu Linux Desktop Live CD
- ◆ How to configure a new virtual machine in VirtualBox
- ◆ How to adjust your virtual machine's basic settings to install Ubuntu on it
- ◆ How to install the Ubuntu Linux Desktop operating system on your VM
- ◆ How to use the VirtualBox host key and to 'capture'/'uncapture' the mouse and keyboard in your virtual machine
- ◆ How to run your Ubuntu virtual machine and test some basic functions such as Internet access and OpenOffice.org applications
- ◆ The different choices available when shutting down your virtual machine

Now that you are capable of creating a virtual machine based on the Ubuntu Linux distribution, let's see how to create another virtual machine and try a Windows operating system in the next chapter...

3

Creating Your Second Virtual Machine: Windows 7

In this chapter, you'll get to install Windows 7 on a virtual machine. I used a Windows XP host computer for the exercises, but you can use any host supported by VirtualBox.

In this chapter you shall:

- ◆ Create your second virtual machine in VirtualBox using Microsoft Windows 7
- ◆ Learn about your virtual machine's basic configuration
- ◆ Install Windows 7 on your virtual machine
- ◆ Learn how to use the VirtualBox host key and to 'capture'/'uncapture' the mouse and keyboard in your virtual machine
- ◆ Learn to run your Windows 7 virtual machine and test some basic functions like using Internet Explorer and installing the Microsoft Office 2007 trial edition
- ◆ Learn how to start and stop your virtual machine

So let's get on with it...

You'll need a Windows XP/Vista/7 original CD or ISO image for the exercises in this chapter. I wrote the exercises using a Windows 7 DVD (32-bits), but the process is very similar for any other Windows version. If you have any questions or doubts about the exercises or the Windows version you plan to use, don't hesitate to send an email to questions@packtpub.com.

Creating your Windows VM

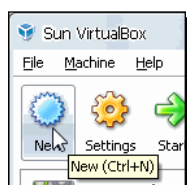
If you read Chapter 1, this is going to be your second virtual machine on VirtualBox. Maybe you're thinking, "Nah, I'll skip this chapter 'cause I already know how to use Windows 7." Well, what if you wanted to try a new software application but were afraid to mess up your main computer at work? You could create a Windows 7 virtual machine and test the software application on it to keep your main PC completely isolated in case anything went wrong. How about that?

There are a lot of things you can do with a Windows 7 virtual machine inside a Windows XP host PC. Just think of a VM as if it were a separate PC where you can test software, or you can even share one physical PC with two or more people, creating a virtual machine for you and your colleagues! Now let's keep on reading...

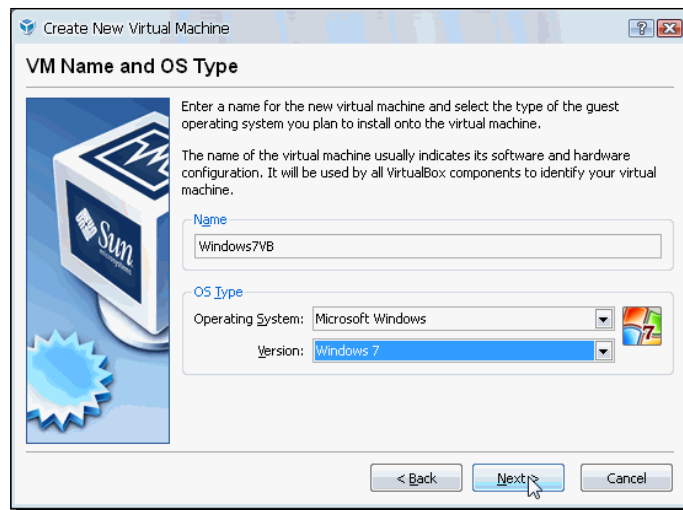
Time for action – creating a virtual machine

In this exercise, I'll show you how to create a Windows 7 virtual machine in a Windows XP host.

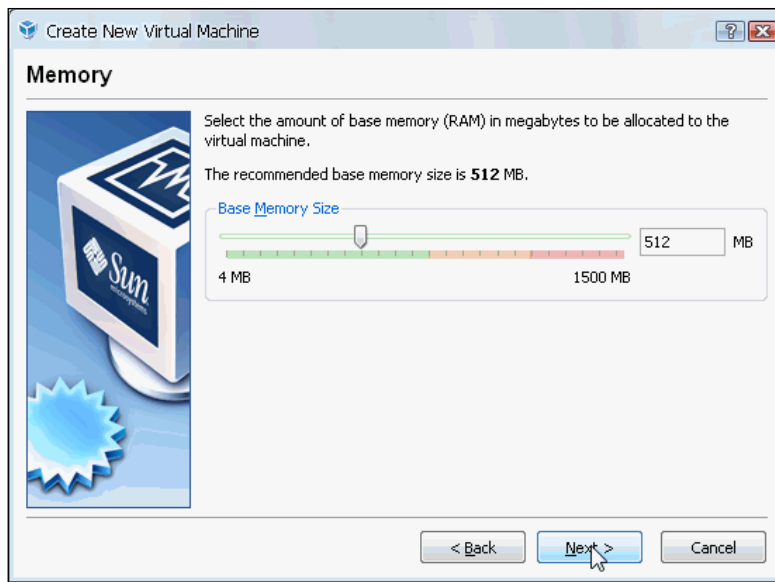
1. Open VirtualBox, and click on the **New** button (or press *Ctrl+N*) to create a new virtual machine:



2. The **Welcome to the New Virtual Machine Wizard!** dialog will show up. Click on **Next** to go to the **VM Name and OS Type** dialog. Type `Windows7VB` in the **Name** field, select `Microsoft Windows` as the **Operating System** and `Windows XP` as the **Version**:



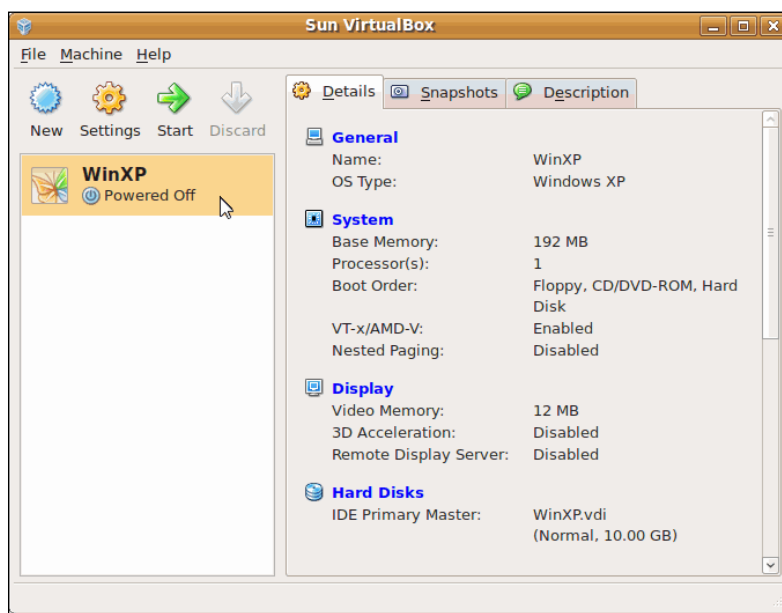
3. Click on **Next** to continue. You can leave the default **512 MB** value in the **Memory** dialog box or choose a greater amount of RAM, depending on your hardware resources:





It's wise to stick with the default values when creating a new virtual machine.

4. Click on **Next** to continue. Leave the default values in the **Virtual Hard Disk** dialog, and click on **Next** twice to enter the **Create New Virtual Disk** wizard. Leave the default **Dynamically Expanding Storage** option in the **Hard Disk Storage Type** dialog, and click on **Next** to continue.
5. Leave the default values chosen by VirtualBox for your Windows XP machine in the **Virtual Disk Location and Size** dialog, and click on **Next** to continue.
6. A **Summary** dialog will appear, showing all the parameters you chose for your new virtual hard disk. Click on **Finish** to exit the **Create New Virtual Hard Disk** wizard. Now another **Summary** dialog will appear to show the parameters you chose for your new virtual machine. Click on **Finish** to exit the wizard and return to the VirtualBox main window:



7. Your new **Windows7VB** virtual machine will appear on the VirtualBox main screen, showing all its parameters in the **Details** tab.

What just happened?

Now your Windows 7 virtual machine is ready to run! In this exercise, you created a virtual machine with all the parameters required for a typical Windows 7 operating system. The first parameter to configure was the **base memory** or **RAM**. As you saw in step 3, the recommended size for a typical Windows 7 installation is 512 MB.



Exercise caution when selecting the RAM size for your virtual machine; the golden rule of thumb dictates that if you have less than 1 GB of RAM, you can't assign more than one half of your physical RAM to a virtual machine because you'll get into trouble! Your **host** PC (the one that runs VirtualBox) will start to behave erratically and could crash!



The recommended minimum amount of RAM for Windows 7 is 1 GB, although you can get it to run with 512 MB.

With 2 GB, for example, you can easily assign 1 GB to your virtual machine and 1 GB to your physical PC without any problems. Or you could even have three virtual machines with 512 MB of RAM each and leave 512 MB for your host PC. The easiest way to find out the best combination of RAM is to do some experimenting yourself and to know the minimum RAM requirements for your host and guest operating systems.



Don't assume that assigning lots of RAM to your virtual machine will increase its performance.

If, for example, you have 4 GB of RAM on your host, and you assign 2 GB to a Windows virtual machine, it's very unlikely there will be a bigger performance increase than if you were assigning only 1 GB to the same Windows 7 virtual machine. On the contrary, your host PC will work better if you only assign 1 GB to the Windows 7 virtual machine because it will use some of the extra RAM for disk caching, instead of recurring to the physical hard disk. However, it all depends on the guest operating system you plan to use and what applications you need to run.

If you look closely at the **Base Memory Size** setting in the **Memory** dialog when creating a virtual machine, you'll notice that there are three memory areas below the slider control: the green-colored area indicates the amount of memory range you can safely choose for your virtual machine; the yellow-colored area indicates a dangerous memory range that you can choose, but nobody knows if your virtual machine and your host will be able to run without any problems; and the red-colored area indicates the memory range that your virtual machine can't use. Later on, if you need to run a memory-intensive application, you can add more RAM through the **Settings** button, as we'll see in the following section.

Another setting to consider when creating a virtual machine besides memory is the **virtual hard drive**. Basically, a virtual hard disk is represented as a special file on your host computer's physical hard disk, and it's commonly known as a **disk image file**. This means that your host computer sees it as a 'large' file on its system, but your virtual machine sees it as a 'real' hard disk connected to it. As virtual hard drives are completely independent from each other, there is no risk of accidental overwriting, and they can't be larger than the free space available on your real computer's physical hard drive. This is the most common way of handling virtual storage in VirtualBox; later on, we'll see more details about disk image files and the different formats available.

VirtualBox assigns a default value based on the guest operating system you plan to install in your virtual machine. For Windows 7, the default value is 20 GB. That's enough space to experiment with Windows 7 and learn to use it, but if you really want to do some serious work—desktop publishing or movie production, for example—you should consider assigning your virtual machine more of your hard disk space.

Just keep in mind there are a lot of combinations you can explore when assigning virtual disks to your virtual machines: for example, you can get two hard drives on your physical machine and assign one for your host system and the other one for your virtual machine. You can also create a new virtual hard disk image and add it as if it were a second hard drive. We'll talk more about this in a later chapter dealing with virtual storage.

There are two virtual hard disk storage types available in VirtualBox: **dynamically expanding storage** and **fixed-size storage**. The **Hard Disk Storage Type** dialog you saw in step 4 of the previous exercise contains a brief description for both types of storage. At first the dynamically expanding option might seem more attractive because you don't see the space reduction in your hard drive immediately.

When you are using dynamically expanding storage, VirtualBox needs to expand the storage size continuously, and that could mean a slight decrease in speed when compared to a fixed-size disk, but most of the time this is unnoticeable. Plus, when the virtual hard disk is fully expanded, the differences between both types of storage disappear.

Most people agree that dynamically expanding storage represents a better choice than fixed storage, since it doesn't take up unnecessary space from your host's hard disk until needed. Personally, when experimenting with a new virtual machine, I use the dynamically expanding option, but when doing some real work, I like to set apart my virtual machine's hard disk space from the beginning, so I choose the fixed-size storage option in these cases.

Have a go hero – experimenting with memory and hard disk storage types

Create another virtual machine named `Windows7VB2`, and try to assign the entire RAM available to it. You won't be able to continue until you select a lower value because the **Next** button will be grayed out; that means you're in the red-colored memory range. Now move back the slider until the **Next** button is active again; you'll probably be in the yellow-colored memory range. See if your virtual machine can start with that amount of memory and if you can use both your host PC and your VM without any problems. In case you encounter any difficulties, keep moving back the memory range until all problems disappear.

Once you're done experimenting with the memory setting, use the `Windows7VB2` virtual machine with the same exact settings as the one you created in the previous exercise, but this time use a fixed-size hard drive. Just take into account that because VirtualBox must prepare all the storage space at once, this process may take a long time depending on the storage size you selected and the performance of your physical hard disk. You can try using a Windows XP guest instead of Windows 7 because it doesn't require the same amount of hard disk space.

Pop quiz – creating virtual machines

1. When you create a virtual hard disk on VirtualBox:
 - a. A 'large file' is created on your host PC's filesystem to represent the virtual hard disk.
 - b. Your new virtual machine uses up all available space on your physical hard disk.
 - c. The virtual hard disk downloads the guest operating system automatically.
2. How many Windows 7 virtual machines could you create on a 160 GB empty physical hard drive?
 - a. Only one, because it would take up all the available space on the physical drive.
 - b. 4, if you're using half the space on the drive for your host PC operating system.
 - c. None, because you can't have virtual machines on the same physical drive used by your host PC.
3. If you create two virtual machines with a 20 GB virtual drive each and your host PC has a 200 GB hard drive,
 - a. Your host PC will crash.
 - b. The virtual hard drives will overwrite each other.
 - c. All will be OK because the virtual hard drives will be isolated from each other and from your host PC.

4. If you have 2 GB of RAM on your host PC and you want to create a 1 GB Windows 7 virtual machine, would it run without any problems?
 - a. It would be better to assign only 768 MB or less to your Windows 7 virtual machine or to go out and buy some more RAM.
 - b. Your Windows 7 virtual machine and your host PC would run without any problems.
 - c. It would be better to assign the 2 GB of available RAM to your Windows 7 virtual machine.

Booting your Windows 7 installation disk

The next thing to do is tell your virtual machine where to boot your Windows 7 installation DVD from, as if you were working on a real PC. In Chapter 2, *Creating Your First Virtual Machine: Ubuntu Linux*, you learned how to adjust the basic configuration settings for your VM so you could start the Ubuntu installation.

Now I'll show you a quicker way to install a guest operating system: if you click on the **Start** button without adjusting your virtual machine settings, the **First Run Wizard** will show up and guide you through the whole booting process, as we'll see in the next exercise.

Time for action – booting your Windows 7 installation disk through the First Run Wizard

In this exercise, you'll learn how to use the VirtualBox First Run Wizard to install a guest operating system the quick way.

1. Insert your Windows 7 Installation DVD in your host computer's DVD/CD-ROM drive.
2. Open VirtualBox, select your **Windows7VB** virtual machine, and click on the **Start** button.
3. An **Information** dialog will appear to tell you that the **Auto capture keyboard** option is turned on. Enable the **Do not show this message again** option, and click on **OK** to continue.
4. Now the **Welcome to the First Run Wizard!** dialog will show up. Click on **Next** to continue.
5. The **Select Installation Media** dialog will show up next. Make sure the **CD/DVD ROM Device** option is selected, select your CD/DVD drive letter in the **Media Source** field, and then click on **Next** to continue:



6. The **Summary** dialog will appear to confirm the options you selected. Leave this window open for the next exercise.

What just happened?

As you can see from the previous exercise, your virtual machine is just like a real PC. You need to 'insert' a bootable CD so the VM can boot and start installing Windows 7. You could also use an ISO image of any Windows operating system instead of having to insert a real bootable CD. That's one of the benefits of VirtualBox: forget about having to burn a CD to test the new version of any operating system! You can just go to the Microsoft web store (<http://store.microsoft.com>), purchase a Windows ISO image, and configure your virtual machine to act as if there was a real CD inserted in the CD-ROM drive!

And thanks to the First Run Wizard, you don't have to go through the process of adjusting your virtual machine's CD/DVD drive setting to insert your operating system's installation CD/DVD!

Whichever option you choose, your virtual machine will start to boot like a real physical PC! In the following section, you'll get to see it with your own eyes...

Have a go hero – using the First Run Wizard with other operating systems

Now that you know how easy it is to insert installation media in your virtual machines, why don't you go and create a virtual machine with another operating system, such as Ubuntu or OpenSolaris?

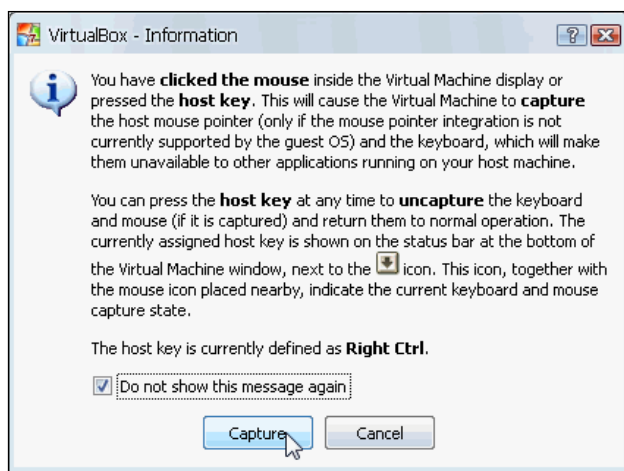
Installing Windows 7 on your VM

Now everything's ready to start installing Windows 7! At last you're about to see this wonderful piece of software in action...

Time for action – installing Windows XP on your VM

Ok, it's time to try out our new toy! Get ready for the ride of your life...

1. Click on the **Finish** button from the **Summary** dialog you left open in the previous exercise to start booting Windows 7.
2. An **Information** dialog will appear to tell you that the virtual machine window is optimized to work in **32 bit color mode** but the virtual display is currently set to **24 bit**. Don't worry about this for now; just enable the **Do not show this message again** option, and click on **OK** to continue.
3. Wait until the **Install Windows** screen shows up. Click anywhere inside the virtual machine screen, and the following information dialog will pop up:



4. Enable the **Do not show this message again**, and click on **Capture** to continue. The mouse pointer will disappear, and you will be able to move your mouse inside your **Windows7VB** virtual machine screen. Select your desired language, time/currency format and keyboard layout, and click on **Next** to continue:

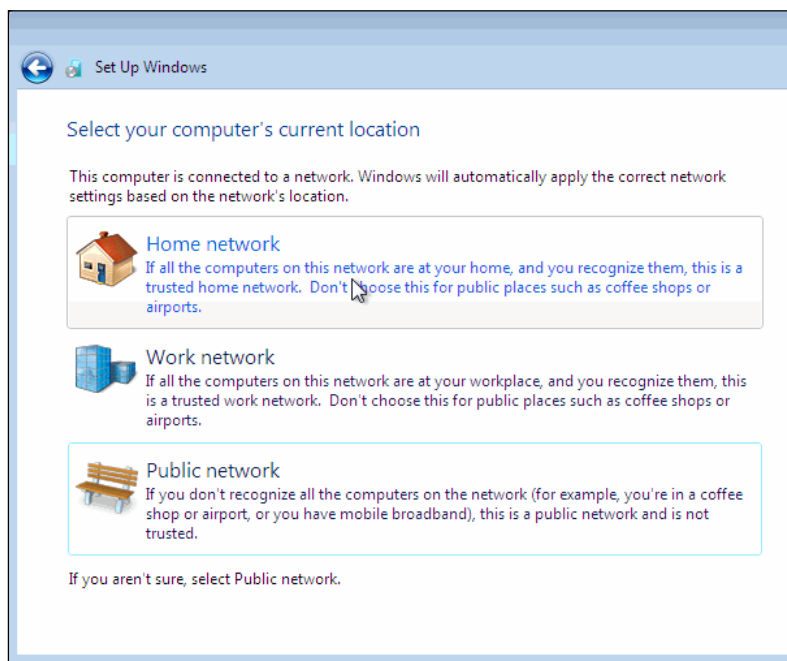


5. The screen will show the **Install Now** button next. Click on this button to continue.
6. The **Setup is starting** message will show up on the screen. Wait until the Windows 7 Licensing Agreement appears. Read it and, if you agree, enable the **I accept the license terms** checkbox. Click on **Next** to continue.
7. Click on the **Custom (Advance)** option in the **Which type of installation you want?** dialog to install a fresh copy of Windows 7. The **Where do you want to install Windows?** dialog will show up next. The only option available will be **Disk 0 Unallocated space**, so just click on **Next** to continue.

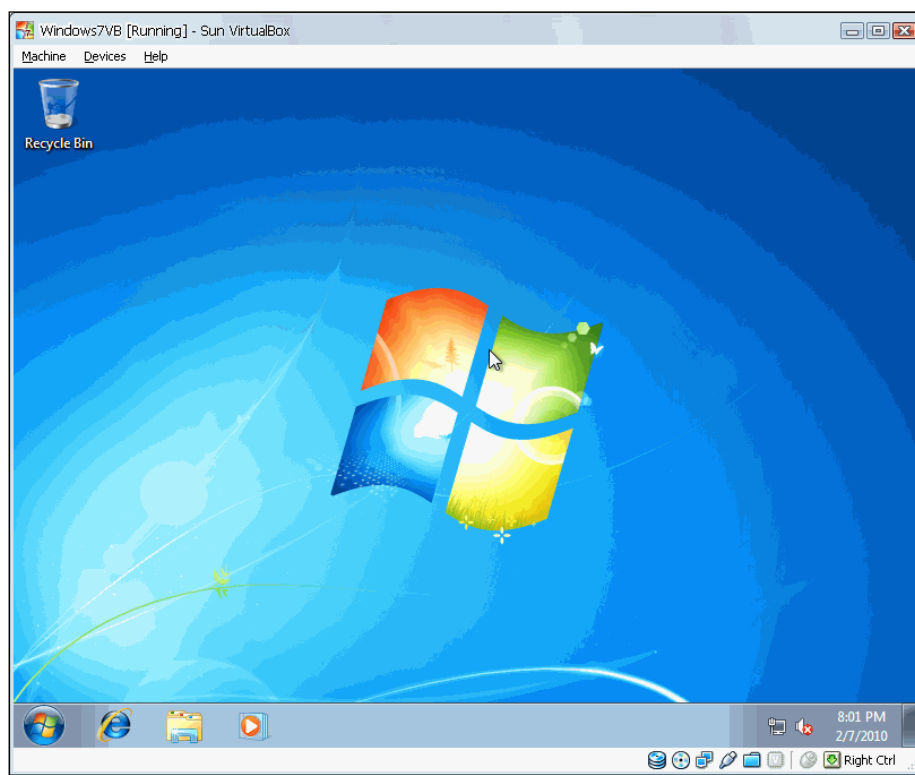


Remember, you're using a virtual hard disk and not the physical hard drive of your real PC; it's perfectly ok to use the entire virtual hard disk for your Ubuntu guest operating system. The rest of the files in your real hard disk won't be affected.

8. The **Installing Windows...** dialog will appear next to indicate the installation process has begun. Wait until the setup program asks you to choose a username for your Windows 7 account and a name for your computer. Click on **Next** when you're ready to continue.
9. The **Set a password for your account** dialog will appear next. Type the password you've chosen in the first two fields and a hint to remember it in the last field, then click on **Next** to continue.
10. The **Type your product key** screen will show up next. Type the Windows 7 key that came with your copy of Windows 7, enable the **Automatically activate Windows when I'm online** checkbox if you want to activate automatically your copy of Windows, or disable it if you're only testing Windows 7 temporarily on a virtual machine, and click on **Next** to continue.
11. Now the **Help protect your computer and improve Windows automatically** screen will show up. If you're not sure which option is right for you, click on **Ask me later** to continue.
12. Make sure the correct date and time are displayed in the **Review your date and time settings** dialog, and click on **Next** to continue.
13. The **Select your current location** screen will appear next. Select one of the three available options depending on your home/work environment to continue:



- 14.** Now you just have to wait until Windows finishes configuring your network settings and the Windows 7 desktop appears, as shown below:



- 15.** Your Windows 7 virtual machine is now ready and running! You can shut it down for now or leave it open for the next exercise.

What just happened?

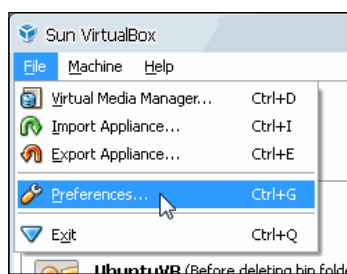
Cool! You have a virtual machine with the most recent Windows operating system installed! And it was as easy as knocking on your grandma's door to get some free cookies!

As you can see, you just need a few clicks to install a completely standalone guest operating system inside a host operating system... Now you can start playing with your brand new Windows 7 virtual machine! But before you go jumping to the next exercise, there are a few things I want to talk about.

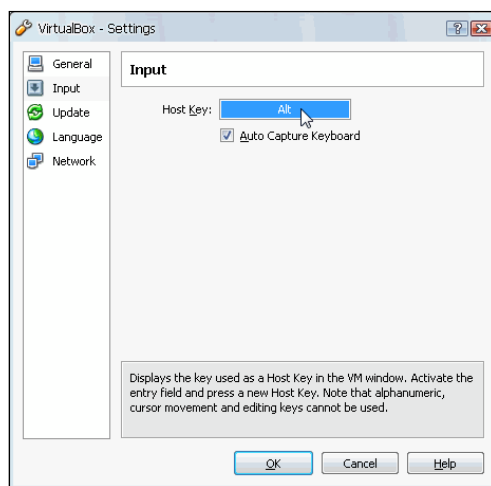
Now that Windows 7 is installed in your virtual machine, what happens if you try to move your mouse out of the virtual machine window? You'll need to 'uncapture' your mouse and keyboard because the **Auto capture keyboard** feature lets your virtual machine 'capture' all the keyboard action automatically every time you activate its window, and it's enabled by default on every new virtual machine. As you saw in step 2 of the *Booting your Windows installation disk through the First Run Wizard* exercise, an information dialog shows up to tell you about this feature when you are starting a virtual machine for the first time.

Basically, when you click inside the virtual machine's window, all your mouse's movements are 'captured' by your virtual machine. VirtualBox uses a special key, called the **Host Key**, to alternate between 'capturing' and 'uncapturing' your mouse and keyboard. By default, the host key is *Right Ctrl*.

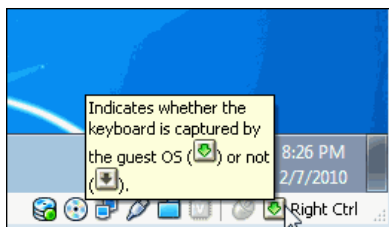
To redefine the host key, you need to go to the VirtualBox main screen and select **File | Preferences** in the main menu:



Then you need to select the **Input** category in the **VirtualBox – Settings** dialog, click on the **Host Key** field, and then hit the key you want to use as the new host key. In the following screenshot, I hit the *Alt* key:



Once you define the key you want to use as the host key, you can use it to 'capture' and 'uncapture' the keyboard and the mouse in your virtual machine. Every VM shows the actual state of the host key at the bottom-right part of its main window:



In short, to use your virtual machine, you need to move your mouse inside its main window and click on it, or you can also select the virtual machine's window and hit the host key to 'capture' the keyboard and the mouse. Then, if you want to exit your virtual machine and use something on your host PC, you just hit the host key, and you can move your mouse out of the virtual machine's window, along with your keyboard. Simple enough, right?

Ok, that's enough smart-talk for now. Let's go and try out your new Windows 7 virtual machine!

Pop quiz – using the auto capture and host key features

1. To use the mouse and keyboard in your virtual machine, you need to:
 - a. Use the First Run Wizard.
 - b. Take a PHP programming course.
 - c. 'Capture' and 'uncapture' your keyboard and mouse with the host key.
2. Can you use the *Right-Shift* key to 'capture' and 'uncapture' the keyboard and mouse?
 - a. Yes, you just need to use the **VirtualBox – Settings** dialog to change the host key.
 - b. No, because the host key is *Right-Ctrl*, and you can't change it.
 - c. Only if you're using a Windows host.
3. How can you know which key is currently defined as the host key?
 - a. You need to call 911.
 - b. Look at the bottom-right part of your virtual machine's window.
 - c. Hire a professional programmer to find out.

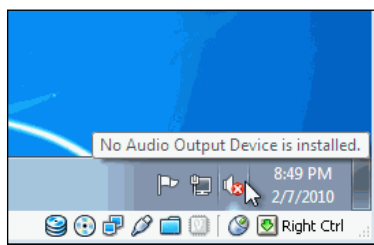
Making sound work on your Windows 7 VM

At the time of this writing, there was a problem with the sound drivers when one was running Windows 7 inside a virtual machine. The default audio controller when we are creating a virtual machine is AC97, so we need to get the correct driver for that audio controller from the manufacturer's website.

Time for action – enabling audio on your Windows 7 virtual machine

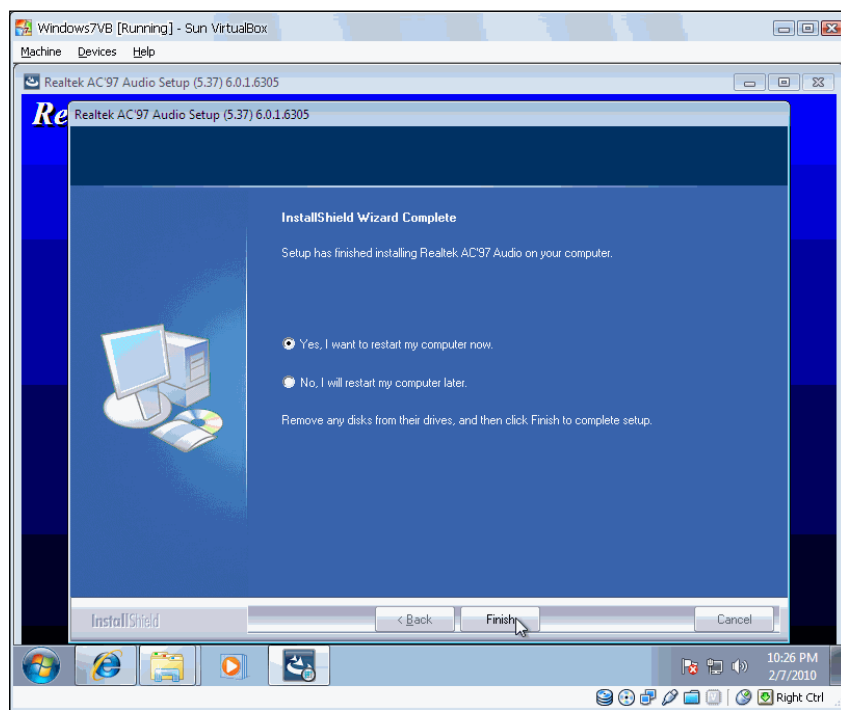
In the following exercise, I'll show you how to download the AC97 driver from the <http://www.realtek.com.tw/> website.

1. Open VirtualBox (in case you closed it after last section's exercise), select your **Windows7VB** virtual machine, and click on **Start** to turn it on. If the **Press any key to boot from CD or DVD** message shows up, just ignore it, and don't press any key.
2. Windows 7 will start to boot in your virtual machine. Eventually, the Windows 7 logo will show up along with the progress bar and, after a few seconds (or minutes, depending on your hardware), the Windows Desktop will show up.
3. If you look at the bottom-right part of the screen, you'll see the **No Audio Output Device** icon:

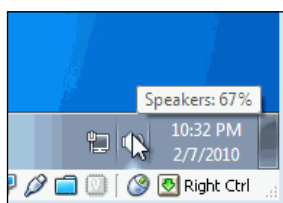


4. Open Internet Explorer, go to the <http://www.realtek.com.tw/> website, click on the **Download** link from the webpage main menu, and search for the AC97 Audio Codecs drivers, or go to the <http://www.realtek.com.tw/downloads/downloadsView.aspx?Langid=1&PNid=23&PFid=23&Level=4&Conn=3&DownTypeID=3&GetDown=false> direct link, and click on one of the download links for the **Vista/Win 7 Driver** file.
5. The **File Download** dialog will appear next. Save the file on your virtual hard disk, and wait for it to finish downloading, then extract the zip file contents and double-click on the setup file to start installing the AC97 drivers on your Windows 7 virtual machine.

6. Click on **Yes** in the **User Account Control** dialog to allow the setup program to execute on your Windows 7 VM, and follow the InstallShield Wizard instructions to install the AC97 driver. If a **Windows Security** dialog pops up complaining that Windows can't verify the publisher of the driver software, just click on the **Install this driver software anyway** option to continue the installation process.
7. Wait until the **InstallShield Wizard Complete** dialog appears, make sure the **Yes, I want to restart my computer now** option is selected, and click on **Finish** to continue:



8. Wait until your Windows 7 virtual machine restarts. If the **Press any key to boot from CD or DVD** message shows up, ignore it, and don't press any key; just wait until Windows 7 starts booting up automatically. Login with the username and password you chose when installing Windows 7, and wait for the Desktop screen to appear. The speaker icon at the bottom-right part of the screen will now be enabled:



9. At last! You have audio on your Windows 7 virtual machine! You'll need to shut it down for the next exercise.

What just happened?

In this exercise, we downloaded the updated AC97 driver to enable sound on your Windows 7 virtual machine. On Windows XP the default driver works seamlessly, but in Windows 7 we had to download and install an updated driver from the manufacturer's website, as if it were a real computer.

Since Windows 7 is the most recent operating system from the Windows family, I know you'll be anxious to try it out, if you haven't yet. So now that we've got sound working, let's remove the installation media to get rid of the annoying **Press any key to boot from CD or DVD** message!

Have a go hero – verifying audio in Windows guests

It would be cool if you could get copies of Windows Vista and Windows XP and check if there's an audio problem with those operating systems when they are in use in a virtual machine.

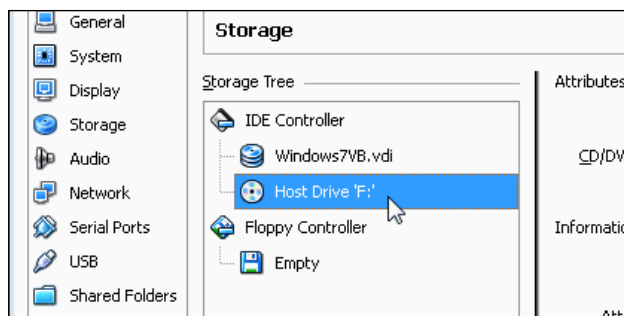
Removing the installation media from your Windows 7 VM

Now that your Windows 7 virtual machine is running and the audio problem is solved, let me show you how to remove the installation media from your CD/DVD virtual drive so that you don't have to see the **Press any key to boot from CD or DVD** message every time you start your virtual machine.

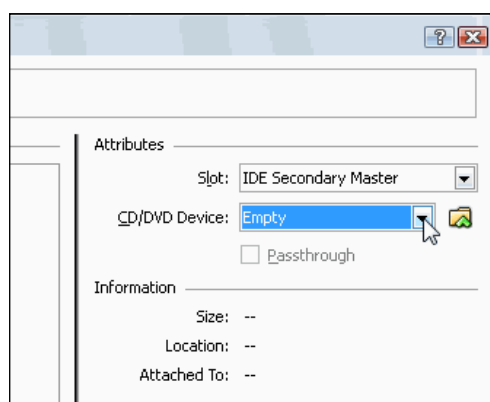
Time for action – removing installation media from your VM

Removing CD/DVD media from a virtual machine is as easy as if it were a real PC. In this exercise, you'll get to see it for yourself.

1. Make sure your Windows 7 virtual machine is shut down, and click on the **Settings** button to open the **Windows7VB – Settings** window.
2. Select the **Storage** category from the left panel, and then select the CD/DVD drive slot under the **IDE Controller** in the **Storage Tree** panel:



3. Now go to the **Attributes** panel, and select the **Empty** item from the **CD/DVD Device** list box:



4. Click on **OK** to close the **Windows7VB - Settings** dialog.

What just happened?

Now you can see I wasn't lying about how easy it is to remove media from your virtual machine, right? You can use your real CD/DVD drive just as if it were directly connected to the virtual machine, and you can also use ISO images as if they were real CD/DVD media. That's one of the benefits of virtualization: you can virtualize your physical installation media, too!

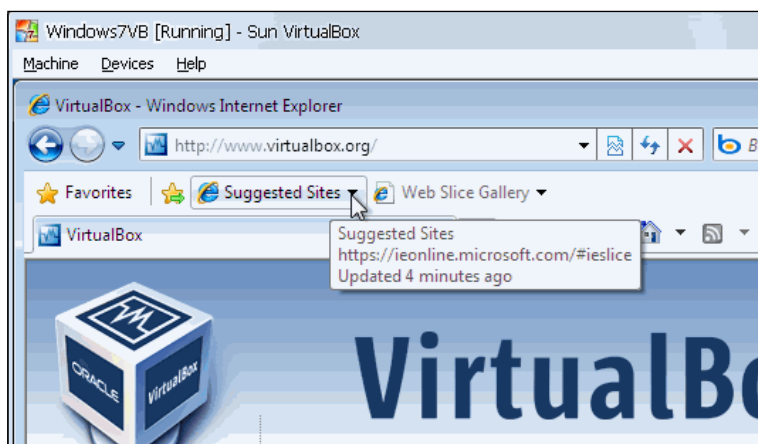
Web browsing with Internet Explorer

Windows 7 includes Internet Explorer 8 out of the box. In the following exercise, we'll see this web browser in action.

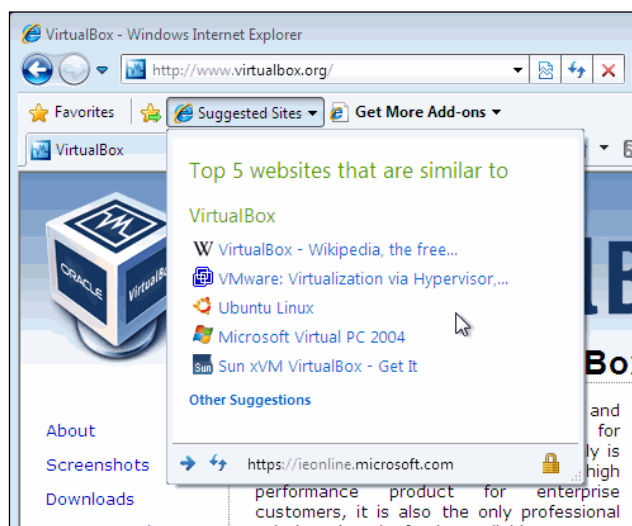
Time for action – web browsing in your Windows 7 VM

You have your virtual machine installed. Now what? Let's surf the web! After all, what could be more important than that?

1. Start your Windows 7 virtual machine, and open Internet Explorer. The `MSN.com` webpage will appear next.
2. If this is the first time you have run Internet Explorer, the **Welcome to Internet Explorer 8** screen will pop up to ask if you want to learn about its new features. Click on **Next** to continue.
3. The **Turn on Suggested Sites** screen will ask if you want IE to make personalized web suggestions based on the websites you've visited. Select **Yes**, and click on **Next** to continue.
4. The **Choose your settings** screen will pop up next. Choose the **Use express settings** option; you'll be able to change your mind later if you wish. Click on **Finish** to continue.
5. Now you'll be able to see the `MSN.com` webpage contents; type `http://www.virtualbox.org` in the address bar to go to the VirtualBox website, and then click on the **Suggested Sites** button:



6. The Suggested Sites window will open up. If the window is empty, click on the **Suggested Sites** link to see some website suggestions:



7. Click on one of the suggested links to go to that webpage. Leave your virtual machine open for the next exercise.



If you cannot connect to Internet from your virtual machine, check your host's network settings. If you can connect from your host, try using another virtual network adapter type in your virtual machine to see if the problem disappears. In Chapter 6, *Networking with Virtual Machines*, you'll find out how to change the network adapter type in your virtual machines.

What just happened?

Well, I don't think this exercise is really hard, right? But this is a cool way to test your new virtual machine to see if it has Internet enabled by default. Later on we'll talk about the different settings related to virtual network interfaces and VirtualBox. For now, it's good to know we can surf the web! Now let's see how you can do some real work inside your Windows 7 VM.

Have a go hero – using other web browsers

How about installing Mozilla Firefox and Google Chrome to compare them with Internet Explorer 8? You can get Mozilla Firefox from <http://www.mozilla.com/firefox> and Google Chrome from <http://www.google.com/chrome>.

Have a go hero – instant messaging from your virtual machine

Go to <http://download.live.com> to download and install the most recent Live Messenger version, and test it to see if you detect any difference when sending instant messages from your virtual machine and from your host PC.

Using Microsoft Office in your virtual machine

Ok, we have Internet enabled on our Windows 7 virtual machine, what else could we ask for? How about some word processing, a spreadsheet and some presentations, for starters? I know it's boring, but some of us also use VirtualBox to work!

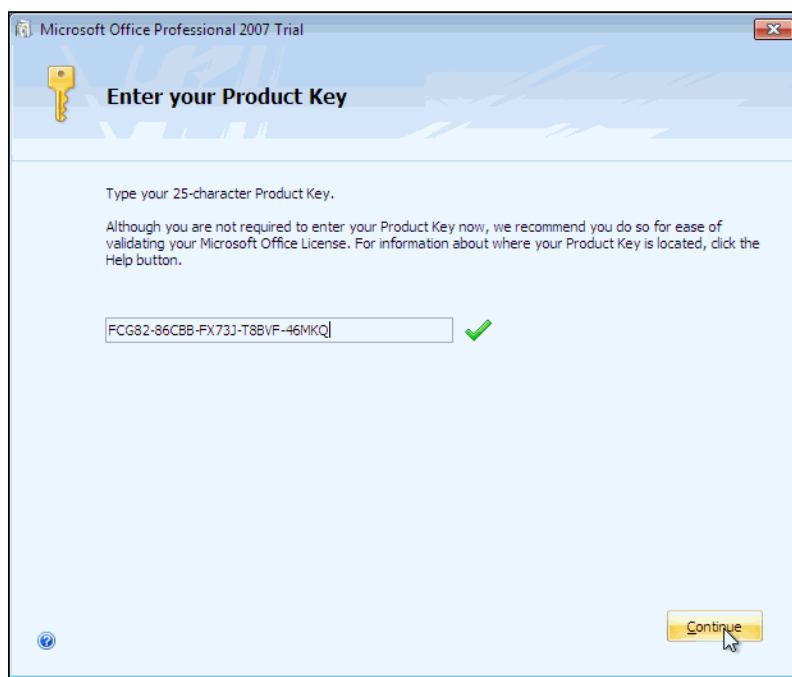
Time for action – using OpenOffice.org

Windows 7 doesn't come with Microsoft Office, but you can download a 60-day trial version from the Microsoft downloads website.

1. Open your web browser, go to <http://www.trymicrosoftoffice.com>, and click on the **Download free-trial** button:

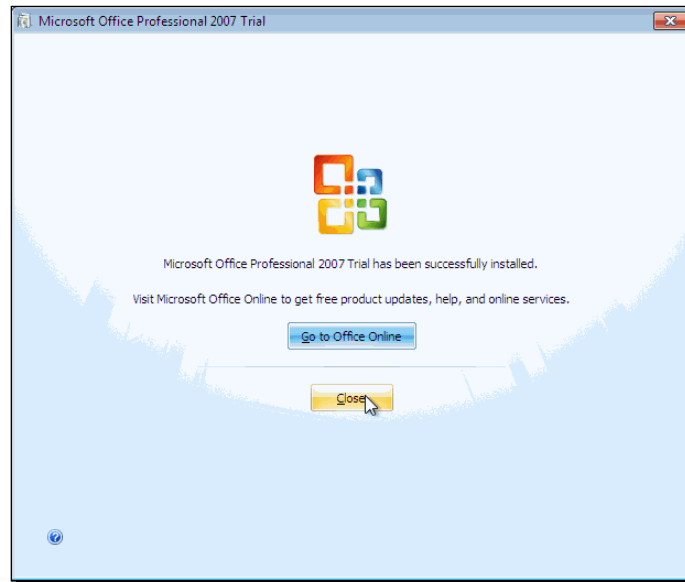


2. Select the **English** language on the next webpage, and click on the **Download now** button to continue. You'll need to login to your Windows Live ID account. If you don't have one, go on and get it! Once you're logged in, follow the instructions on-screen to download your trial copy of Office, and wait for it to download.
3. Click on **Yes** every time a **User Account Control** dialog appears to ask if you want to allow the Office 2007 installation program to run, and follow the instructions on-screen to finish installing your Office 2007 trial copy.
4. Wait until the following dialog shows up:



5. Type the Product Key you received when registering to download the Microsoft Office 2007 trial edition, and click on the **Continue** button.
6. The **Software License** dialog will appear next. Read it and, if you agree, enable the **I accept the terms of this agreement** checkbox, and click on the **Continue** button.
7. The **Choose the installation you want** dialog will show up next. Since this is the first time you're installing MS Office 2007 on your Windows 7 virtual machine, click on the **Install Now** button to continue.
8. The **Installation Progress** dialog will appear. Now you just need to wait until the setup program finishes installing your MS Office 2007 trial edition.

9. The following dialog will show up when the installation process completes:



10. Click on **Close** to exit the Office 2007 setup program. Now click on **Start | All Programs | Microsoft Office | Microsoft Office Word 2007** to open the *Microsoft Word* application.
11. The **Microsoft Office Activation Wizard** dialog will appear the first time you open a Microsoft Office application. Click on the **Next** button to continue, then select the **I want to activate the software over the Internet** checkbox, and click on **Next** again to continue.
12. The Activation Wizard will show the date of expiration of your Microsoft Office 2007 trial copy. Click on **Close** to exit the wizard.
13. Now the **Privacy Options** dialog will show up. You can click on **Next** to continue, then select the **Download and install updates from Microsoft Update when available** option, and click on **Finish**.
14. You can now use the Microsoft Word, Excel, or Power Point trial versions for 60 days on your virtual machine! And they will be updated, thanks to Microsoft Update!

What just happened?

How about that? A complete office productivity suite inside your main PC! And Internet access too! So, if you always wanted to learn about Windows 7 or any other operating system but were afraid of messing up your main PC, VirtualBox has come to your rescue!

Now let's see how to turn off your virtual machine...

Have a go hero – trying out Microsoft Office Live

Now that you have a Windows 7 virtual machine, why don't you try out the Microsoft Office Live service where you can store, share, and work with your Microsoft Office documents from a centralized web location? And on top of that, it's free! To open an account, go to the <http://www.officelive.com> website, and create a free account.

Have a go hero – sharing information between your VM and your host PC

Use your Microsoft Office Live account to share information between your Windows 7 virtual machine and your host PC, and experiment with all the features available.

Pop quiz – running your Windows 7 VM

1. Can you use the Mozilla Firefox web browser on your Windows 7 virtual machine?
 - a. Yes, and you can also use Internet Explorer 8 at the same time.
 - b. Can I answer this later?
 - c. No, you can only use the default Internet Explorer 8 web browser included.
2. Can you use OpenOffice on your Windows 7 virtual machine?
 - a. Yep, you can even use it at the same time as Microsoft Office 2007!
 - b. Nope, you cannot install another productivity suite besides Microsoft Office 2007.
 - c. There is no such thing as an OpenOffice productivity suite!
3. Mr. Jones and his wife need to share the same Windows XP computer due to financial problems, but they don't want to mess up each other's files and applications. Mr. Jones is a freelance NET programmer who needs to work with Windows 7, and Mrs. Jones needs MS Word and Windows XP for her daily tasks. Could Mr. Jones use a Windows 7 virtual machine to keep all of his work isolated from his wife's files?
 - a. No.
 - b. That would be impossible!
 - c. Yes, that would be a cool idea!

Shutting down your virtual machine

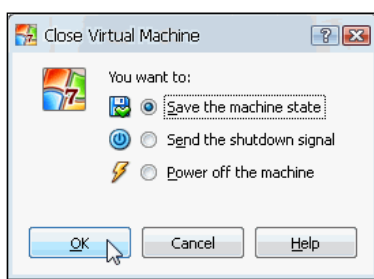
I bet you're thinking that it's silly to waste some of this book's valuable space just to show you how to shutdown a virtual machine, right?

Well, I just have to say that you need to consider that we're talking about a virtual machine here, not a real PC! And there are some useful things you could learn from reading this section about shutting down virtual machines!

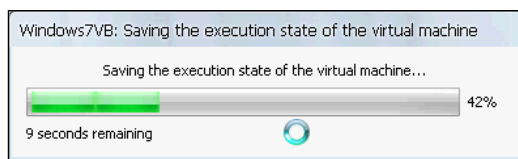
Time for action – shutting down your VM

Now it's time to stop whining and start learning how to shut down your virtual machine...

1. Make sure you close all the applications inside your virtual machine, open the Internet Explorer Web browser, 'uncapture' your keyboard and mouse so you can move to the VirtualBox main menu, and select the **Machine | Close** option.
2. The **Close Virtual Machine** dialog will show up with three choices for shutting down your virtual machine. Select the **Save the machine state** option, and click on **OK** to continue:



3. VirtualBox will show the following dialog before shutting down the virtual machine:



4. Once your virtual machine's state is saved, it will shut down automatically, and you'll be returned to the VirtualBox main screen. Also, your **Windows7VB** virtual machine status indicator will change from **Powered Off** to **Saved**. If you start it again, it will continue exactly where it was left off, with the Internet Explorer web browser window open.
5. Start your **Windows7VB** virtual machine again, leave the Internet Explorer window open, and select the **Machine | Close** menu option from the VirtualBox main menu again to open the **Close Virtual Machine** dialog.
6. This time select the **Send the shutdown signal** option, and click on **OK** to continue. This has the same effect as if you had pressed the power off button on a real PC. Your Windows 7 virtual machine will shut down automatically. If you later start it again, the Mozilla Firefox window won't be open.

What just happened?

You must admit it was everything but a simple 'shutdown your virtual machine' exercise, don't you agree? When shutting down a virtual machine in VirtualBox, you have three choices. The first one—**Save the machine state**—is kind of like suspending a laptop computer because the state of your virtual machine is saved until you later start it again. Then it resumes normal operation as if nothing had happened. All the applications you left open will still be there.

The second choice, **Send the shutdown signal**, acts as if you pushed the power button on a real machine. Windows 7 and most modern operating systems will try to use a proper shutdown mechanism when you select this option.

The third choice, **Power off the machine**, is like pulling the power plug on a real PC, so you need to be careful when choosing it because you could lose sensitive data! I only use this option when a virtual machine stops working, and I can't shut it down with the other two choices...



Always try to use the guest operating system interface first to shutdown the virtual machine. If that fails, you can use the **Send the shutdown signal** option instead, but, don't forget, it's better to treat your virtual machine as if it were a real PC.

Remember that the **Send the shutdown signal** option only works if your guest operating system supports ACPI events. For example, this option is useless when you press **F8** in your virtual machine to enter the Windows Safe Boot Mode.

Pop quiz – your first VM

1. How could you avoid having to share your only 24" LCD screen with your Linux and your Windows 7 computers? It's really annoying when you need to shut down your Linux PC, disconnect your LCD screen from it, and then connect the LCD screen to the Windows 7 computer so you can work on your Word and Excel reports!
 - a. Use VirtualBox to get rid of dual-booting so you can have Linux Ubuntu as the host operating system and Windows 7 as a virtual machine.
 - b. Call the Fonz for help! Heeyyy!
 - c. Go to <http://www.amazon.com/> and buy two new, more powerful computers.
2. Can you use a Windows 7 virtual machine inside a Windows 7 host?
 - a. No, because you can't use the same guest operating system as your host; you must use a different operating system for your virtual machines.
 - b. Only when you're using a recent PC.
 - c. Sure, that's what VirtualBox is for!
3. What is the best way to shut down a virtual machine?
 - a. Click on the **Close(X)** button of your virtual machine window.
 - b. Shut it down using the guest operating system interface, as if it were a real PC.
 - c. Use the **Power off the machine** option from the **Close Virtual Machine** dialog.
4. The best way to lose information on your virtual machine is when you:
 - a. Select the Power off the machine option from the Close Virtual Machine dialog.
 - b. Change the default host key.
 - c. Use Windows 7 as the guest operating system.

Have a go hero – experimenting with other Windows versions

Excellent, my friend... You installed and ran successfully your Windows 7 virtual machine! Now the world can be yours. But wait! What about trying other Windows versions? It would be great if you could get a copy of Windows Vista, Windows XP, or Windows 2000. Or you could try installing a virtual Windows 2003 server! Heck, you could even try Windows 98, Windows 95, or Windows 3.1!

You can also try creating virtual machines for Linux distributions, such as Ubuntu, Slackware, Fedora, Debian, and so on, to see which one is better for you...

Summary

I hope you enjoyed the exercises in this chapter, especially if you're using—or planning to use—Windows 7 for your everyday tasks. Here you learned about the basic settings needed to create a virtual machine, install the Windows 7 operating system, run the virtual machine, and shut it down appropriately.

Specifically, we covered:

- ◆ How to configure a new virtual machine in VirtualBox
- ◆ How to adjust your virtual machine's basic settings to install Windows 7 on it
- ◆ How to install the Windows 7 operating system on your VM
- ◆ How to use the VirtualBox host key, and to 'capture'/'uncapture' the mouse and keyboard in your virtual machine
- ◆ How to run your Windows 7 virtual machine and test some basic functions like Internet access and installing the Microsoft Office 2007 trial edition
- ◆ The different choices available when shutting down your virtual machine

Now that you are capable of creating a virtual machine based on the Windows 7 distribution, let's see how to adjust some advanced settings in your virtual machines and use the VirtualBox Guest Additions to maximize your virtual machine's potential in the following chapter...

4

Installing Guest Additions and Advanced Settings

Now that you learned to install and run Windows/Linux virtual machines, it's time to experience the real power of using VirtualBox: Guest Additions. And what on earth is that? Well, as you may have noticed by now, the virtual machines we have been using are not quite usable as your host, right? You need to capture the keyboard and mouse every time you want to use your virtual machine and then uncapture them when you want to use an application in your host PC.

And what if you want to copy some text from your host to your virtual machine? Isn't that clumsy little screen so annoying that you can't work decently? No need to worry anymore! Guest Additions will take care of all those annoying details for us, and your host and guest systems will be closely integrated so that you can get the best interactive experience of your life! You won't even notice you're working with a virtual machine!

In this chapter you shall:

- What the Guest Additions are and how to install them on Windows, Linux, and Open Solaris virtual machines
- How to share folders among your host PC and your virtual machines
- How to use the seamless windows feature to run applications from your virtual machines side to side with applications from your host PC

- What hardware 3D acceleration is and how to activate it in your virtual machines
- How to use snapshots to revert your virtual machines to previous states so you can recover from application installation failures or other misfortunes

Introducing Guest Additions

Ok, you have been playing with a couple of virtual machines by now, and I know it feels great to be capable of running two different operating systems on the same machine, but as I said at the beginning of this chapter, what's the use if you can't share information between your host and guest systems, or if you can't maximize your guest screen? Well, that's what Guest Additions are for.

With the Guest Additions installed in a virtual machine, you'll be able to enjoy all of these:

- ◆ *Full keyboard and mouse integration between your host and your guest operating systems:* This means you won't need to use the capture/uncapture feature anymore!
- ◆ *Enhanced video support in your guest virtual machine:* You will be able to use 3D and 2D video acceleration features, and if you resize your virtual machine's screen, its video resolution will adjust automatically. Say hello to full screen!
- ◆ *Better time synchronization between host and guest:* A virtual machine doesn't know it's running inside another computer, so it expects to have 100% of the CPU and all the other resources without any interference. Since the host computer needs to use those resources too, sometimes it can get messy, especially if both host and guest are running several applications at the same time, as would be the case in most situations. But don't worry about it! Guest Additions re-synchronize your virtual machine's time regularly to avoid any serious problems.
- ◆ *Shared folders:* This is one of my favorite Guest Addition features! You can designate one or more folders to share files easily between your host and your guest, as if they were network shares.
- ◆ *Seamless windows:* This is another amazing feature that lets you use any application in your guest as if you were running it directly from your host PC. For example, if you have a Linux host and a Windows virtual machine, you'll be able to use MS Word or MS Excel as if you were running it directly from your Linux machine!
- ◆ *Shared clipboard:* This is a feature I couldn't live without because it lets you copy and paste information between your host and guest applications seamlessly.
- ◆ *Automated Windows guest logons:* The Guest Additions for Windows provide modules to automate the logon process in a Windows virtual machine.

Now that I've shown you that life isn't worth living without the Guest Additions installed in your virtual machines, let's see how to install them on Windows, Linux, and Solaris hosts.

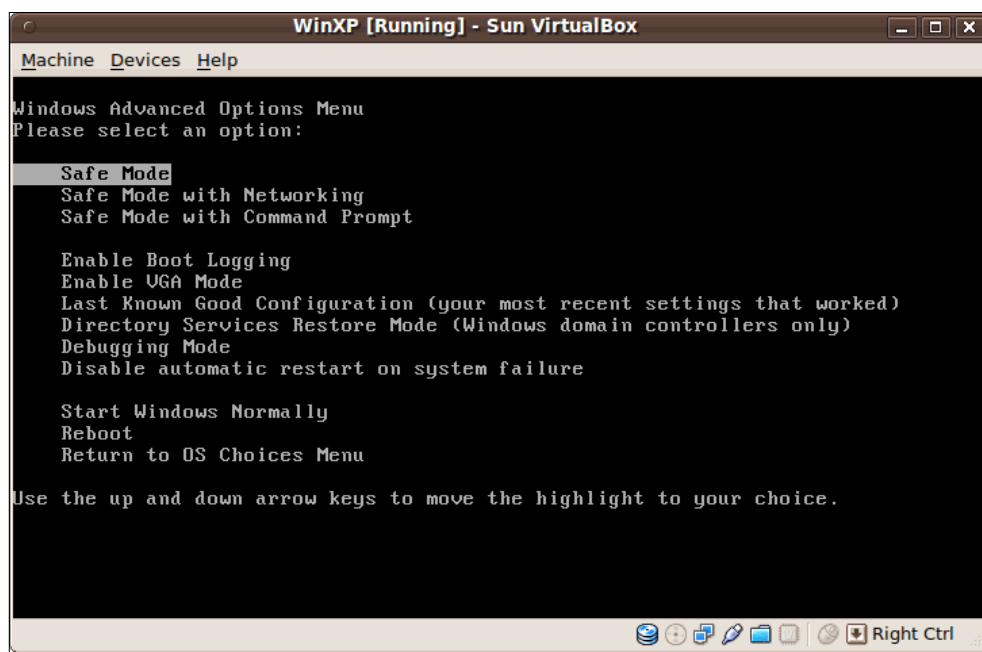
Installing Guest Additions for Windows

"Hmmm... I still can't see how you plan to get everyone in this office using VirtualBox... you can't even use that darned virtual machine in full screen!" your boss says with a mocking tone of voice. "Well boss, if you stick with me during these chapter's exercises, you'll get your two cents worth!" you respond back to him, feeling like the new kid in town...

Time for action – installing Guest Additions on a Windows XP virtual machine

In this exercise, I'll show you how to install Guest Additions on your Windows XP virtual machine so that your boss can stop whining about not being able to use the Windows VM as a real PC...

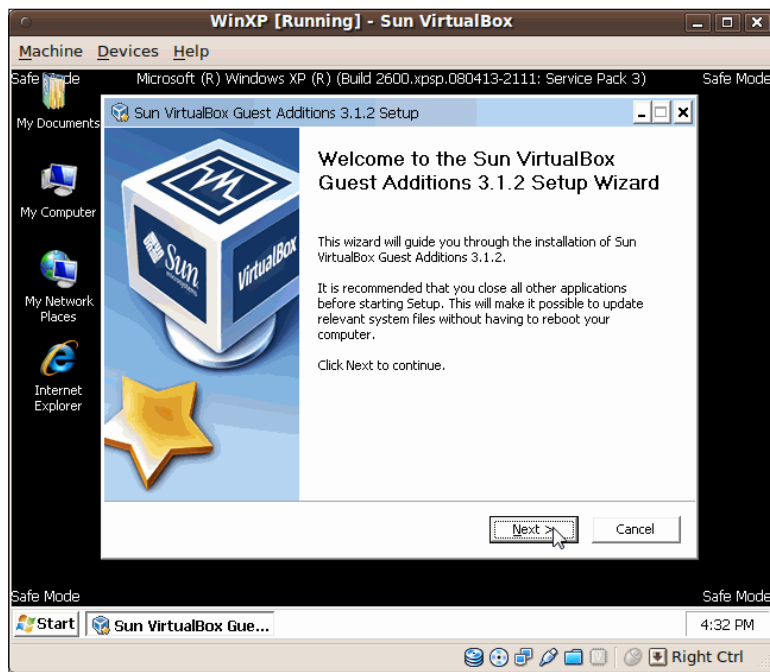
1. Open VirtualBox, and start your WinXP virtual machine. Press *F8* when Windows XP is booting to enter the **Windows Advanced Options Menu**, select the *Safe Mode* option, and press *Enter* to continue:



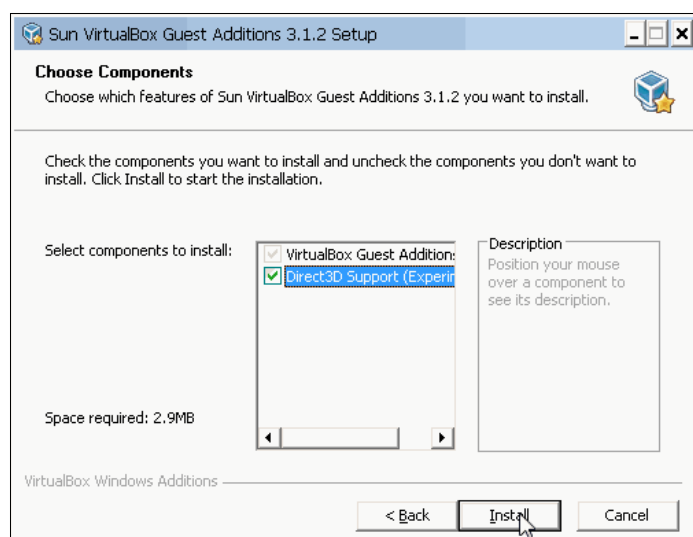
2. Wait for Windows XP to boot, and then login with your administrator account. The **Windows is running in safe mode** dialog will show up.
3. Click on **Yes** to continue, and wait for Windows to finish booting up. Then select the **Devices** option from VirtualBox's menu bar, and click on the **Install Guest Additions** option:



4. VirtualBox will mount the Guest Additions ISO file on your virtual machine's CD/DVD-ROM drive, and the Guest Additions installer will start automatically:



5. Click on **Next** to continue. The **License Agreement** dialog will appear next. Click on the **I Agree** button to accept the agreement and continue.
6. Leave the default destination folder on the **Choose Install Location** dialog, and click on **Next** to continue.
7. The **Choose Components** dialog will appear next. Select the **Direct 3D Support** option to enable it, and click on **Install** to continue:



8. Guest Additions will begin to install in your virtual machine.
9. The next dialog will inform you that the Guest Additions setup program replaced some Windows system files, and if you receive a warning dialog from the Windows File Protection mechanism, you need to click on the **Cancel** button of that warning dialog to avoid restoring the original files.
10. Click on **OK** to continue. The setup program will ask if you want to reboot your virtual machine to complete the installation process. Make sure the **Reboot now** option is enabled, and click on **Finish** to continue.
11. Your WinXP virtual machine will reboot automatically, and once it has finished booting up, a **VirtualBox - Information** dialog will appear, to tell you that your guest operating system now supports mouse pointer integration.
12. Enable the **Do not show this message again**, and click on the **OK** button to continue. Now you'll be able to move your mouse freely between your virtual machine's area and your host machine's area! You'll also be able to resize your virtual machine's screen, and it will adjust automatically!

What just happened?

Hey, it was pretty simple and neat, huh? Now you can start to get the real juice out of your Windows virtual machines! And your boss will never complain again about your wonderful idea of virtualizing your office environment with VirtualBox!

The Guest Additions installation process on Windows virtual machines is a little bit more complicated than its Linux or OpenSolaris counterparts due to the fact that Windows has a file protection mechanism that interferes with some system files VirtualBox needs to replace. By using 'Safe Mode', VirtualBox can override the file protection mechanism, and the Guest Additions software can be installed successfully. But if you don't want or need 3D guest support, you can install Guest Additions in Windows, normal mode.

Installing Guest Additions for Linux

Ok, in the previous section, you saw how to install Guest Additions on a Windows virtual machine. Now go and show your boss the real benefit of using a Linux virtual machine on a Windows XP host PC!

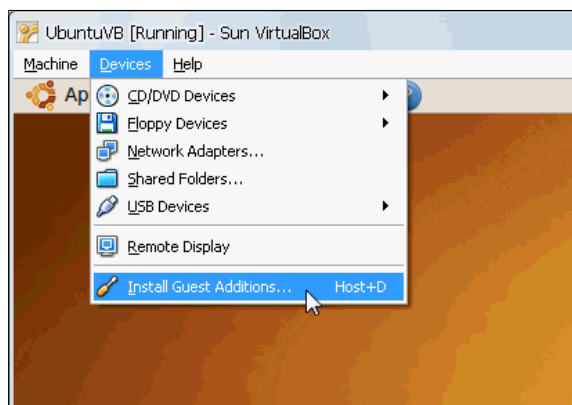


In the following exercise, I'll use the `UbuntuVB` virtual machine we created in Chapter 2, *Creating your First Virtual Machine: Ubuntu Linux*, of this book.

Time for action – installing Guest Additions on Linux Ubuntu

In this exercise, I'll show you how to install Guest Additions on your Linux Ubuntu virtual machine. I'm using a Windows XP host, but it doesn't matter if you're using Windows, Linux, or Solaris as the host operating system because the process to install Guest Additions on a Linux virtual machine is nearly the same!

1. Open VirtualBox, and start your `UbuntuVB` virtual machine. After logging into Ubuntu with your user account, open a terminal window, and type `sudo apt-get install dkms`, followed by *Enter*. Then follow the instructions on screen to install the DKMS package. Once installed, close the terminal window.
2. Select **Devices | Install Guest Additions** from the VirtualBox menu:



3. VirtualBox will mount the Guest Additions ISO file on your virtual machine's CD/DVD-ROM drive, and the **VBOXADDITIONS** icon will appear on your desktop.
4. Now open a terminal window, and type `cd /media/cdrom` followed by *Enter* to change to the CD-ROM directory containing the Guest Additions. Then type `sudo sh ./VBoxLinuxAdditions-x86.run`, and press *Enter* to start installing the VirtualBox Guest Additions on your virtual machine. When finished, the setup program will ask you to restart your Ubuntu VM.



If you're running a 64-bit Linux guest, replace `VBoxLinuxAdditions-x86.run` with `VBoxLinuxAdditions-amd64.run`.

5. Close the terminal window, restart your virtual machine, and wait for Ubuntu to boot up. Once it has finished booting up, a **VirtualBox - Information** dialog will appear to tell you that your guest operating system now supports mouse pointer integration.
6. Enable the **Do not show this message again**, and click on the **OK** button to continue. Now you'll be able to move your mouse freely between your virtual machine's area and your host machine's area! You'll also be able to resize your virtual machine's screen, and it will adjust automatically!

What just happened?

Ok, that was pretty quick, right? I bet you're wondering why you waited for so long to try out Linux, despite the good things some of your 'geeky' friends kept telling you about! Well my friend, now's the time to vindicate yourself by exploring the fantastic world of VirtualBox and Linux virtual machines!

As you may have noticed by now, I'm using Ubuntu on every exercise related to Linux. I've used several Linux distributions during the last 10 years or so, but I decided to stick with Ubuntu for this book's exercises because of the great friendliness it offers to first-time users. If you've never used Linux before, Ubuntu is (in my humble opinion) the best way to enter the Linux world, and if you've used Linux before but haven't tried Ubuntu yet, by the time you finish this book, you'll understand why I turned into such a big fan of this excellent Linux flavor.

Now let's see some tiny details before advancing to the next exercise. In step 1 of the previous exercise, we used the `sudo apt-get install dkms` command to install the `dkms` package on your Ubuntu virtual machine. Since I don't want to bore you to death with all the details involved with this package, let's just say that `dkms` stands for Dynamic Kernel Module Support, and this tiny little piece of software takes care of maintaining the Linux so that your virtual machine can operate without any problems.



If for any reason DKMS isn't available for your Linux distribution, you can still run the Guest Additions. Just be sure to execute `sudo/etc/init.d/vboxdrv setup` everytime you update your kernel, or re-install the Guest Additions.

What happens if you want to use Fedora, Red Hat, Slackware, or any other Linux distribution instead of Ubuntu? Well, for starters, let me tell you that, at the time of this writing, the most popular Linux distributions officially supported by VirtualBox are these:

Linux distribution	Versions
Fedora	Fedora Core 4 and later
Red Hat Enterprise Linux	3 and later
SuSE and openSUSE Linux	9 and later
Ubuntu	5.10 and later
Mandriva	2008 and later
Mandrake	10.1
Slackware	10.1

For a complete list of supported guest operating systems, visit the http://www.virtualbox.org/wiki/Guest_OSes web page.

Of course, that doesn't mean you can't use a Linux distribution not officially supported by VirtualBox. You'll just have to be careful because nobody knows if Guest Additions will work flawlessly or not.



Not all Linux distributions accept the `sudo apt-get install dkms` command for installing the DKMS package. For example, in Fedora you'll need to use `yum install dkms`, and with Mandriva you'll need to use `urpmi dkms`. In fact, openSUSE doesn't support DKMS, and it's a very popular distribution! For more information on how to install Guest Additions on Linux distributions without DKMS support, visit the VirtualBox official forum at <http://forums.virtualbox.org/index.php> or send me an e-mail at questions@packtpub.com. I'll be more than glad to help!

Installing Guest Additions for OpenSolaris

That's two down, one more to go! OpenSolaris is the third operating system I'll cover in this chapter. This is the free open source alternative to the popular Solaris operating system, so the steps to install Guest Additions on OpenSolaris apply also to Solaris.

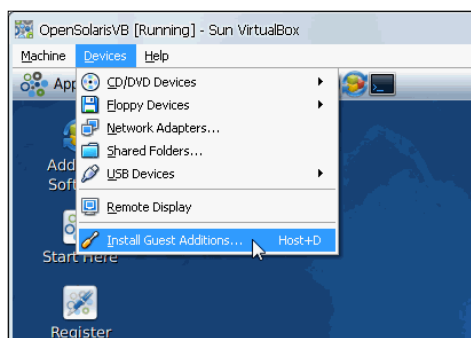


In the following exercise, I'll use an OpenSolaris virtual machine named OpenSolarisVB on a Windows XP host. You can download OpenSolaris from <http://www.opensolaris.com/get/index.jsp>, create a new virtual machine, and install OpenSolaris on it to follow the exercise.

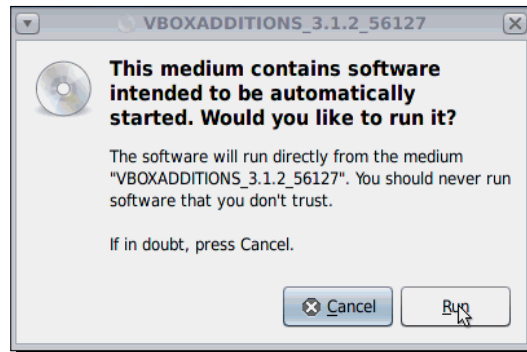
Time for action – installing Guest Additions on OpenSolaris

Ok, you showed your boss how to virtualize two of the most popular operating systems around. If you haven't tried out OpenSolaris, this is your chance to learn about it, and if you've used it before, this is your chance to compare it with Windows and Ubuntu in a virtualized environment!

1. Open VirtualBox, and start your OpenSolarisVB virtual machine. After logging into OpenSolaris with your administrator account, select **Devices | Install Guest Additions** from the VirtualBox menu:



2. VirtualBox will mount the Guest Additions ISO file on your virtual machine's CD/DVD-ROM drive. The VBOXADDITIONS icon will appear on your OpenSolaris VM desktop, along with the following dialog:



3. Click on the **Run** button to start installing the Guest Additions. The **Installing VirtualBox Additions** dialog will appear to show the progress of the installation process. Wait until the installation process finishes.
4. Press *Enter* to close the Installing VirtualBox Additions window when it tells you to do so, then select **System | Shutdown** from the OpenSolaris menu, and click on the **Restart** button in the **Shut down this system now?** dialog to restart your virtual machine.
5. The next time your OpenSolaris VM boots up, you'll be able to move your mouse freely between your virtual machine's area and your host machine's area! You'll also be able to resize your virtual machine's screen, and it will adjust automatically!

What just happened?

As you may have noticed, OpenSolaris looks a lot like Ubuntu Linux. That's because both operating systems use the Gnome desktop manager. You can find out more about Gnome at <http://www.gnome.org/> and about OpenSolaris at <http://www.opensolaris.org>.

Now you have the opportunity to test Windows, Ubuntu Linux, and OpenSolaris with a single PC! And all thanks to the virtualized environment created by VirtualBox! In the following sections, you'll learn to use several Guest Additions features that will provide a closer integration between your host PC and your virtual machines.

Pop quiz – Guest Additions

1. You need to use Internet Explorer to access certain web application in your company's LAN because it doesn't work with Mozilla Firefox. The only problem is that you are using an Ubuntu Linux PC! What can you do?
 - a. Throw your Ubuntu PC, and order a brand new Vista machine from Amazon.
 - b. Back up all your important data and applications, reformat your hard drive, and install Vista.
 - c. Install VirtualBox on your Ubuntu PC, and create a Windows virtual machine with IE installed.
2. You need to use Microsoft Word to write a user's manual for a software application running in a Linux environment on your company's server. You can:
 - a. Run back and forth between your Word document and the Linux system to capture screenshots and write your tutorial.
 - b. Install VirtualBox on your Windows PC and create a Linux virtual machine to run the software application, take screenshots, and write your tutorial on MS Word.
 - c. Open Solitaire on your Windows PC, and play all day long...
3. The company you work for is planning to get a new application server for the internal LAN, but the company isn't sure about what operating system to use for it: some people want to use OpenSolaris, others want to use a traditional Windows 2008 server, and there are several Linux worshippers that want to use Ubuntu. What can you do to help them make their decision?
 - a. Roll some dice or play 'Rock, Paper, Scissors' to decide between each operating system.
 - b. Use VirtualBox with one of the three operating systems as the host, and create two virtual machines for the other two operating systems so they can test the three of them to see which one suits their needs best.
 - c. Tell your co-workers to buy one server for each operating system, test which one can suit their needs best, and then sell the other two.

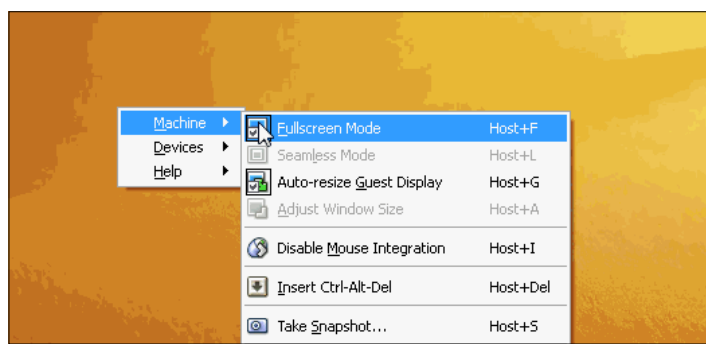
Using the fullscreen feature

One of the immediate benefits of installing the Guest Additions in your virtual machines is the ability to run them in your host PC's whole screen as if they were the host PC. In the following exercise, I'll show you how to alternate between the **fullscreen** and the **windowed** modes in a Windows XP host and an Ubuntu Linux guest.

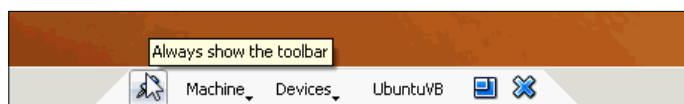
Time for action – using the fullscreen and windowed modes

There are two ways to alternate between the fullscreen and windowed modes: using the **Machine** menu and pressing *Host+F* on your keyboard (where Host represents the currently defined host key, the default value of which is *Right-Ctrl*).

1. Open your UbuntuVB virtual machine, and select **Machine | Fullscreen Mode** from the VirtualBox main menu. A **VirtualBox – Information** dialog will pop up to inform you the virtual machine will switch to full-screen and that you can go back to the windowed mode by pressing *Host+F*.
2. Select the **Do not show this message again** option, and click on **Switch** to enter the fullscreen mode. Your virtual machine's screen will resize automatically to full-screen mode, and you will be able to interact with your VM as if it were the host PC.
3. When in your machine is fullscreen mode, the Virtual menu bar is hidden. Press *Host+Home*, and a context menu will pop up to show the three menu items from the VirtualBox main menu bar:



4. Select **Machine | Fullscreen Mode** from the pop-up menu to return to the windowed mode. Now press *Host+F* to enter the fullscreen mode again, and then move your mouse pointer to the bottom-central part of the screen. The VirtualBox toolbar will show up next:



5. As you can see, this toolbar has the same options as the pop-up menu that appears when you hit *Host+Home*. The first icon lets you alternate between always showing the bar and hiding it until you move the mouse pointer over the area where the toolbar is hiding. The **Machine** and **Devices** menus are the same ones as in the VirtualBox main screen. The fourth item in the menu bar is the name of your virtual machine. The fifth item is the **Exit Fullscreen or Seamless Mode** button, and the last one is the **Close VM** button; it has the same effect as selecting **Machine | Close** from your virtual machine main menu.
6. Click on the **Exit Fullscreen or Seamless Mode** button to return to the windowed mode.

What just happened?

Being able to run your virtual machine at fullscreen mode is one of the most useful features you can have after installing the Guest Additions. And thanks to the *Host+F* combination, you can alternate between the fullscreen and windowed modes. In the previous exercise, you saw how to use the *Host+F* combination along with *Host+Home* and the VirtualBox mini toolbar to operate your virtual machine in both fullscreen and windowed modes.

You can also access the fullscreen mode from the **Machine | Fullscreen Mode** of your virtual machine's main menu. And the **Machine | Auto-resize Guest Display** option of the same menu lets you alternate between automatically resizing your virtual machine's screen when the window is resized or keeping the same screen size even if you try to resize your virtual machine's window.

Have a go hero – adjusting the VirtualBox Mini Toolbar settings

There are two settings regarding the VirtualBox mini toolbar that you can change in the **Advanced** tab from the **General** category. The first setting is **Show In Fullscreen/Seamless**; if it's checked, the Mini Toolbar will show up in the fullscreen/seamless modes. The other setting is **Show At Top Of Screen**; if it's checked, the Mini Toolbar will show up at the top of the screen instead of the default position at the bottom of the screen. Play with these settings to see which option best suits your needs. Experiment with different virtual machines because each VM can have its own Mini Toolbar settings.

Pop quiz – using the fullscreen feature

1. What do you need to do in order to use the fullscreen feature in your virtual machine(s)?
 - a. Install OpenOffice
 - b. Install Ubuntu Desktop
 - c. Install the Guest Additions

2. What's the quickest way to alternate between fullscreen and windowed mode?
 - a. Hit the *Host+L* keys.
 - b. Hit the *Host+F* keys.
 - c. Maximize your virtual machine's window.
3. Your boss is very happy because he can run an Ubuntu virtual machine in fullscreen mode on his old Windows XP system, but he's starting to lose his mind due to the Mini Toolbar that shows up occasionally on his screen. "I'm pretty comfortable with the keyboard! I don't need that stinkin' mini toolbar staring at me all the time!" he says. What can you do to help him?
 - a. Exchange his computer for a brand new Ubuntu Desktop system.
 - b. Tell him to go and ask in the VirtualBox forum.
 - c. Adjust the Mini Toolbar setting in his virtual machine.

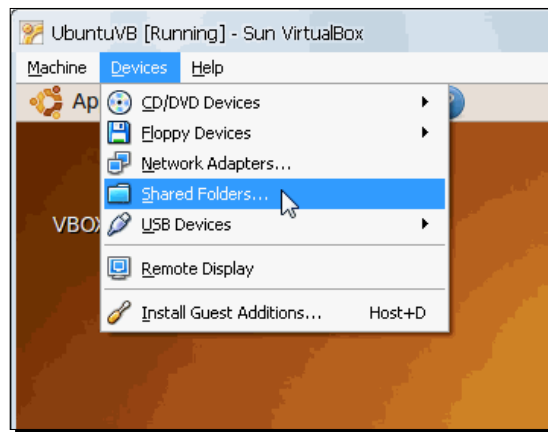
Sharing folders between your host and guest PCs

In this section, I'm going to show you how to use the folder sharing feature because sooner or later you'll need to share information between your host PC and one of your virtual machines. And thanks to the Guest Additions software, it's as easy as sharing information between your PC and another PC connected through a local area network. But in this case, you don't need a real network because VirtualBox takes care of all the connection processes for you.

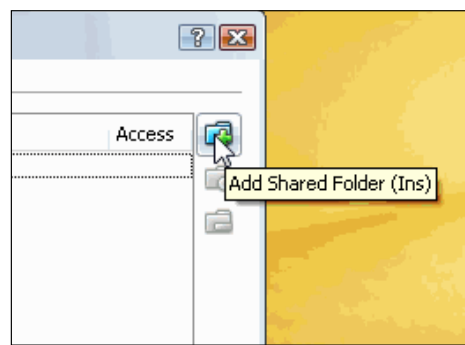
Time for action – sharing folders between a Windows XP host and a Ubuntu guest

Now I'm going to show you how to share a folder between a Windows XP host operating system and an Ubuntu Linux guest OS (the virtual machine).

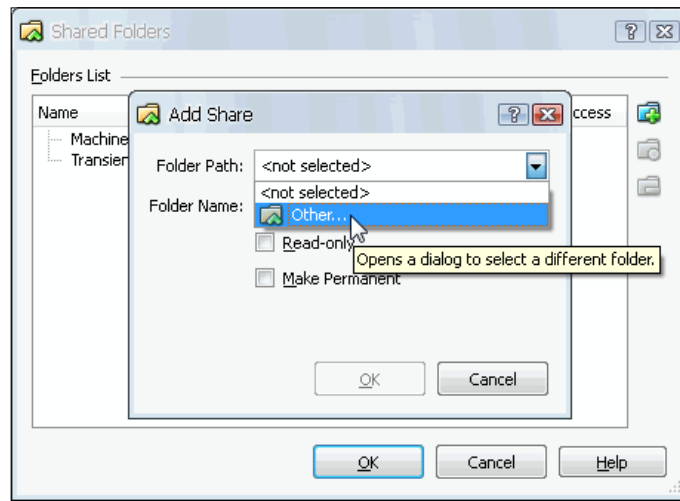
- 1.** Create a new folder on your Windows XP host, and name it `SharedVB`, then open VirtualBox, start your Ubuntu Linux VM, and log into it.
- 2.** Select **Devices | Shared Folders** from the VirtualBox main menu:



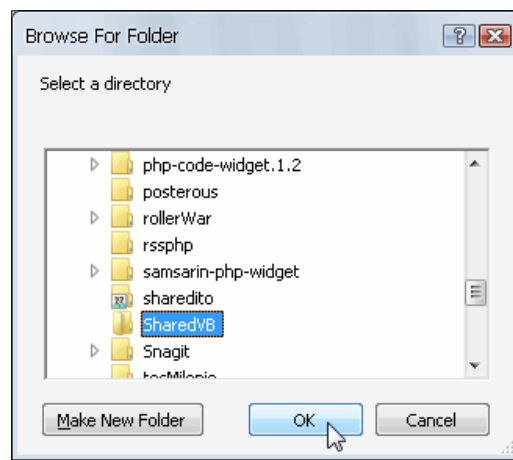
3. The **Shared Folders** dialog will show up next. Click on the **Add Shared Folder** button at the upper-right part of the dialog:



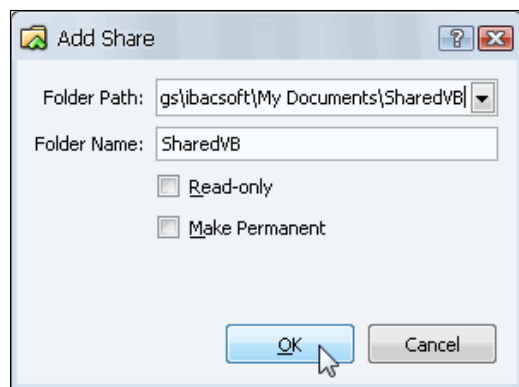
4. The **Add Share** dialog will appear next. Click on the **Folder Path** drop-down list box, and select the **Other** option:



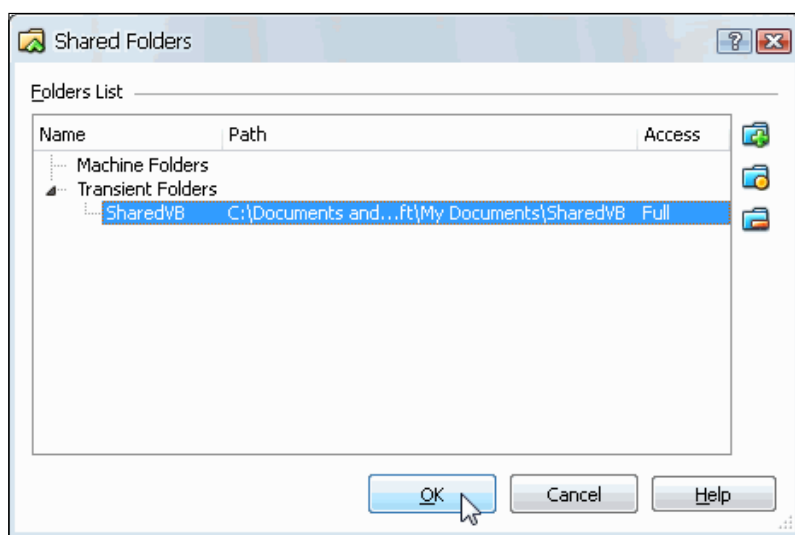
5. Locate the **SharedVB** folder on the **Browse for Folder** dialog, and click on **OK** to select it:



6. VirtualBox will return you to the **Add Share** dialog. The **Folder Path** list box will contain the **SharedVB** folder path, and the **Folder Name** field will be automatically filled in with **SharedVB**. Click on **OK** to continue:



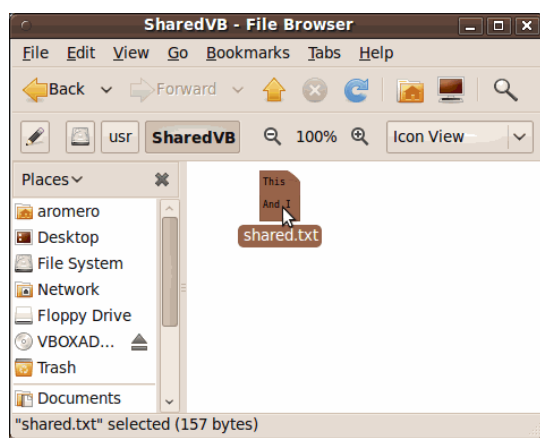
7. Now the `SharedVB` folder will be listed under the **Transient Folders** category in the **Shared Folders** dialog. Click on **OK** to continue:



8. The next step is to create a folder named `SharedVB` in your Ubuntu Linux virtual machine and mount the new shared folder on it. Select **Applications | Accessories | Terminal** from the Ubuntu menu to open a terminal window, and type the following two lines, pressing *Enter* after each one:

```
sudo mkdir /usr/SharedVB
sudo mount -t vboxsf SharedVB /usr/SharedVB
```

9. Now you can send information back and forth between your host system and your virtual machine through the SharedVB shared folder! To try it out, create a text file named `shared.txt`, and write the following information in it: This text file is for testing the shared folders feature in VirtualBox. I wrote these lines on my host PC.
10. Now save the file in the SharedVB folder of your host PC, and close it, and then go to your virtual machine's SharedVB shared folder:

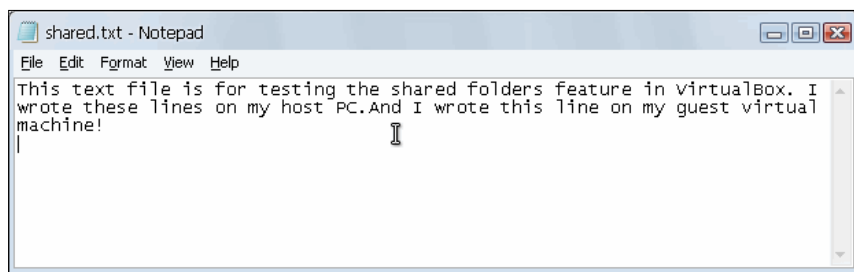


11. Right-click on the `shared.txt` file inside this directory, and select the **Open with "OpenOffice.org Word Processor"** option.



If you try to use the `gedit` application on Linux Ubuntu, you won't be able to write changes to files inside a shared folder. I hope they can correct this behavior on future versions of VirtualBox or Ubuntu.

12. The **ASCII Filter Options** dialog will appear. Click on **OK** to continue. Once the file opens, press *Enter* twice, and type `And I wrote this line on my guest virtual machine!` at the end of the file.
13. Select **File | Save** to save the changes you made to the `shared.txt` file. OpenOffice will then ask if you want to keep the current format of the file, or if you want to save it in ODF format. Click on the **Keep Current Format** button to keep the `shared.txt` file's current format, and then close OpenOffice.org Writer.
14. If you open the `shared.txt` on your host PC again, it will reflect the changes you made to it in the virtual machine:



What just happened?

Now you can share all the files you want between your host PC and a virtual machine! This is definitely another feature you won't be able to live without when doing some real work with your virtual machines.

When adding a shared folder, you have two options: make it permanent or transient. For example, in the previous exercise, the `SharedVB` shared folder is transient because if you shutdown your virtual machine and start it later, the folder will not show up. If you want to make a shared folder permanent, you need to select the **Make Permanent** option in the **Add Share** dialog (step 6 of the Sharing folders between a Windows XP host and an Ubuntu guest exercise).



If you use a transient folder to share information, don't worry about losing your data after shutting down your VM. This only means the guest virtual machine won't be able to access your host's data once you start it again.

And when you need to share a folder but don't want to let other people write stuff on it or delete its contents, you can make it read-only by selecting the **Read-only** option on the **Add Share** dialog. Note that this only affects read/write permissions on your guest; your host's permissions won't be changed.

Have a go hero – comparing file formatting characteristics between host and guest

When you are sharing files between a Windows host and a Linux guest (or vice versa), some files can lose certain formatting characteristics such as special font types, sizes, or carriage returns. This applies only when you're using different applications in the guest and in the host to open the files. Try creating one or more documents in MS Word, Excel, and PowerPoint in your Windows host, and then use the shared folder you created before to open those documents with OpenOffice in an Ubuntu guest.

Use different fonts in the same document along with bullets, tables, graphics, and everything you can think of. Then see what changes and what doesn't change during the sharing of those documents between host and guest. You can also try the other way: create some documents in OpenOffice, and then try to open them in MS Office to see if there are any differences between them.

Have a go hero – creating transient and permanent shared folders

Use the Sharing folders between a Windows XP host and an Ubuntu guest exercise as a guide to create another shared folder, but this time, make it permanent. Then see what happens when you shutdown your virtual machine and restart it. Don't forget to create or put some test files in your shared folder!

Have a go hero – creating shared folders as a regular user in an Ubuntu guest

In the previous exercise, you created a shared folder in your Ubuntu virtual machine with the `sudo mkdir /usr/SharedVB` command. This creates a shared folder named `SharedVB` inside the `usr` system directory. You need to use the `sudo` command along with the `mkdir` command because a regular user cannot create folders inside `usr`, `bin`, or any other Linux system folder. Now it's your turn to create a shared folder as a regular user:

1. Create a new folder named `MySharedVB` on your host PC.
2. Select **Devices | Shared Folders** from the VirtualBox main menu.
3. Add the new `MySharedVB` folder to your virtual machine.
4. Create and mount the `MySharedVB` folder as a regular user:

```
mkdir MySharedVB
sudo mount -t vboxsf -o uid=yourusername MySharedVB MySharedVB
```

Remember to replace `yourusername` with your Ubuntu username for the command to work correctly. The `-o uid=yourusername` part in the `mount` command tells Ubuntu you want to mount the `MySharedVB` folder using `yourusername` as the owner. Now, you'll be able to create, delete, or modify files inside the `MySharedVB` folder from your host PC or your Ubuntu guest with your regular Ubuntu user account.

Have a go hero – sharing folders between a Windows XP host and an OpenSolaris guest

Now try sharing folders between a Windows XP host and an OpenSolaris guest. The process is basically the same as sharing folders between a Windows XP host and an Ubuntu Linux guest:

1. Create a new folder on your host.
2. Select **Devices | Shared Folders** from the VirtualBox main menu.
3. Add the new `SharedVB` shared folder to your virtual machine.
4. Create a shared folder named `SharedVB` in your OpenSolaris virtual machine, and mount your host's new shared folder in it.

Just an observation: to mount the host shared folder in OpenSolaris, you need to use the `pfexec mount -F vboxfs SharedVB /usr/SharedVB` command instead of the `sudo mount` command used in Linux.

Have a go hero – sharing folders between an Ubuntu Linux host and a Windows XP guest

Now try it the other way around:

1. Create a new folder named `MyUbuntuWindows` on your Ubuntu host's home directory.
2. Select **Devices | Shared Folders** from the VirtualBox main menu.
3. Add the `MyUbuntuWindows` shared folder to your virtual machine.
4. To access your new shared folder in your Windows guest, use the Windows Explorer to browse in **My Network Places | Entire Network | VirtualBox Shared Folders**.

Pop quiz – using the folder sharing feature

1. If you don't want to lose your shared folder the next time you start your virtual machine:
 - a. Make it transient.
 - b. Make it permanent.
 - c. Use Google Maps to locate it.
2. If you shutdown your virtual machine and you have several transient shared folders, will you lose the information they contain?
 - a. Yes, because they're transient.
 - b. No, because having a transient shared folder only means you will not be able to access it the next time you start your virtual machine, but you can access it from your host PC. You'll just have to add it again using the **Devices | Shared Folders** option.
 - c. Maybe... Maybe not.

Activating the Seamless Windows feature

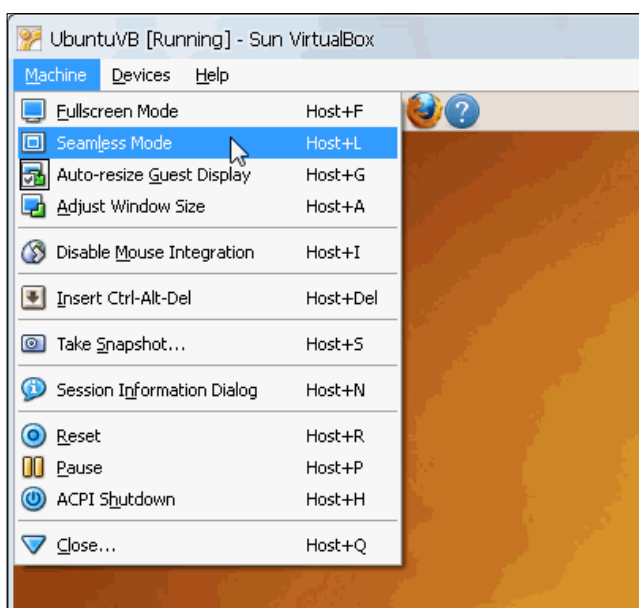
Although your boss has been happier about using VirtualBox during the last few weeks, this morning he showed up in your office and started whining again about how annoying it is for him to change between the Microsoft Word window inside his Windows XP virtual machine and the Mozilla Firefox window on his Ubuntu host PC.

Fortunately for you, this section will get you ready to fight back because the Seamless Windows feature will allow your boss to use the Microsoft Word application on his Windows XP virtual machine side by side with all the other software applications in his Ubuntu Linux host PC!

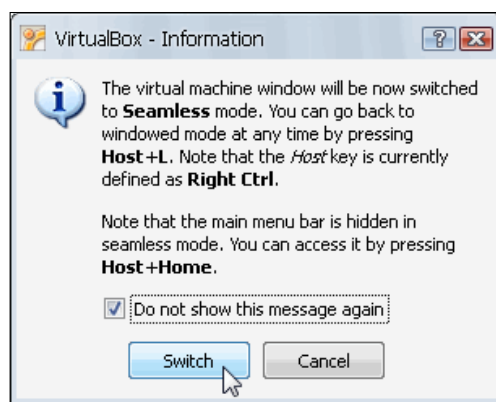
Time for action – activating Seamless Windows with Windows and Linux

In the following exercise, I'll show you how to activate the Seamless Windows feature with a Windows XP host and an Ubuntu Linux guest operating system:

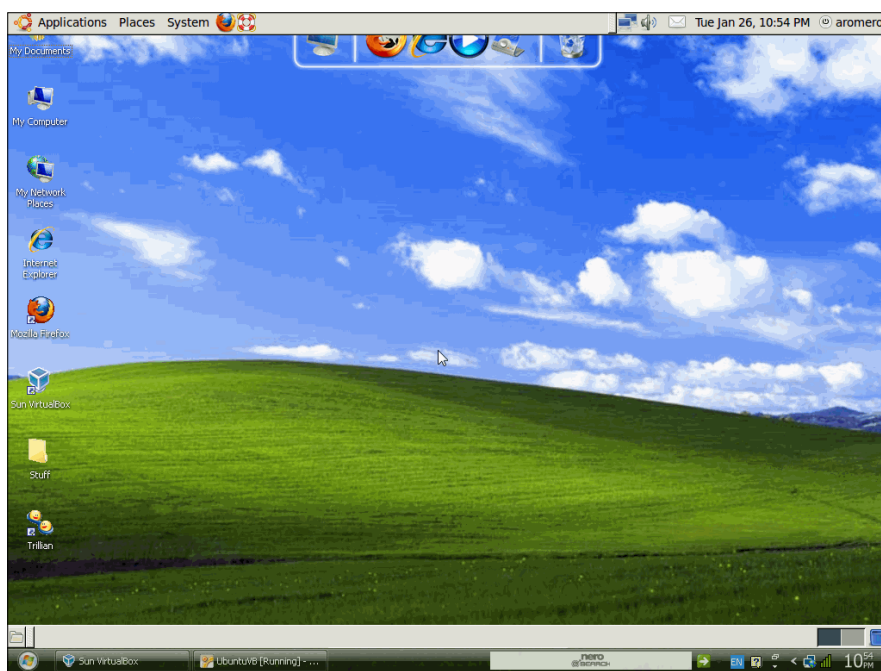
1. Start your Ubuntu Linux virtual machine, login with your administrator account, and select **Machine | Seamless Mode** from the VirtualBox menu:



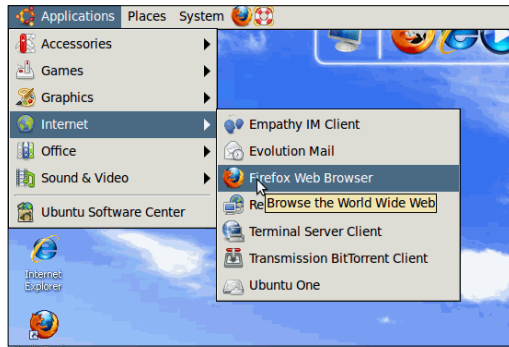
2. The **VirtualBox – Information** dialog will show up to inform you that the virtual machine will switch to **Seamless** mode. Enable the **Do not show this message again** check box, and click on **Switch** to continue:



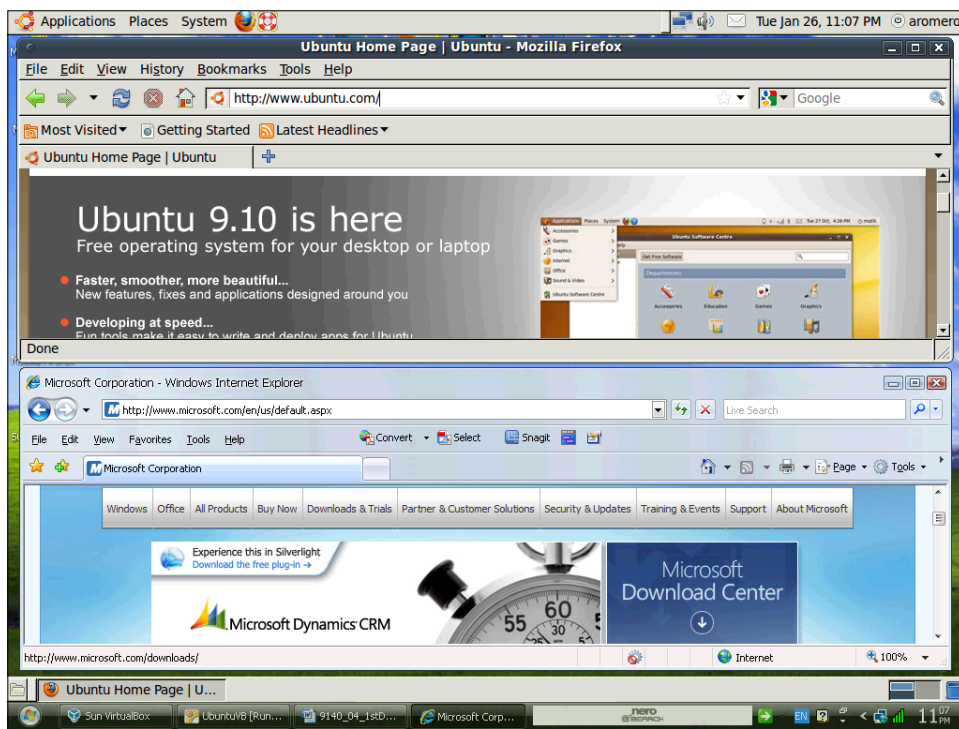
3. Your virtual machine's desktop will merge with your host PC's desktop. In the following screenshot, the host PC is running WindowsXP, and the guest VM is running Ubuntu Linux. (The Windows taskbar appears at the bottom the Ubuntu menu bar appears at the top, and the background belongs to the Windows XP host:



4. Select **Applications | Internet | Firefox Web Browser** from your Ubuntu virtual machine's menu to open Mozilla Firefox:



5. If the Mozilla Firefox window appears maximized, click on the **Restore Window** button. Adjust its window size so that it occupies the upper-half of the screen.
6. Now select **Start | All Programs | Internet Explorer** on your Windows host PC to open the IE web browser. Adjust its window size so it occupies the lower half of the screen, as shown below:



7. Now you can work with your Ubuntu applications side by side with your Windows XP applications! If you want to disable the Seamless Windows mode, just press your Host key (usually *Right Ctrl*)+ *L*.

What just happened?

This was just a little demonstration of all the things you can do with the Seamless mode in VirtualBox. Your boss will be more than happy when you teach him how to apply the Seamless Windows feature to his Windows XP virtual machine and his Ubuntu Linux host PC!

In step 2 of the previous exercise, a dialog showed up to tell you how to switch back from the Seamless Window mode. In this case, you must press the Host key and the *L* key from your keyboard. The default key in VirtualBox is *Right Ctrl*, but you can change it to any other key of your choice. To return to the Seamless Windows mode, you can press the Host key + *L* again.

Have a go hero – trying the Seamless Windows feature with other VMs

Now that you've learned how to activate/deactivate the Seamless Window mode in your VirtualBox VMs, it would be cool if you tried it out on different host and guest operating systems. For example, you can try using Microsoft Word on a Windows XP/Vista virtual machine running inside a Linux/OpenSolaris host.

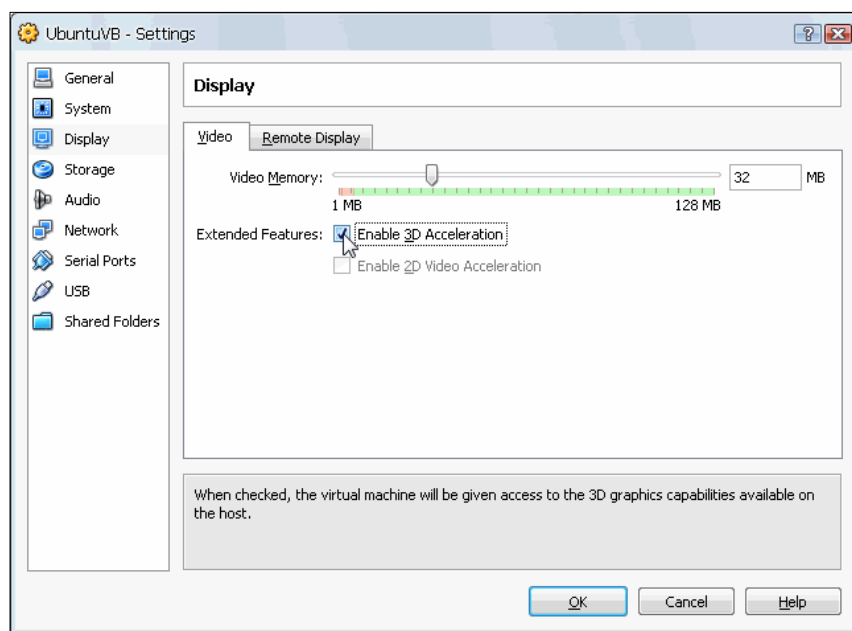
Allowing 3D Hardware Acceleration in your virtual machines

Hey, wanna show your boss a cool trick in VirtualBox? In this section, I'll show you what a VirtualBox VM can do if you have a decent graphics card. The only drawback is you need a powerful host machine with at least 2 GB of RAM and a 64 MB graphics card to share between your host and guest operating systems because 3D acceleration demands a lot of horse power on the graphics hardware components.

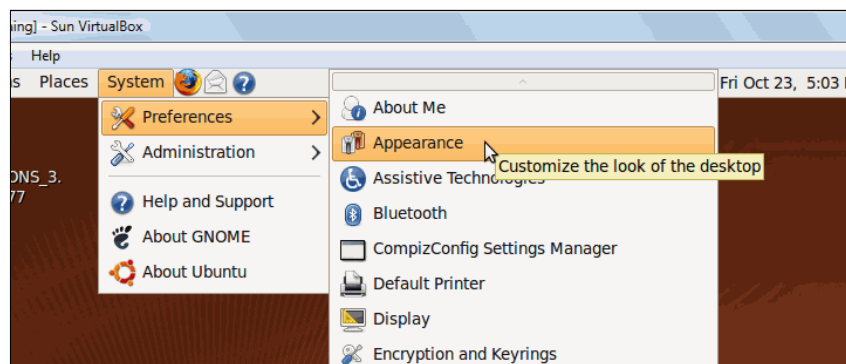
Time for action – using Compiz on your Ubuntu VM

In the following exercise, you'll learn how to use Compiz in an Ubuntu Linux virtual machine.

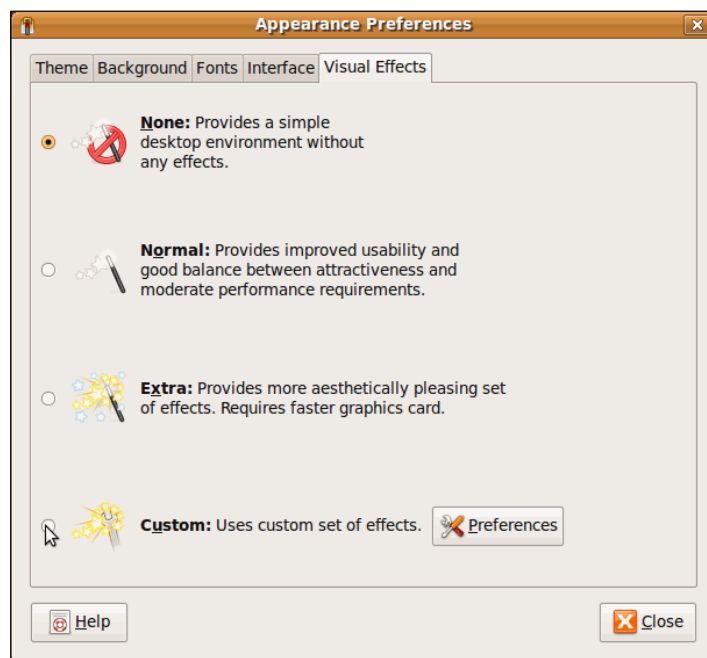
1. Start VirtualBox, select your Ubuntu virtual machine, and click on the **Settings** button to go to the **UbuntuVB – Settings** page. Select the **Display** category, click on the **Enable 3D Acceleration** check box to enable it, and adjust the **Video Memory** slider to the amount of video memory you want to assign to your virtual machine. (In this exercise, I selected 32 MB, but you can select a higher value if you have a powerful graphics card:



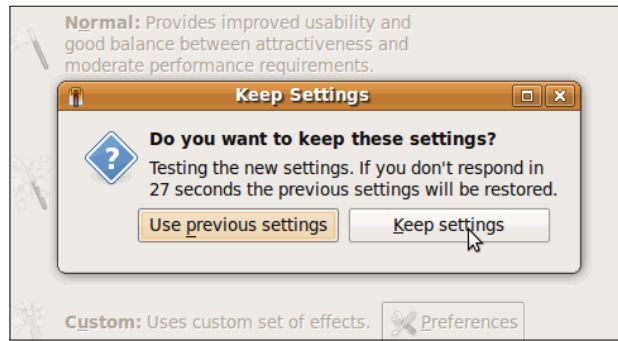
2. Click on **OK** to save your changes and return to the VirtualBox main screen. Now you can start your UbuntuVB virtual machine.
3. Once you're logged in, open a terminal window from the Ubuntu menu bar, type `sudo apt-get install simple-ccsm`, and hit *Enter* to install the Simple Compizconfig Settings Manager.
4. Once the `simple-ccsm` package finishes installing, close the terminal window, and select **System | Preferences | Appearance** from the Ubuntu menu bar:



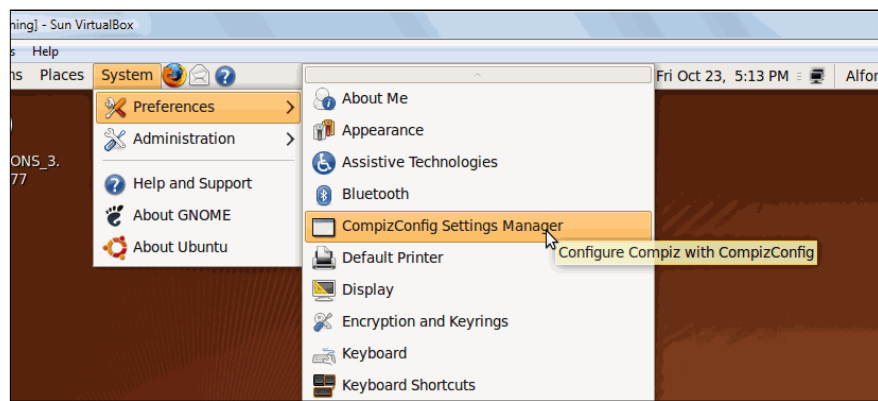
5. On the **Appearance Preferences** dialog, select the **Visual Effects** tab. Then click on the **Custom** option:



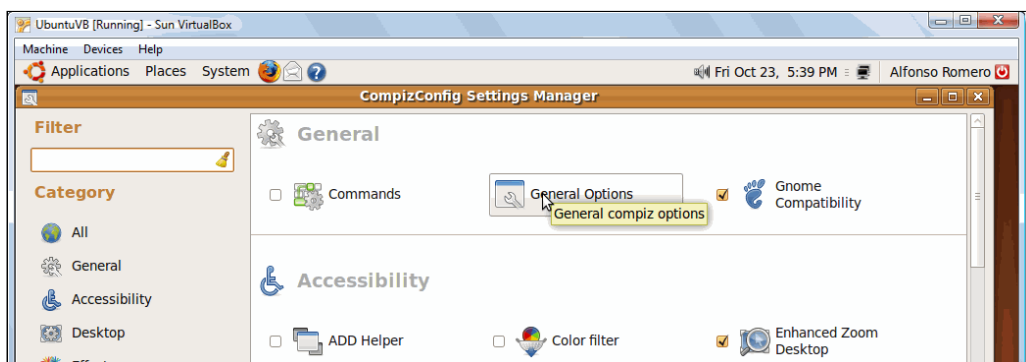
- The **Keep Settings** dialog will show up. Click on the **Keep settings** button to continue:



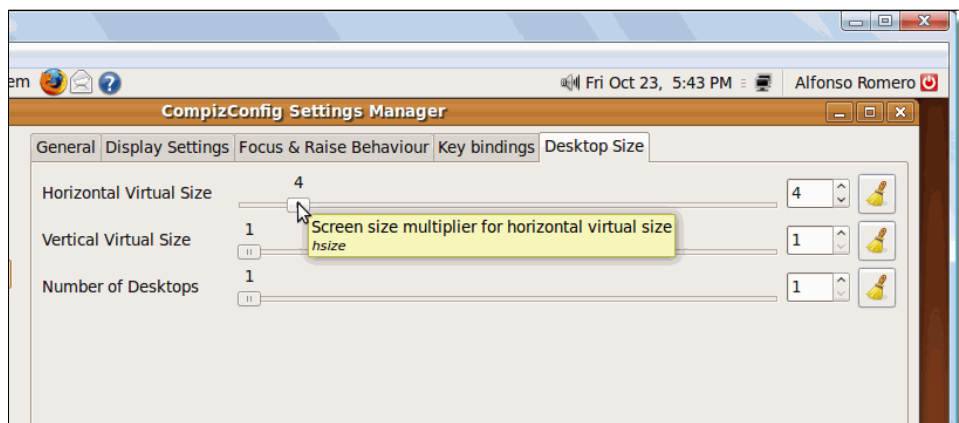
- Now select **System | Preferences | CompizConfig Settings Manager** from the Ubuntu menu bar:



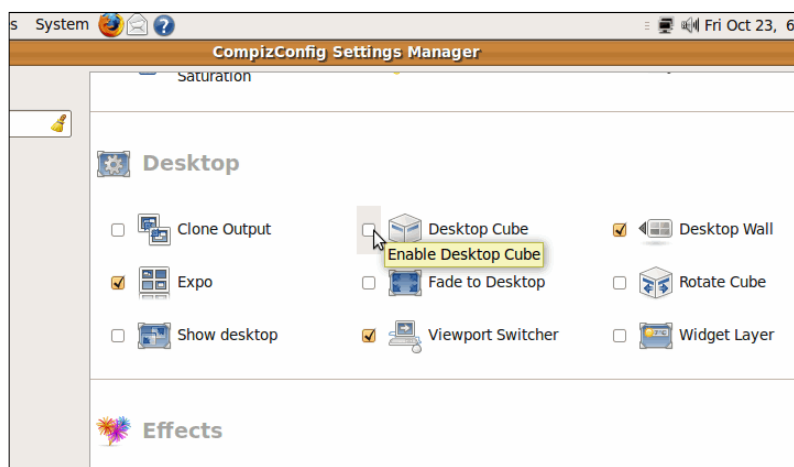
- Click on the **General Options** icon from the **CompizConfig Settings Manager** window:



9. Select the **Desktop Size** tab, and adjust the **Horizontal Virtual Size** slider to 4:



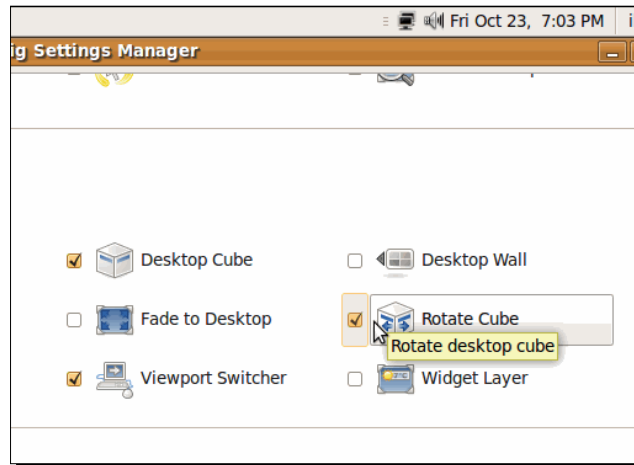
10. Click on the **Back** button, scroll down the **CompizConfig Settings Manager** window until you locate the **Desktop** section, and select the **Desktop Cube** option:



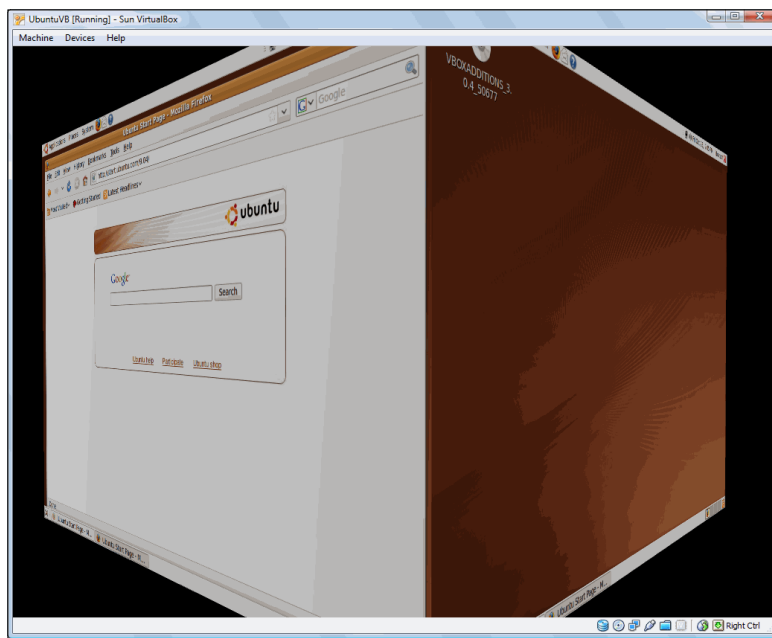
11. The following dialog will show up (click on **Disable Desktop Wall** to continue):



- 12.** Now click on the **Rotate Cube** option to enable it, too:



- 13.** Click on the **Close** button at the bottom-left part of the **CompizConfig Settings Manager** window to save your changes. Now select **Applications | Internet | Mozilla Firefox** to open a web browser window, and then hold **Ctrl + Alt** while pressing the left button in your mouse and scrolling it to the left. The screen will turn into a cube, with the Mozilla Firefox window to the left and the second Ubuntu desktop to the right, as shown below:



- 14.** Leave your Ubuntu virtual machine open for the next exercise.

What just happened?

Nice trick, huh? Compiz has a lot of nice and stunning effects to knock the socks out of your boss! I definitely recommend that you explore the **Effects** section located below the **Desktop** section in the **CompizConfig Settings Manager** and experiment with all the cool effects included. Just be sure to assign enough Video memory to your virtual machine!

As you saw in the exercise, to allow 3D Hardware acceleration in your virtual machines, you simply need to enable an option in the **Settings | Display** panel of your virtual machine (step 1 of the Using Compiz on your Ubuntu VM exercise) and assign an amount of Video memory. Just be sure to never use more than half of your video RAM, for performance reasons.



Remember also that, at the time of this writing, the 3D Acceleration feature in VirtualBox is experimental, so there's no guarantee it will work on every single piece of graphics hardware available.

If we begin with VirtualBox version 3.1, we can see also that there's a 2D Acceleration feature available for Windows guests only, and it's also experimental. You'll find that setting below the 3D Acceleration setting in the Display category of your virtual machine Settings page, as shown in the screenshot from step 1 of the Using Compiz on your Ubuntu VM exercise.

Have a go hero – using 2D and 3D Hardware Acceleration on a Windows VM

Ok, you saw how to enable 3D Hardware Acceleration on an Ubuntu Linux virtual machine. Now's your turn to experiment with a Windows virtual machine and your favorite games such as Age of Mythology, Rise of Nations, or any other games that require 3D Hardware Acceleration features to work. Just be sure to follow the steps indicated in the Installing Guest Additions for Windows exercise.

Oh, and don't forget to try out the 2D Acceleration feature, too! You can try to play some videos from YouTube, for example, and see if your Windows virtual machine can handle the work load.

Summary

In this chapter, I showed you how to take real advantage of your virtual machines by means of the VirtualBox Guest Additions.

Specifically, we covered:

- ◆ How to install the Guest Additions on Windows, Linux, and Solaris guests
- ◆ How to alternate between the fullscreen and windowed modes in your virtual machines
- ◆ How to share folders between your host PC and your virtual machines
- ◆ How to use the seamless windows feature to run applications from your virtual machines side by side with applications from your host PC
- ◆ How to activate hardware 3D acceleration in your virtual machines
- ◆ How to use snapshots to revert your virtual machines to previous states so you can recover from application installation failures or other misfortunes

Now that we've covered how to use Guest Additions to maximize your virtual machine's potential, in the following chapter we'll see everything you need to know about virtual storage in VirtualBox...

5

Storing Data in VirtualBox

By now you've probably figured out that everything on VirtualBox has to include the 'virtual' word, right? And storage media isn't the exception, no siree! In this chapter, I'm going to teach you some useful information about using virtual disk images and three of the most popular formats used with virtualization software like VirtualBox.

In this chapter you shall:

- ◆ Learn how VirtualBox uses 'virtual' hard disks so virtual machines can see them as if they were real
- ◆ Learn how VirtualBox uses the VDI format for all the new virtual machines created and how it can read the VMDK format used by other virtualization products (like VMware) and the VHD format used by Microsoft
- ◆ Learn how to clone hard disks using the Virtual Media Manager
- ◆ See the difference between using fixed and dynamically expanding hard drive images when creating a virtual machine
- ◆ Learn what hard disk controller to choose when creating a virtual machine, depending on the guest operating system for your virtual machine and the number of drives you want to use

Using Virtual Disks in VirtualBox

The hard disk is one of the most important components of any computer equipment. And virtual machines are no exceptions to the rule. To represent hard disks, VirtualBox uses special files called **disk image files**. There are three popular formats used by virtualization software nowadays: **VDI**, **VMDK**, and **VHD**. The following table lists each one of them, along with a brief definition and the virtualization software that uses each:

Virtual Disk Format	Definition	Virtualization Software
VDI	Virtual Disk Image	VirtualBox
VMDK	Virtual Machine Disk	VMware
VHD	Virtual Hard Disk	Microsoft Virtual PC

One of the cool things about VirtualBox is that you can use any of these virtual disk formats on your virtual machines. Sure, you can create VDI images with only the VirtualBox GUI, but with the VBoxManage command-line interface, you can just open a Command Prompt or terminal window and issue the `VBoxManage createhd` command, which lets you specify if you want a VDI, VMDK, or VHD disk.

And to top it off, there are a lot of pre-installed software applications and operating systems that you can download for free as VDI, VMDK, or VHD images, as we'll see in Chapter 7 when we talk about virtual appliances. With VirtualBox available, you'll be able to access almost every virtual disk image available! In the meanwhile, let's see how to use additional hard drives in your virtual machines...



Starting with VirtualBox 3.1, you can also use HDD images from the Parallels virtualization software, version 2. However, you cannot create these types of images with VirtualBox yet.

Pop quiz – using virtual disks in VirtualBox

1. A disk image file is:
 - a. The physical drive connected to your real computer.
 - b. Another name for a virtual machine.
 - c. The way VirtualBox represents hard disk drives.
2. Can you access virtual machines created with VMware in virtualbox?
 - a. Yes, you just need to add the VMDK disk image instead of creating a new virtual hard drive.

- b. No, because VirtualBox doesn't support the VMDK format.
 - c. No, because VirtualBox only supports the VHD format.
3. Since you introduced your boss to VirtualBox, he doesn't want to go back to VMware, but he still needs several virtual machines in the VMDK format. Can he use those virtual machines in VirtualBox?
- a. Yes.
 - b. No.
 - c. Gee, I don't know!

Using an additional VDI hard drive

As VDI is the default format for virtual hard drives in VirtualBox, let me show you how to add a brand new drive to a virtual machine. This is the same as installing a physical hard drive in your host PC, with the advantage that you don't need to go out and buy a new "virtual" disk. Oh, and you can create a virtual disk of any size, as long as you don't exceed the total size of your physical hard drive.

In the following exercise, you'll practice with the `UbuntuVB` virtual machine we created in previous chapters, but the process is nearly the same on any other virtual machine. The only difference is the way each operating system handles the process of adding and formatting a new hard drive.

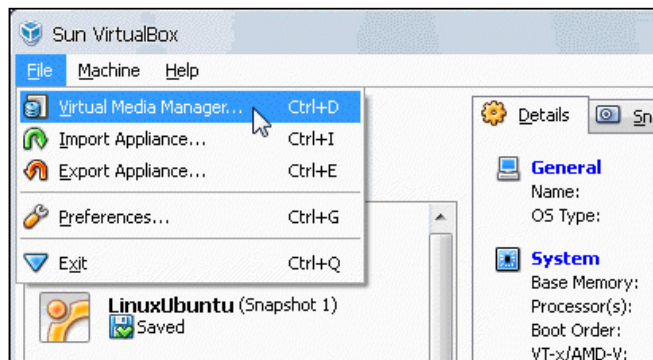
Time for action – adding a secondary virtual drive to your VM

In this exercise, I'll show you how to add another virtual drive to your VM, using the Virtual Media Manager included in VirtualBox and the VDI format.

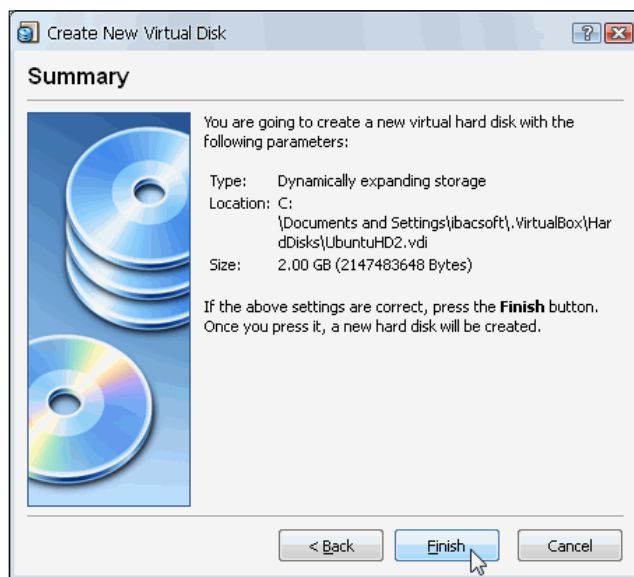


Remember to use the `UbuntuVB` virtual machine we created in previous chapters because we'll refer to it on several exercises in this chapter.

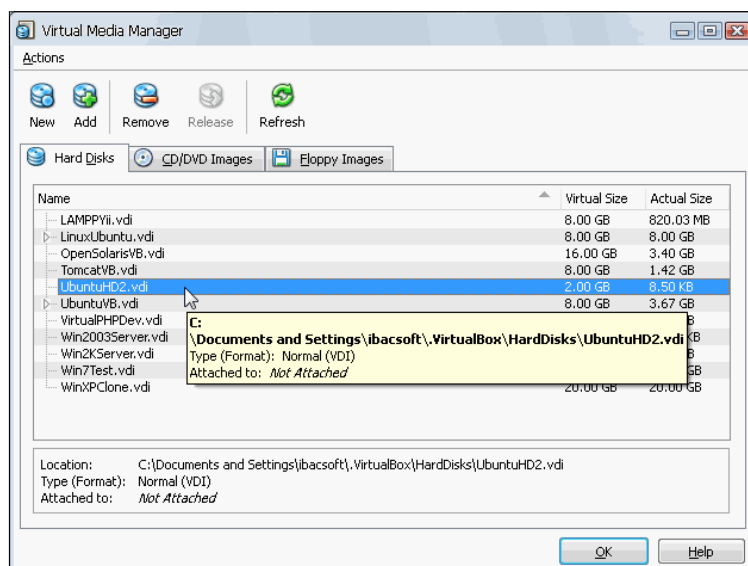
1. Open VirtualBox, and select **File | Virtual Media Manager...** from the main menu:



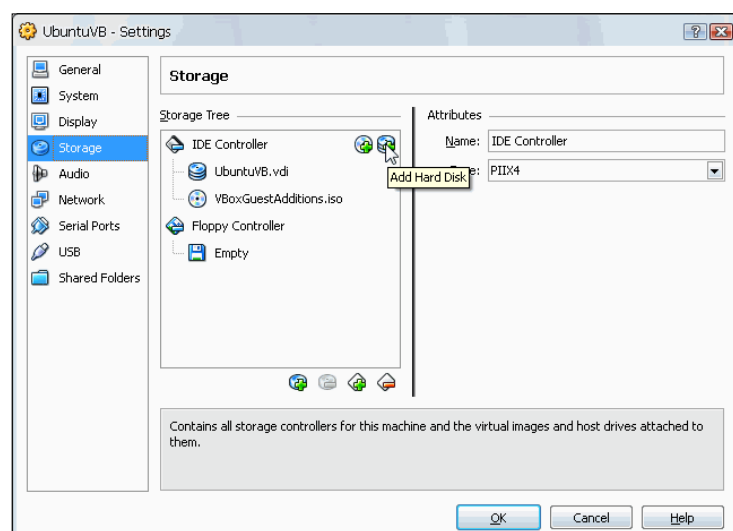
2. The **Virtual Media Manager** window will appear, showing a list of all the hard disks available in your VirtualBox installation. Click on the **New** button to create a new virtual hard disk.
3. Leave the **Dynamically expanding storage** option selected on the **Hard Disk Storage Type** screen, and click on **Next** to continue.
4. Type `UbuntuHD2.vdi` in the **Location** field, leave the default value of `2.00 GB` in the **Size** section, and click on **Next** to continue.
5. The **Summary** dialog will appear next. Verify that your settings are correct, and click on **Finish** to continue:



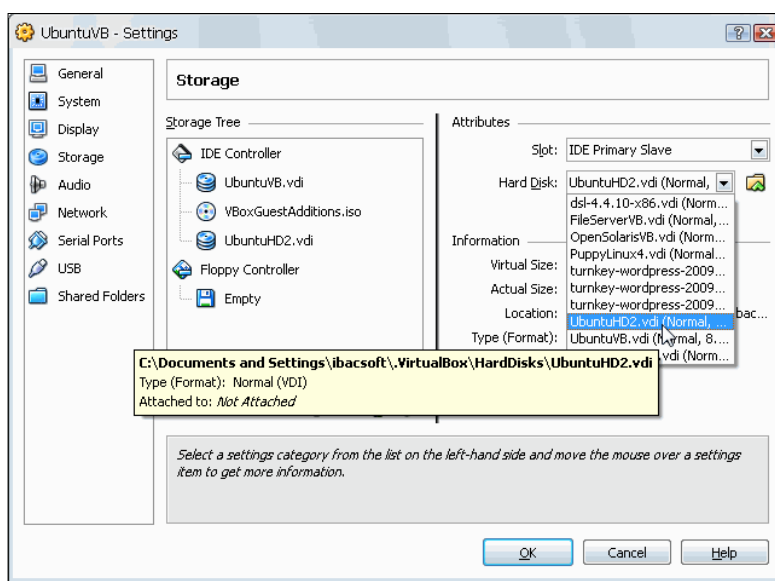
6. Now your UbuntuHD2.vdi virtual disk will show up in the **Virtual Media Manager Hard Disks** tab:



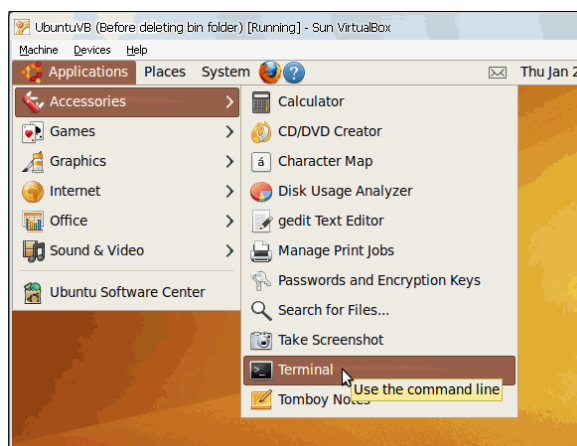
7. Click on **OK** to close the Virtual Media Manager and return to VirtualBox.
8. Select your UbuntuVB virtual machine and click on the **Settings** button.
9. The **UbuntuVB – Settings** dialog will show up. Select the **Storage** category, and click on the **Add Hard Disk** button to attach a new hard disk to your virtual machine:



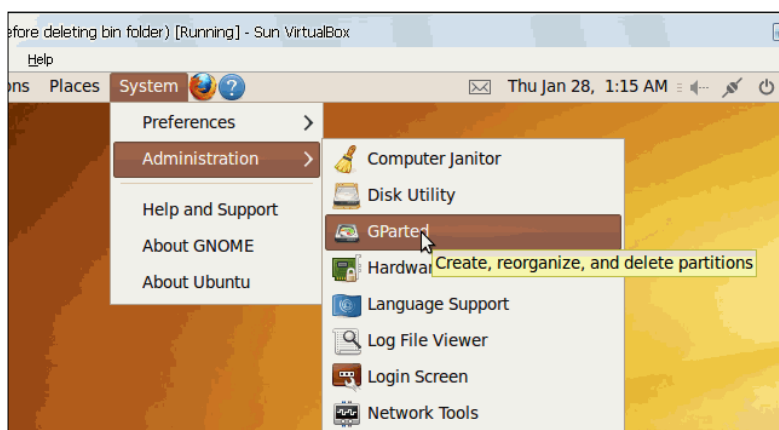
10. A new slot will appear below the **VBoxGuestAdditions.iso** slot under the **IDE Controller**, inside the **Storage Tree**. Click on the new slot to select it; the **Attributes** panel will change to show the **Slot** and **Hard Disk** fields. Click on the drop-down list from the **Hard Disk** field, and select the **UbuntuHD2.vdi** hard disk image:



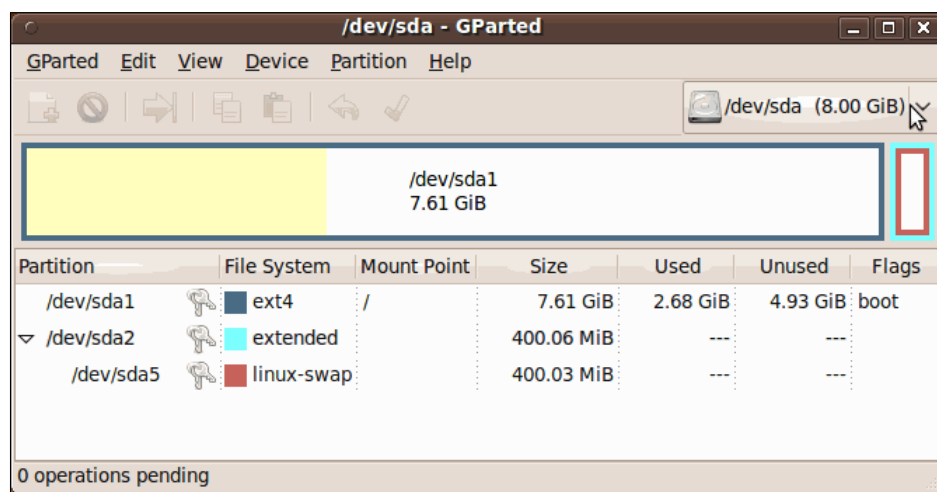
11. Click on **OK** to close the **UbuntuVB – Settings** dialog, and then click on the **Start** button to start your UbuntuVB virtual machine.
12. Login to your Ubuntu virtual machine. Then select **Applications | Accessories | Terminal** from the Ubuntu main menu to open a terminal window:



13. Type `sudo apt-get install gparted`, and press *Enter* on the Terminal window to install the GNOME Partition Editor. Wait until the installation process finishes, and type `exit` to close the Terminal window.
14. Now select **System | Administration | GParted** to open the partition editor:



15. Type your `sudo` password when Ubuntu asks for it. The **GParted** window will appear next, showing information about your first hard drive (`/dev/sda` in this example):

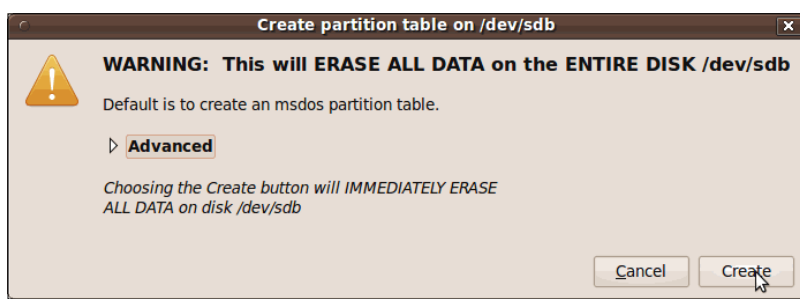


16. Select **GParted | Devices | /dev/sdb** from the **GParted** main menu to see the hard drive you added to your virtual machine in steps 9 through 11 of this exercise.

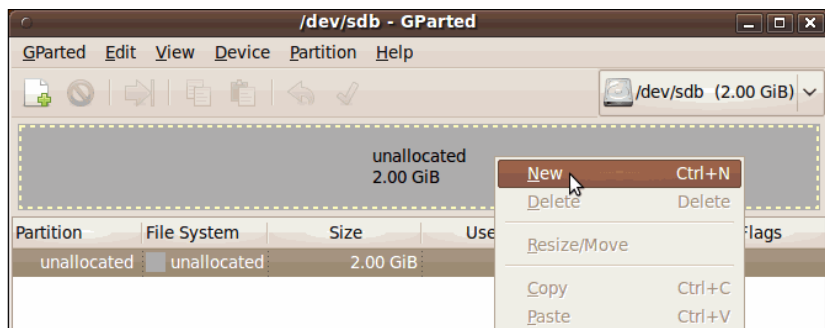


You can also select the hard drive from the drop down box located in the upper-right corner of the GParted window.

17. Your virtual hard drive will show up as a grey rectangle with the **unallocated 2.00 GiB** caption inside. Select **Device | Create Partition Table...** from the GParted main menu.
18. The **Create partition table on /dev/sdb** dialog will show up to warn you about erasing all your hard drive's data:

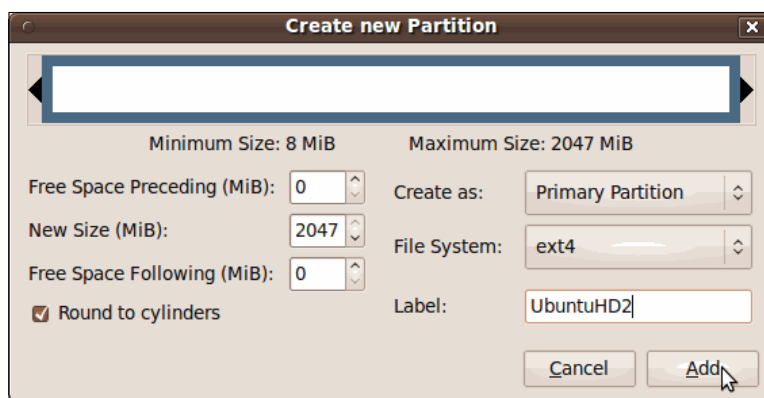


19. Click on **Create** to continue, wait until GParted creates the partition table for your second hard drive. Then right click on the grey rectangle, and select **New** from the pop-up menu:



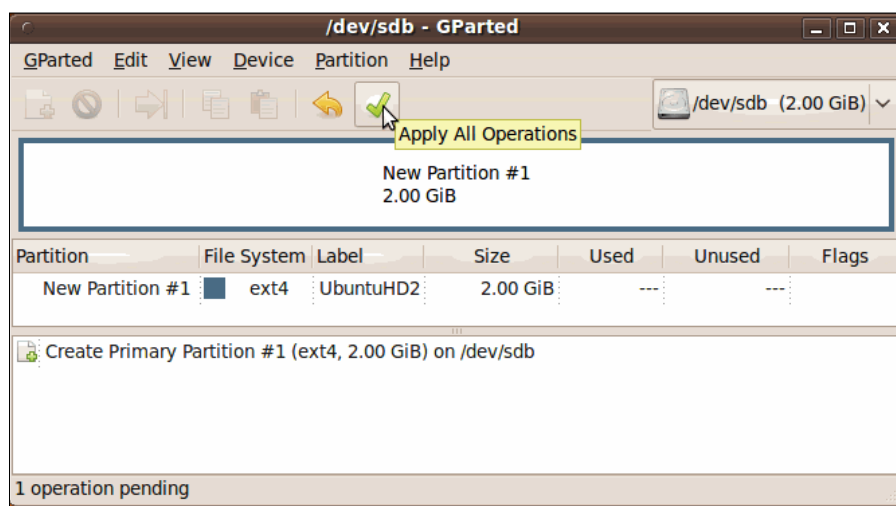
You can also click on the **New** button from the GParted toolbar.

- 20.** The **Create New Partition** window will appear next. Type `UbuntuHD2` in the **Label** field, select `ext4` in the **File System** list box, and leave the default values on the other fields; then click on **Add** to continue:

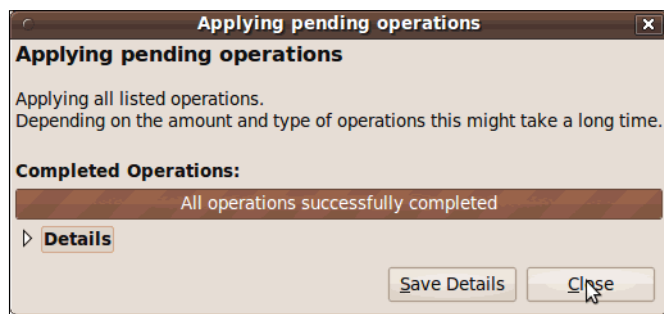


The default filesystem whenever you create a new partition is `ext2`; I selected the `ext4` filesystem for this exercise because it's slightly faster and has more features, and it would be good for you to try it out on a virtual machine.

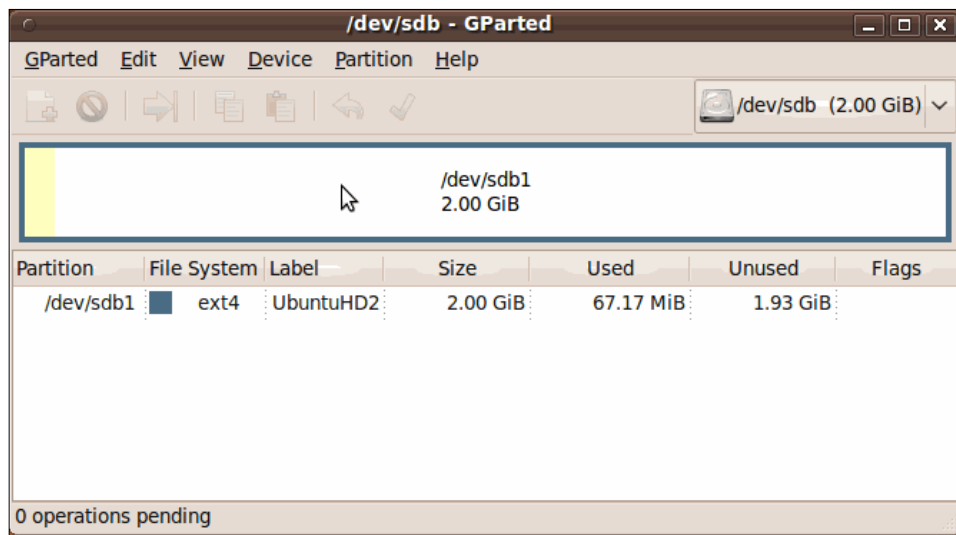
- 21.** You'll return to the **GParted** main screen where you'll see your new partition details. Click on **Apply** to create the partition on your hard drive:



22. The **Apply operations to device** dialog will pop up, asking if you really want to apply the changes to your hard drive. Click on **Apply** to continue.
23. The **Applying pending operations** dialog will appear to show you that GParted is applying the changes to your hard drive. When it is finished, the following screen will show up:



24. Click on **Close** to continue. GParted will return you to the main screen, where you'll see the newly created partition on your hard drive:



25. Close the GParted application, open a terminal window (**Applications | Accessories | Terminal**), and type the following lines (remember to press *Enter* after each line):

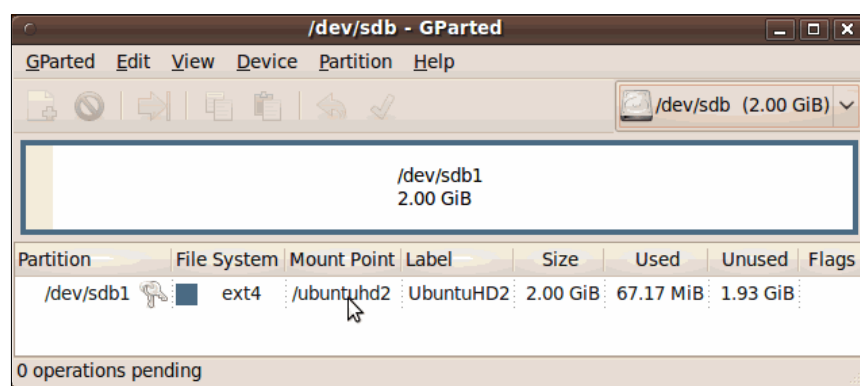
```
sudo mkdir /ubuntuhd2
sudo mount -t ext4 /dev/sdb1 /ubuntuhd2
```



If you shutdown your virtual machine and then start it again, you'll notice the `/ubuntuhd2` partition you mounted has disappeared, and you'll have to execute the mount command again.

If you want more information about mounting persistent partitions, you can visit <https://help.ubuntu.com/community/AutomaticallyMountPartitions>

- 26.** Now you can use your new additional hard drive on your Ubuntu virtual machine! If you open GParted again, you'll notice that there's an extra column named **Mount Point** on the partition info area:



- 27.** You can close the GParted application and your Ubuntu virtual machine.

What just happened?

Ok, I must admit that was a lengthy exercise, but I'm sure you will find it very useful when dealing with several virtual machines at the same time. You can create a secondary virtual hard drive on your Ubuntu VM, for example, and then use it in your Windows XP virtual machine to share information between both virtual machines as if you were using a real external hard drive!

You only need to partition and format your virtual hard drive the first time you use it. If you later decide to attach that virtual hard drive to another virtual machine, it will be detected automatically for you. Just be sure your virtual machine's operating system supports the filesystem in your virtual disk! For example, if you create a virtual disk in Ubuntu Linux and follow the above exercise, the filesystem will be `ext4`, and your Windows virtual machines won't be able to read it natively, but if you create a virtual hard disk on a Windows XP machine and format it with the NTFS or FAT filesystem, both Windows and Linux virtual machines will be able to use it.



Although you can't read ext2/ext3/ext4 hard drives natively on a Windows system, you can use the ext2fsd driver (<http://www.ext2fsd.com>) or the Freeware Linux Reader for Windows (<http://www.diskinternals.com/linux-reader/>) to read ext2 and ext3 hard drives.

Basically, what VirtualBox does is it creates a file on your physical hard drive and makes your virtual machine's operating system think it's a real working hard disk. That way, if you later move your virtual machine to another host, it won't matter if the new host has a different physical hard drive because all that your virtual machine will work with is a VDI image file. Cool, right?

And where does VirtualBox save its virtual disks? Well, in step 5 of the previous exercise, you typed a name for your new hard drive and selected its size in megabytes. If you open the Virtual Media Manager on VirtualBox and create a new hard disk, when you get to the Virtual Disk Location and Size screen, you can click on the folder icon (📁) to the right of the **Location** field.

The **Select a file for the new hard disk image file** dialog will open up, where you can choose a specific location for your new virtual disk.

The default folder used by VirtualBox to save hard disk images is called `HardDisks`, and it's located under the `.VirtualBox` folder that VirtualBox creates on your user home folder. But if you want to save your virtual disk in another location, go ahead! You can even save it on another physical hard drive, in case your host PC has two or more of them.

So as you can see, the only limit for virtual disk images is your physical drive's available space at the time of creating them. VirtualBox takes care of the technical details for you!

Have a go hero – creating an additional VDI hard disk on a Windows VM

And now's your turn to make the coffee! Create a VDI hard disk named `WindowsHD2.vdi`, and attach it to your Windows XP virtual machine, using the previous exercise as a guide. On Windows Vista/XP you need to use the Disk Management tool (**Start | Control Panel | Administrative Tools | Computer Management**) to partition and format your new hard disk.

Pop quiz – creating additional virtual disk images

1. If you want to add another hard drive to your Windows XP virtual machine, you:
 - a. Go to Amazon and order a new 500 GB disk.
 - b. Split the virtual hard drive your virtual machine already possesses into two virtual hard drives.
 - c. Use the Virtual Media Manager to create a new virtual VDI image, and attach it to your virtual machine. Then you can create a new partition and format your new virtual drive from within your Windows virtual machine.
2. Can you use an additional VDI disk image to save some information on your Ubuntu virtual machine, and then copy that VDI disk image to your boss's computer so he can access the information you saved?
 - a. Yes, because you can attach a VDI disk image on any virtual machine.
 - b. No, your boss will need to use your computer to see that information.
 - c. Hmmm... Guess I'll pass on this one!

Using a VHD hard drive

One of the biggest advantages of virtualization and VirtualBox is that you can use the virtual disk formats from Microsoft Virtual PC and VMware Workstation, and now it's time for you and me to test-drive this feature using a VHD image created with Virtual PC, the virtualization software owned by Microsoft. You can follow the same procedure if you want to use VMDK images from VMware.



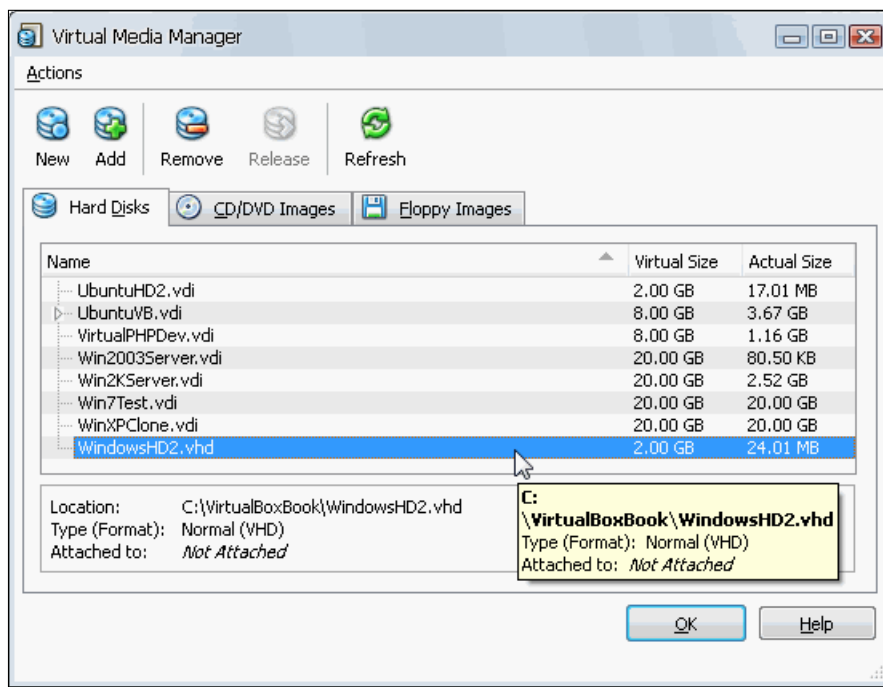
For the following exercise, you'll need to download the `WindowsHD2.vhd` file from the book's website: http://www.packtpub.com/files/code/9140_Code.zip.

Time for action – adding a VHD virtual drive to your VM

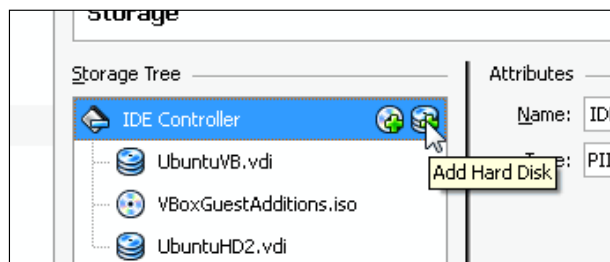
In this exercise, I'll show you how to add a VHD virtual drive, created on the Microsoft Virtual PC virtualization software, to your VM.

1. Open VirtualBox, and select **File | Virtual Media Manager...** from the main menu. Then click on the **Add** button, or press the *Insert* key to add a virtual hard disk.
2. The **Select a hard disk image file** dialog will open up next. Navigate to the directory where you downloaded the `WindowsHD2.vhd` image file, and double-click on it.

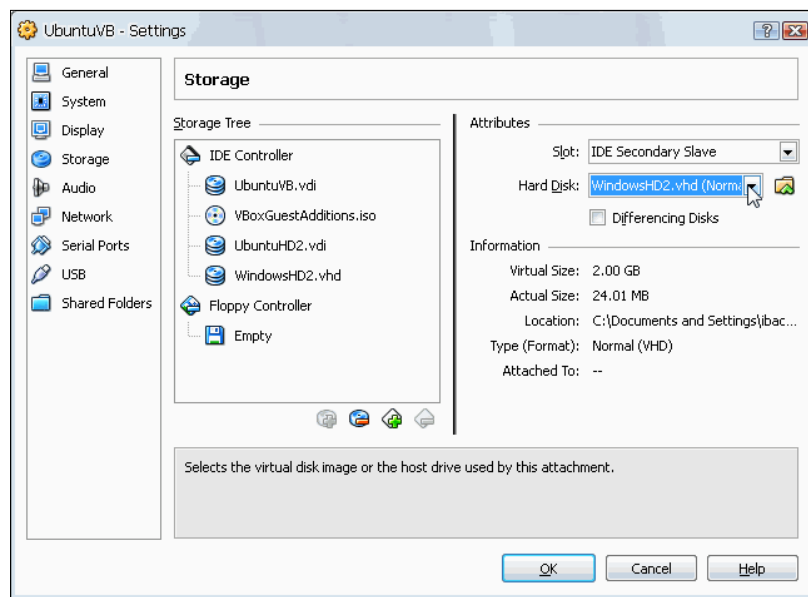
3. Your new hard drive will appear on the **Virtual Drive Manager Hard Disks** tab:



4. Click on **OK** to return to the VirtualBox main screen. Then select your UbuntuVB virtual machine, and click on the **Settings** button to open the **UbuntuVB – Settings** dialog.
5. Select the **Storage** category, and click on the **Add Hard Disk** button located at the right end of the **IDE Controller** to add a slot for another IDE virtual disk:



6. A new IDE virtual disk slot will appear under the UbuntuHD2.vdi slot. Select the new slot, and then use the **Hard disk** list box under the **Attributes** panel to select the **WindowsHD2.vhd** hard disk image:

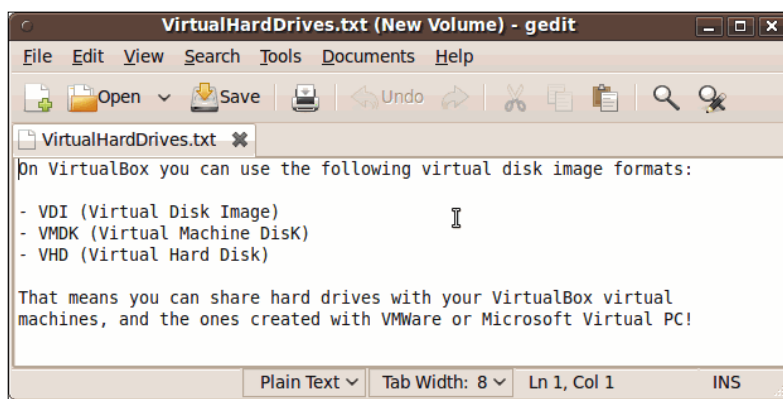


7. Click on **OK** to close the **UbuntuVB – Settings** dialog and return to the VirtualBox main screen. Now start your UbuntuVB virtual machine, and log in.
8. Once logged in, select **Places | New Volume** from the Ubuntu main menu to mount your new VHD disk image.
9. The **Authenticate** dialog will show up next. Type your root password, and click on **Authenticate** to continue:



10. The **New Volume – File Browser** window will appear, showing your VHD hard drive's contents.

11. Double-click on the `VirtualHardDrives.txt` file to open it up and see its contents. The **Do you want to run "VirtualHardDrives.txt", or display its contents?** dialog will appear next. Click on **Display** to continue.
12. The **Gedit** editor will start up to show the `VirtualHardDrives.txt` file's contents:



13. You can close the text editor and your UbuntuVB virtual machine now.

What just happened?

As you saw in the previous exercise, VirtualBox doesn't care if the hard disk image format is VDI, VHD, or VMDK. As long as it can run on the Microsoft Virtual PC, VMware, or Parallels virtualization products, it will also run on VirtualBox! It's like using real hard drives without the hassle of having to open your physical machine and connect them!

Have a go hero – using a virtual disk image on several virtual machines

Let's try something interesting. Remove the `WindowsHD2.vdi` disk image from your UbuntuVB virtual machine; to do that, you need to enter the **UbuntuVB – Settings** window and select the **Storage** category. Then select the `WindowHD2.vhd` disk image, and click on the **Remove Attachment (-)** button under the **Storage Tree** panel. The `WindowsHD2.vdi` disk will disappear from the **Storage Tree** panel. Close your **UbuntuVB – Settings** window.

Now attach the `WindowsHD2.vhd` disk image to a Windows virtual machine, create a WordPad document, copy some content from the VirtualBox webpage, and then save the document on the `WindowsHD2.vdi` disk. Close your Windows virtual machine, remove the `WindowsHD2.vdi` disk from it, and reattach it to your UbuntuVB virtual machine to see if you can open and read the WordPad document you created.

You can also try to write something from your Ubuntu virtual machine and then see if you can read it from your Windows virtual machine, using OpenOffice Writer and MS Word, for example.

Have a go hero – using a virtual disk image on two virtual machines at the same time

Now you're going to try something similar to the previous *Have a go hero* section, but this time leave the `WindowsHD2.vhd` disk image attached to both virtual machines and see what happens. Start your Ubuntu virtual machine, create a file, and save it on the `WindowsHD2.vhd` disk image; then close your Ubuntu virtual machine, and start your Windows virtual machine to see if you can read the file you created before on your `WindowsHD2.vdi` disk.

If you're feeling intrepid, try starting both virtual machines, and see if you can access the `WindowsHD2.vdi` disk at the same time! And don't worry, this is not going to make your computer blow up or shatter into pieces!

Creating multiple virtual machines by cloning

You spent half a day installing your Ubuntu virtual machine, updating and installing all the required software... Finally life starts to make sense...

Suddenly, your boss bursts into your office and starts screaming: "I need you to test two content management systems to see which one we are going to use on our company's web!"

So now you have to test two completely independent applications, and you have only one virtual machine to test them on! You could spend the rest of the day installing, updating, and configuring another Ubuntu virtual machine, or you could just copy the hard drive image from the Ubuntu virtual machine you've just finished configuring, create another Ubuntu virtual machine, and attach that hard drive image so you can avoid the hassle of installing Ubuntu from scratch!

But there's a little problem: VirtualBox assigns a **unique identity number (UUID)** to each hard drive image, and if you try to use two identical disk images on the same host PC, VirtualBox simply will refuse to work, so we need to use a special procedure called cloning.

Want to know how it works? Keep reading...

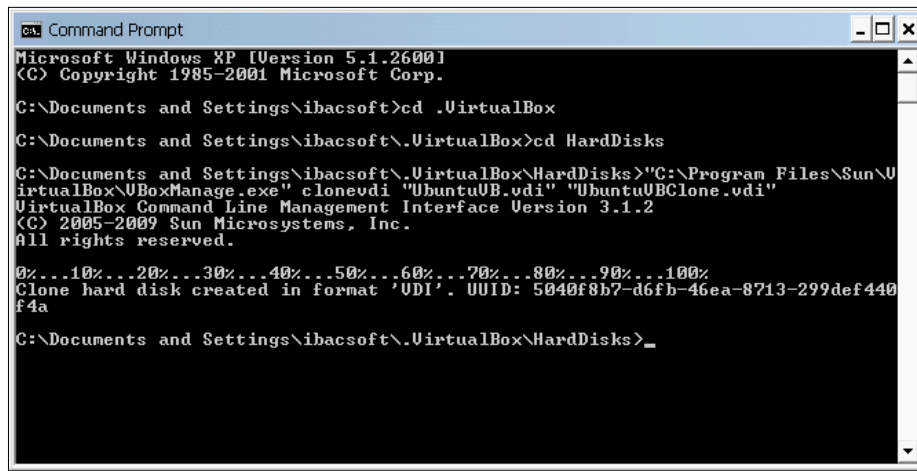
Time for action – cloning an Ubuntu Linux hard disk image

In this exercise, I'll show you how to clone a VirtualBox hard disk using Ubuntu Linux as the guest operating system and Windows XP as the host.

1. Open a **Command Prompt** window (**Start | All Programs | Accessories | Command Prompt**), and type the following lines (don't forget to press *Enter* after each line):

```
cd .VirtualBox
cd HardDisks
"C:\Program Files\Sun\VirtualBox\VBoxManage.exe" clonevdi
    "UbuntuVB.vdi" "UbuntuVBClone.vdi"
```

2. Now just wait until VirtualBox finishes cloning your hard drive. The following screenshot shows all the disk cloning process:



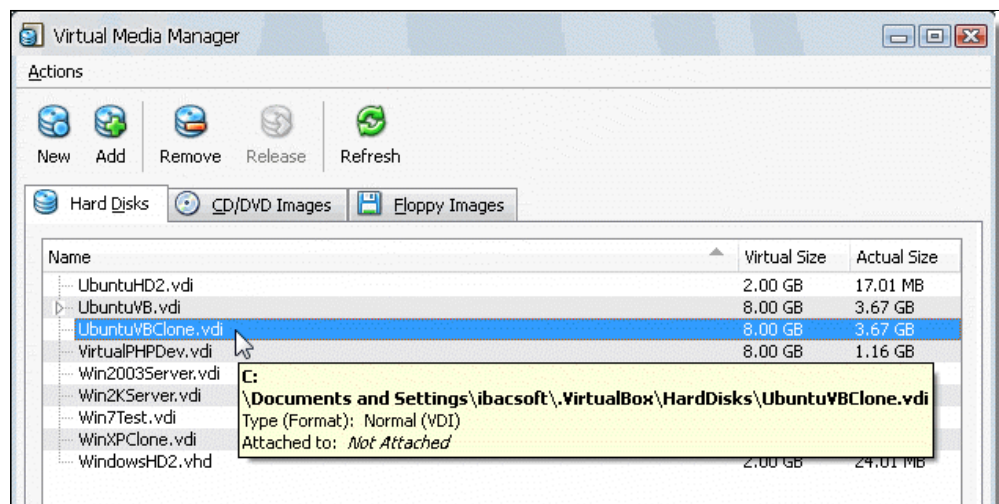
```
Microsoft Windows XP [Version 5.1.2600]
(C) Copyright 1985-2001 Microsoft Corp.

C:\Documents and Settings\ibacsoft>cd .VirtualBox
C:\Documents and Settings\ibacsoft\VirtualBox>cd HardDisks
C:\Documents and Settings\ibacsoft\VirtualBox\HardDisks>"C:\Program Files\Sun\VirtualBox\VBoxManage.exe" clonevdi "UbuntuVB.vdi" "UbuntuVBClone.vdi"
VirtualBox Command Line Management Interface Version 3.1.2
(C) 2005-2009 Sun Microsystems, Inc.
All rights reserved.

0%...10%...20%...30%...40%...50%...60%...70%...80%...90%...100%
Clone hard disk created in format 'VDI'. UUID: 5040f8b7-d6fb-46ea-8713-299def440f4a

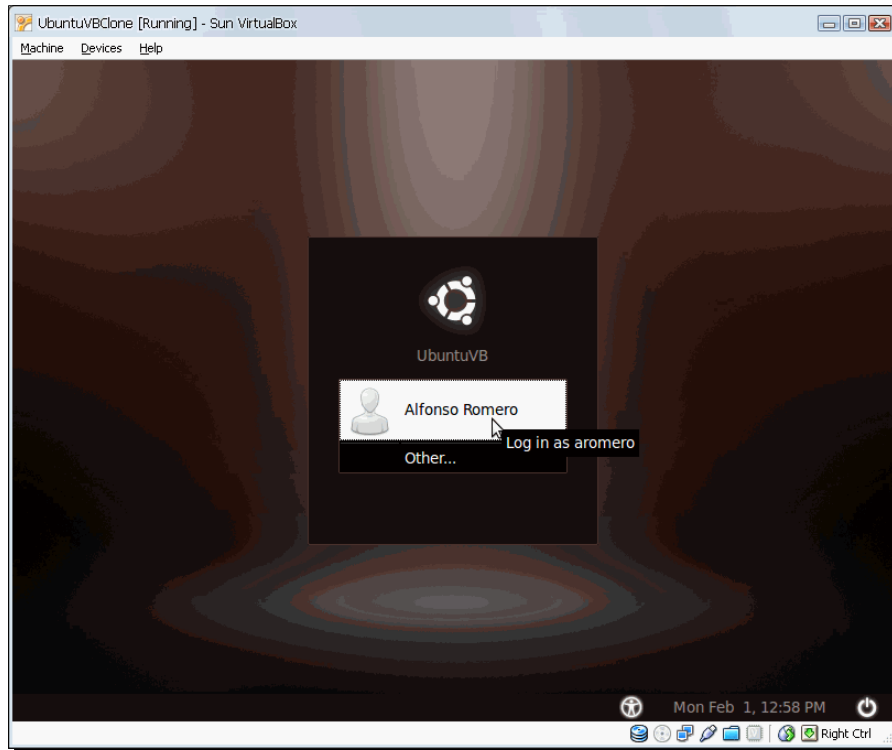
C:\Documents and Settings\ibacsoft\VirtualBox\HardDisks>_
```

3. Close the Command Prompt window. Then open VirtualBox, and select **File | Virtual Media Manager** to open the Virtual Media Manager.
4. Click on the **Add** button, or press the *Insert* key to open the **Select a hard disk image file** dialog, select the `UbuntuVBClone.vdi` disk image you created before, and click on **Open** to continue.
5. The `UbuntuVBClone.vdi` disk image file will be added to the **Virtual Media Manager**:



6. Click on **OK** to close the **Virtual Media Manager** and return to the VirtualBox main screen. Then click on **New** to create a new virtual machine.
7. The **Create Virtual Machine** wizard will show up. Click on **Next** to continue.
8. Type `UbuntuVBClone` in the **Name** field of the **VM Name and OS Type** screen. Then select **Linux** as the **Operating System** and **Ubuntu** as the **Version** in that same screen. Click on **Next** to continue.
9. Leave the default value (384 MB) as the **Base Memory Size** in the **Memory** screen, and click on **Next** to continue.
10. Select the **Use the existing hard disk** option in the **Virtual Hard Disk** screen, and then select the `UbuntuVBClone.vdi` disk image from the pop-up list box. Click on **Next** to continue.
11. The **Summary** screen will appear, showing all the options you selected. Click on **Finish** to create the virtual machine and return to VirtualBox.
12. Now click on the **Start** button to start your new virtual machine.

- 13.** The Ubuntu login page will appear to confirm that your Ubuntu virtual machine was cloned flawlessly:



- 14.** You can login and verify that your virtual machine is working as well as the original UbuntuVB machine.

What just happened?

So now you know how to clone virtual disks! And remember, you only need to do this when using two or more identical copies of a virtual hard drive on the same VirtualBox installation because of the UUID value VirtualBox assigns to each virtual hard disk image on the same host PC. If you copy a virtual disk and then assign it to a virtual machine on another host PC, you don't need to clone it; just copy it like a regular file.



One of the best things about cloning a hard disk is that you don't have to install the Guest Additions on your new cloned virtual machine!

In step 1 of the previous exercise, we used the `VBoxManage clonevdi` command to clone your `UbuntuVB.vdi` disk image. When using a Windows host, you need to add the full path to the `VBoxManage` command; another option is to add the VirtualBox installation folder—the default being `"C:\Program Files\Sun\VirtualBox\"`—to your system path.

On Linux systems, you don't have to specify the VirtualBox path when using the `VBoxManage clonevdi` command.

Have a go hero – cloning and registering your virtual disk images at the same time

You already saw how to clone a virtual hard disk image with the `VBoxManage clonehd` command. This command has an optional parameter that you can use to register your cloned disk image at the same time you create it: `--remember`.

This means you won't have to open the Virtual File Manager and add the cloned virtual disk image because VirtualBox will do it for you. Now go and clone your WindowsXP disk image, but this time use `"C:\Program Files\Sun\VirtualBox\VBoxManage.exe"` `clonevdi "WindowsXP.vdi" "WindowsXPClone.vdi" ---remember`. Then open the Virtual File Manager, and your new cloned image will be already registered!

Have a go hero – cloning and converting virtual disk images to other formats

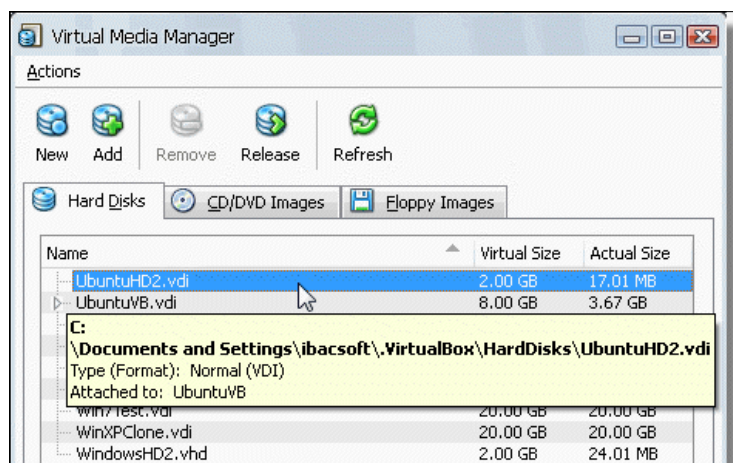
You already saw how to use the `VBoxManage clonehd` command to create cloned VDI disk images; this time, try converting these cloned images to the VMDK or VHD formats using the `--format VMDK` and `--format VHD` parameters. For example, to clone your `WindowsXP.vdi` image and convert the cloned image to the VHD format, use `"C:\Program Files\Sun\VirtualBox\VBoxManage.exe"` `clonevdi "WindowsXP.vdi" "WindowsXPClone.vhd" ---format VHD`.

Pop quiz – virtual storage

1. What's the difference between cloning a virtual disk and copying it into the same virtual machine?
 - a. You won't be able to use two identical virtual disks on the same virtual machine.
 - b. There's no difference.
 - c. You can't clone a virtual disk if you're using a Windows XP virtual machine.
2. You need to test 3 different database management systems—SQL Server, Oracle, and MySQL—and see which one you can recommend for your company's data center. What would be the best thing to do?
 - a. Buy three identical PCs, with Windows 2003 server installed, to test each management system and determine which one suits your needs best.
 - b. Create a clean Windows 2003 server virtual machine, and then clone it two times to have three identical Windows 2003 server virtual machines to test the database management systems.
 - c. Order pizza for everyone in the office!

Expanding hard disk images on the fly

Up to this point in the chapter, we've been using dynamically expanding virtual disk images. That means their size grows each time the operating system writes information for the first time. For example, if you open the Virtual Disk Manager and look at the `UbuntuHD2.vdi` image we created and used in the first section of this chapter (*Using Virtual Disks in VirtualBox*), you'll notice its real size is very small (17 MB) compared to the size we selected when creating it (2.00 GB):



And the same thing happens with all the other virtual hard drives we've been using in this book. Whenever the operating system of your virtual machine needs to write something to the hard drive, its size expands automatically until it reaches the maximum value you assigned to it when creating your virtual machine.



Be careful when selecting the size of a dynamically expanding virtual hard disk because this will be the **MAXIMUM** size this disk will be able to consume on the host. If, for example, you create an 8 GB dynamically expanding virtual disk and then find out you're going to need more space, you won't be able to expand its size. It's better to select a bigger size than what you currently require; your dynamically expanding disk won't waste space in your host disk anyway.

On the other hand, fixed-size hard disk images are more like a real hard drive: they always have the full size value you assign to them right from the start! One disadvantage is that it takes more time to create them, but on the other hand, they tend to be faster than dynamically expanding images because VirtualBox doesn't need additional computing resources to expand them on the fly.

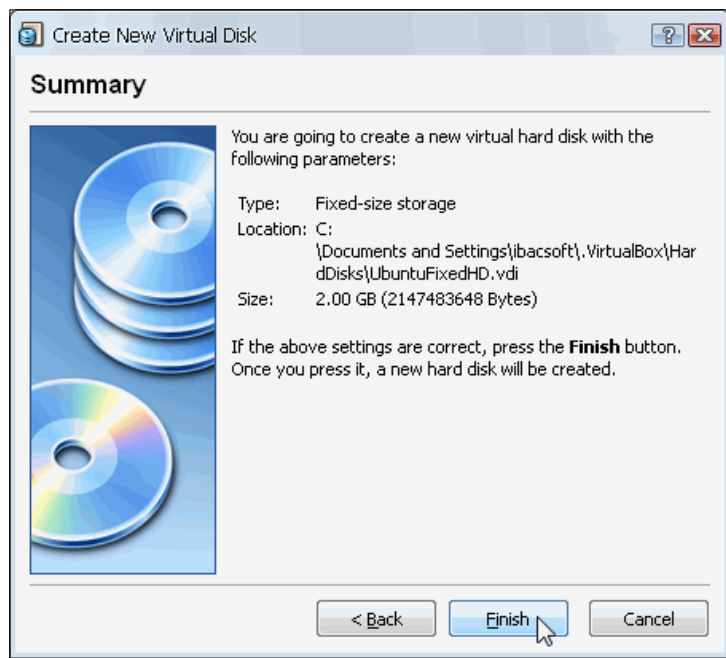
Oh, and since a fixed-size hard disk image doesn't need to increase its size, you don't have to worry about your virtual machine stopping suddenly in case your host hard disk fills up!

Time for action – creating a fixed-size hard drive image

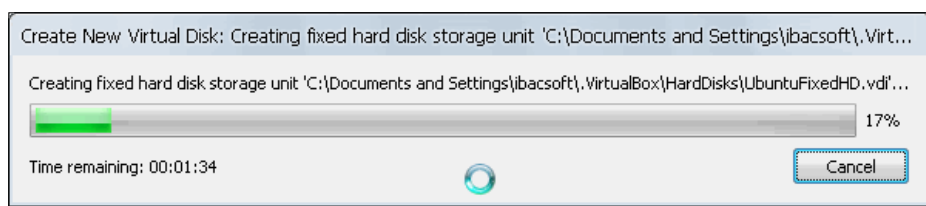
In this exercise, I'll show you the difference between creating a fixed-size hard drive image and a dynamically expanding one.

1. Open VirtualBox, and open the Virtual Media Manager (**File | Virtual Media Manager**). Then click on the **New** button to create a new hard disk image.
2. The **Create New Virtual Disk** wizard will show up next. Click on **Next** to continue, and then select `Fixed-size storage` on **Storage Type** field from the **Hard Disk Storage Type** screen.
3. Click on **Next** to continue. Type `UbuntuFixedHD.vdi` in the **Location** field, and leave the `2.00 GB` default value in **Size**.

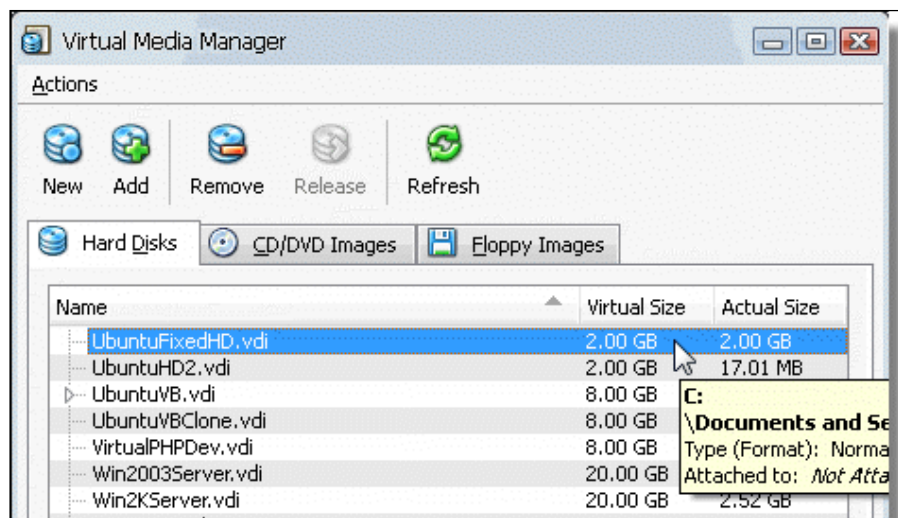
4. Click on **Next** to continue. The **Summary** screen will show up with a brief description of all your choices. Click on **Finish** to start creating your new virtual hard drive:



5. The wizard will start preparing your new virtual hard drive, as shown in the following screenshot:



6. When the wizard finishes creating your new hard drive, it will show up on the **Virtual Media Manager**:



7. Click on **OK** to close the **Virtual Media Manager** and return to the VirtualBox main screen.

What just happened?

Now you know how to create fixed-size images for your virtual machines. Although it seems better to use dynamically expanding images because they take less time to create and, aside from that tiny little detail, everything else seems to work the same way, there are some scenarios where I'd recommend you use fixed-size images. For example, if you need to run a virtual machine as smoothly as possible, maximizing all resource usage, it's better to use fixed-size images because they don't need extra computing power to expand their size. Servers are one area where I personally prefer to use fixed-size images, especially if I plan to use them on production environments.


Choosing your disk controller type: IDE, SATA, or SCSI

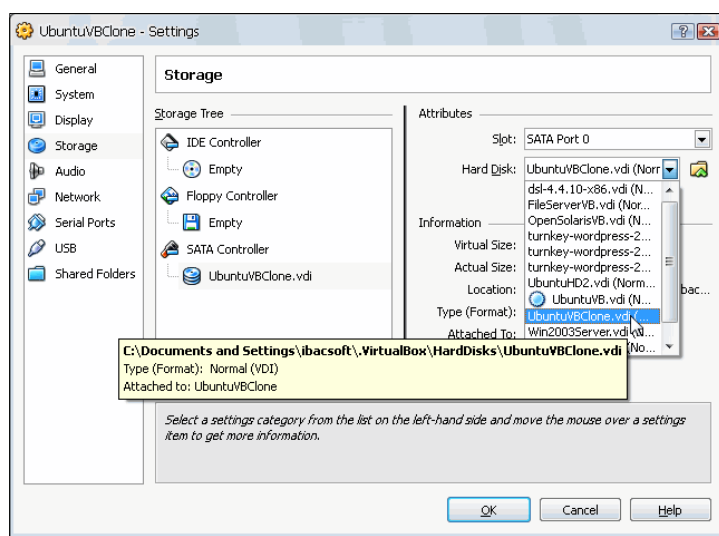
As with real PCs, virtual machines in VirtualBox can emulate the three types of disk controllers available today: IDE, SATA, and SCSI. The default type is IDE whenever you create a new virtual machine. And why is that? Well, the main reason is that almost every operating system available nowadays can recognize IDE devices. But some operating systems such as Linux, OpenSolaris, and Windows Vista can also work with SCSI and SATA drives. SCSI disks are more commonly used on big data/application servers that require the use of **RAID (Redundant Array of Independent Disks)** technology to be fault-tolerant, although you can also use SATA disks for that kind of environment.


My advice for you is to use the default IDE controller type for most virtual machines unless you need the fastest configuration available, in which case I'd recommend using SATA drives. And if you create a new hard drive using the IDE controller but later you want to use the SATA disk controller instead, don't worry! Just go to your virtual machine's settings window and change the disk controller type! Let me show you how it's done...

Time for action – using a SATA disk controller on a VM

In this exercise, you'll learn how to change the disk controller type from IDE to SATA in an Ubuntu Linux virtual machine.

1. Open VirtualBox, select the `UbuntuVBClone` virtual machine you created in last exercise, and click on the **Settings** button. Select the **Storage** category, and press *Insert* or click on the **Add Controller** button () located at the bottom-right part of the **Storage Tree** panel. Select the **Add SATA Controller** option from the pop-up menu. A new **SATA Controller** element will appear below the **Floppy Controller** element in the **Storage Tree** panel.
2. Now right-click on the `UbuntuVBClone.vdi` slot under the **IDE Controller** element in the **Storage Tree** panel. Then select the **Remove Attachment** option from the pop-up menu to remove the virtual disk image from the **IDE Controller** element.
3. Right-click on the **SATA Controller** element, and select the **Add Hard Disk** option from the pop-up menu. A new hard disk slot will appear right under **SATA Controller**. Select this slot, and then go to the **Attributes** panel to select the `UbuntuVBClone.vdi` disk image from the **Hard Disk** list box:



4. Click on **OK** to close the **UbuntuVBClone – Settings** dialog and return to the VirtualBox main screen, and then start your UbuntuVBClone virtual machine.
5. Wait for your VM to finish booting up and login; then scroll your mouse over the hard disks icon located in the lower-right corner of your virtual machine's window: 
6. A yellow information box will appear to show information about your virtual hard disks in activity. In this case, **SATA Controller** and **SATA Port 0:** indicate your primary hard drive is using the SATA controller.
7. You can close your virtual machine now.

What just happened?

This was a short exercise to show you how VirtualBox lets you choose between the three disk controller types available: IDE, SATA, and SCSI. In this case, I focused on the SATA controller, but you can use the SCSI controller the same way. The only thing to take into consideration is that not every operating system has native support for SATA or SCSI disk controllers; for example, Windows XP needs additional drivers to see SATA disks, but Windows Vista and Ubuntu Linux don't.

So, basically you can't install Windows XP on a SATA drive because you need additional drivers, but you can install Windows Vista, Windows 7, or Ubuntu Linux on IDE or SATA disks. And the same principle applies if you want to use additional hard drives on your virtual machines.

Why don't we just stick with IDE to avoid problems on operating systems without native support for SATA or SCSI disks? Well, for one thing, SATA disks are faster than their IDE counterparts, and you can have up to 30 SATA disks, compared to the three available IDE slots (actually you can use up to four IDE devices, but one is assigned to the CD-ROM drive).



Another scenario where it would be very convenient to be able to switch disk controller types is when you want to use a disk image created with another virtualization product such as VMware or Parallels, and it was generated with a SATA or SCSI controller type. You could attach this disk image to a VirtualBox virtual machine using the default IDE controller, and it would work on an Ubuntu guest, but on Windows guests you must use the same controller type, so you would have to change the default IDE controller assigned by VirtualBox.

Have a go hero – using different types of storage controllers on different guests

It would be excellent if you tried out several combinations of controller types on your Ubuntu, Windows, and OpenSolaris virtual machines. Try changing the default IDE controller, and see if you can boot your virtual machine from a SATA or an SCSI controller instead.

Here's another cool thing you can do: go to <http://www.vmware.com/appliances/>, and download some VMDK images; then use these images on different virtual machines and with different controller types to see if you can detect any differences in performance, or maybe you'll get some error message when trying to use an inappropriate storage controller type. Who knows?

Pop quiz – using different storage controller types

1. If your host PC has a SATA disk, can you create an IDE virtual disk in VirtualBox?
 - a. No, the virtual disk has to be SATA, too.
 - b. Sure, because virtual disks are just files stored in your host PC and, therefore, it doesn't matter which disk controller type your host PC uses.
 - c. You can only use VirtualBox in host PCs with IDE devices.
2. Can you have 3 IDE virtual disks and 10 SATA virtual disks on one virtual machine?
 - a. Yes, you can have up to 4 IDE and 30 SATA disks on one virtual machine.
 - b. No, you can have just one SATA disk or one IDE disk on any given virtual machine.
 - c. You can only have SCSI drives on your virtual machines.

Using IDE and SATA drives on a VM

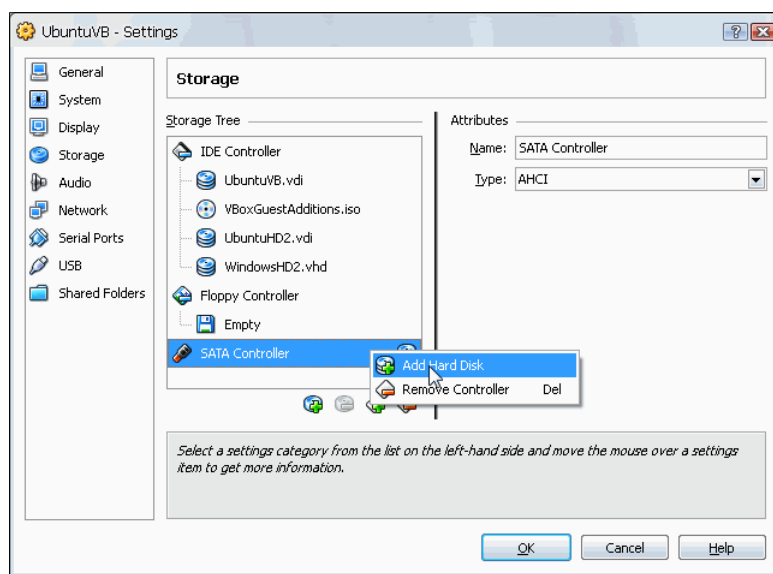
Since SATA devices are becoming more popular each day, along with the fact that they can deliver faster read/write speeds than their IDE counterparts, let's see what happens when you use both disk controller types on the same virtual machine...

Time for action – using IDE and SATA drives

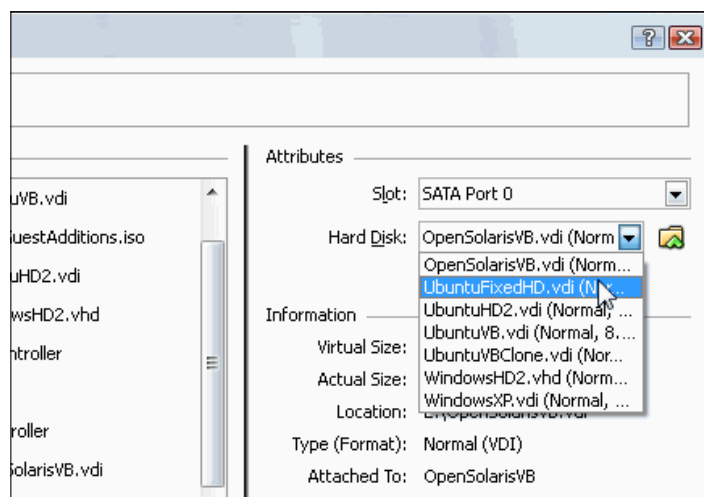
In the following exercise, I'll show you how to add a SATA drive to your UbuntuVB virtual machine that already has three IDE hard disks attached.

1. Open VirtualBox, select your UbuntuVB virtual machine, and click on **Settings** to open the **UbuntuVB – Settings** screen.

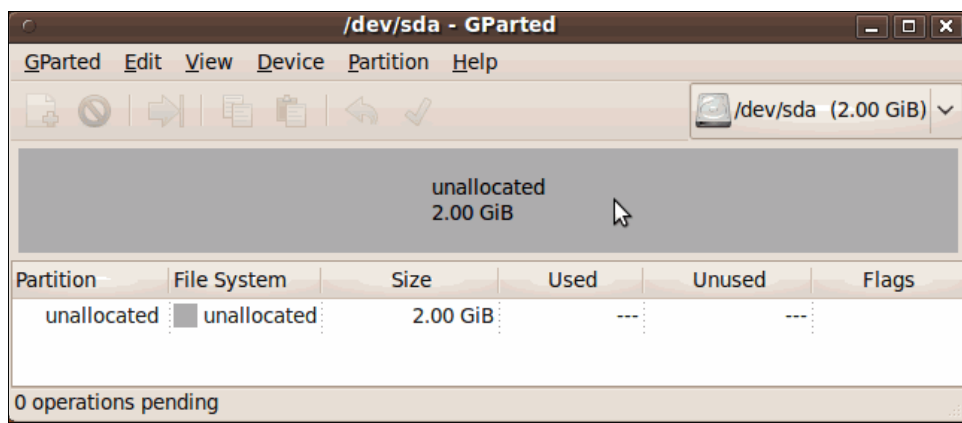
2. Select the **Storage** category, press *Insert*, and select **Add SATA Controller** from the pop-up menu. A new SATA Controller will appear under the **Floppy Controller** in the **Storage Tree** panel.
3. Right-click on the **SATA Controller** element, and select **Add Hard Disk** from the pop-up menu. A new slot will appear under the SATA Controller:



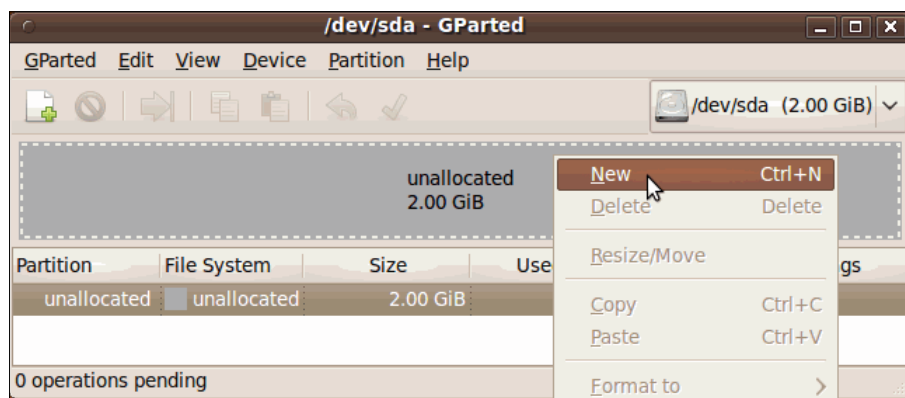
4. Now select the new slot, and then use the **Hard Disk** list box under the **Attributes** panel to select the `UbuntuFixedHD.vdi` disk image you created before:



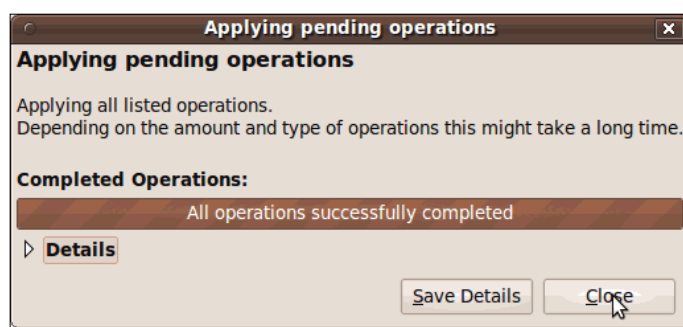
5. Click on **OK** to save your changes and return to the VirtualBox main screen.
6. Now start your UbuntuVB virtual machine, login, and select **System | Administration | GParted** to open the GParted application.
7. Type your `sudo` password when GParted asks for it so you can access your hard drives. Since you added a SATA drive to your Ubuntu system, GParted will show it as the first drive:



8. Select **Device | Create Partition** table to create a partition table for your new drive. The **Create partition table on /dev/sda** dialog will show up. Click on the **Create** button to continue.
9. Right-click on the grey area of your hard drive (unallocated 2.00 GiB), and select **New** from the pop-up menu:



10. The **Create New Partition** dialog will show up. Click on **Add** to leave the default values for your new hard drive partition.
11. Now click on **Apply** to start creating the new partition on your hard drive.
12. The **Apply operations to device** dialog will show up to confirm that you want to apply the changes to your hard drive. Click on **Apply** to continue.
13. GParted will start to create the new partition on your hard drive. When finished, the following dialog will appear:



14. Click on **Close** to continue. Now open a terminal window (**Applications | Accessories | Terminal**), and type the following lines (remember to press *Enter* after each line):

```
sudo mkdir /ubuntufixedhd
sudo mount -t ext2 /dev/sda1 /ubuntufixedhd
```
15. Now you can use your new hard drive in your Ubuntu system!

What just happened?

Ok, now you have three IDE virtual disks and one SATA virtual disk on your UbuntuVB virtual machine! How about that? Try to do that with a real PC, and you'll find out it's not as easy! And if three additional hard drives aren't enough for you, remember you can have up to 30 SATA disks in one virtual machine!

Have a go hero – using SATA and IDE disks on your Ubuntu VM

Now you'll have to start filling up your virtual disks with real data and see how your virtual machine performs! Start copying documents, images, music, and everything your mind comes up with! Then you can experiment attaching your virtual disks to other virtual machines and see for yourself one of the big advantages of virtualization!

Pop quiz – storing data in VirtualBox

1. The Virtual Media Manager lets you:
 - a. Change the memory settings in your virtual machines.
 - b. Create new virtual machines.
 - c. Create new VDI images for your virtual machines.
2. Your boss wants you to create, install, and configure a Windows Vista virtual machine for him. You need to:
 - a. Sit on your boss's computer and create the virtual machine from his VirtualBox installation.
 - b. Create the virtual machine in your VirtualBox installation and then copy the VDI disk image from your VirtualBox installation to your boss's PC.
 - c. Tell him you're not going to do such thing.

Have a go hero – playing with your virtual machines

Now that you have learned some basic information about using virtual storage in VirtualBox, go ahead and create several virtual machines, using several operating systems such as OpenSolaris, Windows XP, Windows Vista, Ubuntu, Fedora, Debian, and so on. Create one or two virtual hard drives, and try to attach them to all the virtual machines you created. This is one good way of learning how to share information among your virtual machines!

Summary

Virtual storage is one of the most critical aspects of virtual machines. I hope you enjoyed the exercises in this chapter. Keep experimenting with your virtual machines because 'practice makes perfect'. Don't you ever forget that!

Specifically, we covered:

- ◆ How VirtualBox uses 'virtual' hard disks so virtual machines can see them as if they were real.
- ◆ How VirtualBox uses the VDI format for all the new virtual machines created and how it can read the VMDK and VHD formats used by other virtualization products (VMware and Microsoft Virtual PC).
- ◆ How to clone hard disks using the Virtual Media Manager and what the difference is between 'cloning' and 'copying'

- ◆ The difference between using fixed and dynamically expanding hard drive images when creating a virtual machine.
- ◆ What hard disk controller to choose when creating a virtual machine, depending on the guest operating system for your virtual machine and the number of drives you want to use.

Now that you can create and use VDI, VHD, and VMDK virtual disks, it's time to learn about the networking capabilities VirtualBox has to offer. In chapter 6, we'll see several exercises about this topic.

6

Networking with Virtual Machines

*Virtual machines can give you a lot of flexibility when you're dealing with several operating systems or different configurations. And networking is a vital part of them. Although you can work on a virtual machine without any network connection available, almost every computer environment nowadays uses some kind of networking: a **local area network (LAN)**, a DSL/Cable connection to the Internet, or a wireless connection.*

VirtualBox offers several networking modes for its virtual machines, and in the following pages, I'll do my best to show you how to make the best use of them with some hands-on exercises I specifically designed for this task.

In this chapter you shall:

- ◆ Practice with all the networking modes offered by VirtualBox
- ◆ Learn how to connect a virtual machine to the Internet and a local LAN through the default NAT mode
- ◆ See the differences between using the NAT mode and the bridged networking mode
- ◆ Learn how to use a virtual machine as a web server and what would be the best networking mode for it
- ◆ Learn how to use the internal networking mode to isolate one or several virtual machines in a private LAN, out of reach from the host computer, from other LANs, or from the Internet
- ◆ Learn how to use the host-only networking mode to restrict outside access for certain virtual machines

Connecting to the default NAT mode

Whenever you create a new virtual machine in VirtualBox, if you don't change the networking settings, your VM will be in NAT mode. And what is that? Well, **NAT** stands for Network Address Translation, which means your virtual machine acts as if it were behind a firewall. If you plan to use your virtual machine as a regular PC doing regular office or school chores such as word processing, Internet surfing, or even playing games, you won't have to change the default NAT mode.

But if you plan to use your virtual machine as a web or application server, you'll need to change to another networking mode because with the NAT mode, your virtual machine can see other computers connected through your host PC's LAN or the Internet, but the other computers won't be able to see your virtual machine.

Before delving into the NAT networking mode, let me introduce you to the network adapter types available in VirtualBox.

Exploring default network adapter types

VirtualBox includes several virtual network adapter types that you can use in your virtual machines (depending on the operating system you intend to use and several other factors, like specialized software):

- ◆ AMD PCNet PCI II
- ◆ AMD PCNet FAST III
- ◆ Intel PRO/1000 MT Desktop
- ◆ Intel PRO/1000 T Server
- ◆ Intel PRO/1000 MT Server
- ◆ Paravirtualized network (virtio-net)

Almost all of the operating systems you can use on your virtual machines support the AMD PCNet FAST III adapter. That's the default adapter type used most of the time. But some operating systems such as Windows Vista and Windows 7 don't include these drivers, so the default adapter type used in Open Solaris, Windows Vista, and Windows 7 virtual machines is the Intel PRO/1000 MT Desktop adapter, as we'll see in the following exercise.

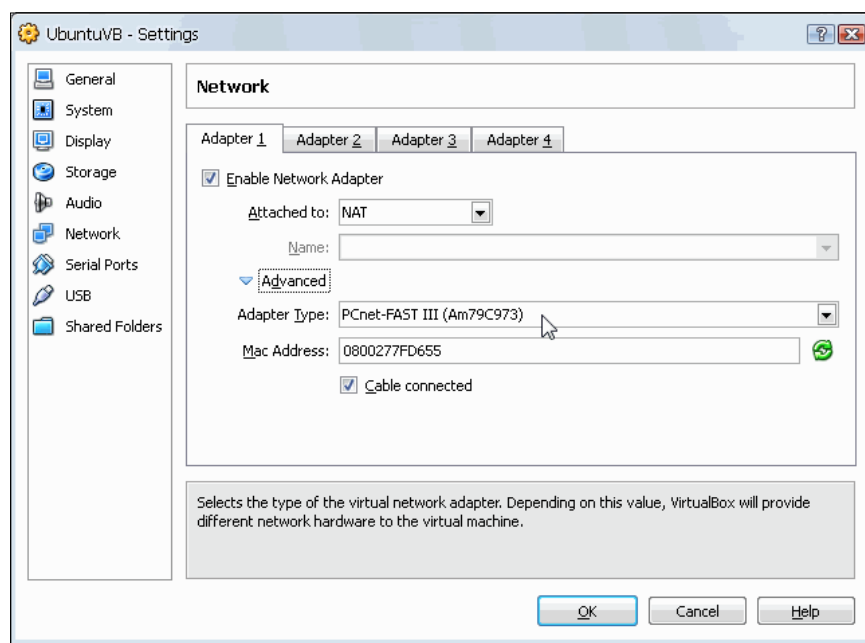
Time for action – viewing the default network adapter types in your virtual machines

In this exercise, I'll show you how we can use VirtualBox to choose different default adapter types, depending on the operating system installed in your virtual machine. I'm using a Windows XP host, but you can use any of the host operating systems allowed in VirtualBox.



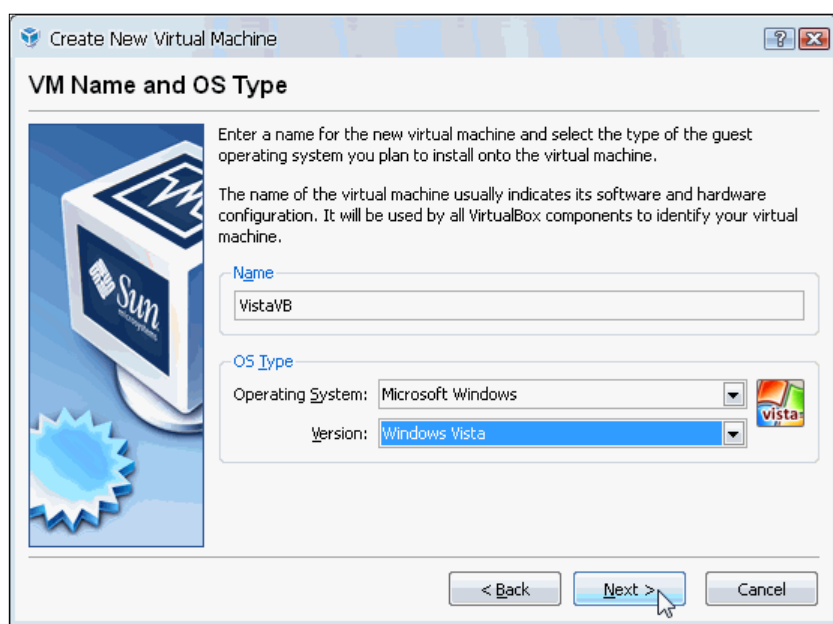
In this exercise and the rest of the exercises in this chapter, you'll need to use the **UbuntuVB** virtual machine we created in previous chapters.

1. Open VirtualBox, select your **UbuntuVB** virtual machine, and click on the **Settings** button to enter the **UbuntuVB – Settings** dialog.
2. Select the **Network** category from the **UbuntuVB – Settings** dialog, click on the **Advanced** setting to expand it, and take a look at the default network adapter:

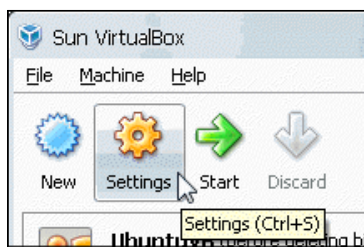


3. As you can see, the **PCnet-FAST III** adapter is the default option in Ubuntu Linux virtual machines. Now click on the **Cancel** button to return to the VirtualBox main screen, and click on the **New** button to create a new virtual machine.

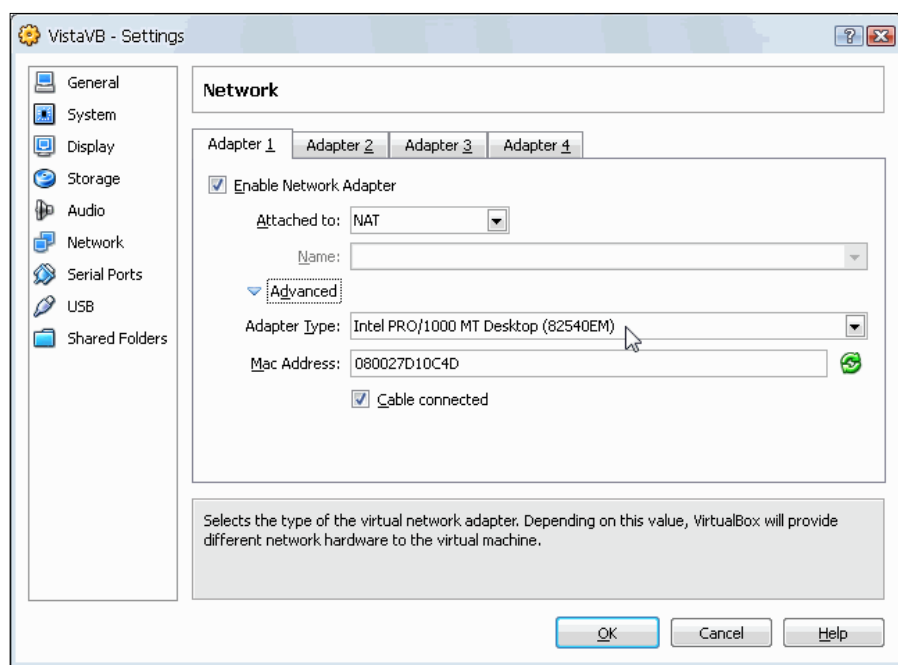
4. The **Create New Virtual Machine** wizard will appear next. Click on the **Next** button to continue.
5. The **VM Name and OS Type** screen will show up. Type **VistaVB** in the **Name** field, and select **Windows Vista** as the operating system. Then click on **Next** to continue:



6. Leave the default **Base Memory Size** value in the **Memory** screen, and click on **Next** to continue.
7. Click on **Next** in the **Virtual Hard Disk** screen to accept the default options.
8. The **Welcome to the Create New Virtual Disk Wizard** screen will appear next. Click on **Next** to continue.
9. Make sure the **Dynamically Expanding Storage** option is selected on the **Hard Disk Storage Type** screen, and click on **Next** to continue.
10. Leave the default values in the **Virtual Disk Location and Size** screen, and click on **Next** to continue.
11. Click on **Finish** twice to exit the **Create New Virtual Disk** and **Create New Virtual Machine** wizards. VirtualBox will return you to the main screen. Now select your new **VistaVB** virtual machine, and click on the **Settings** button:



- 12.** Select the **Network** category in the **VistaVB – Settings** dialog, and take a look at the default network adapter:



- 13.** As you can see, the default network adapter type in a Windows Vista virtual machine (**Intel PRO/1000 MT Desktop**) is different than the one used in an Ubuntu Linux virtual machine. Click on **OK** to close the **Settings** dialog and to return to the VirtualBox main screen.

What just happened?

The default settings VirtualBox chooses for your virtual machines depend on the operating system you intend to use, but generally you can stick with those settings because they're the ones that work with most hardware available nowadays. My advice for you is to stick with the default network adapter type VirtualBox assigns to your virtual machine, unless you experience some irregular behavior. In that case, you can shut down your VM, go to your **Settings** screen, select another network adapter, start your VM again, and see if the problem goes away. Now let's see how the NAT networking mode works...

Pop quiz – working with the default network adapter types

1. If you have a Windows 7 virtual machine, and for some strange reason you can't connect to the Internet, you should:
 - a. Change the default network adapter type to see if the problem goes away.
 - b. Assign more RAM to your virtual machine.
 - c. Create an additional virtual hard disk.
2. Why are there several network adapter types to choose from in VirtualBox?
 - a. Because life is full of choices.
 - b. Because VirtualBox has several sponsors.
 - c. Because some operating systems only support certain types of network adapters.
3. If you're running VirtualBox on a PC with an Intel CPU, you can't use AMD network adapter types.
 - a. True.
 - b. False; you can use any network adapter type with any AMD or Intel CPU.
 - c. False; you can only use an AMD network adapter with an Intel CPU.

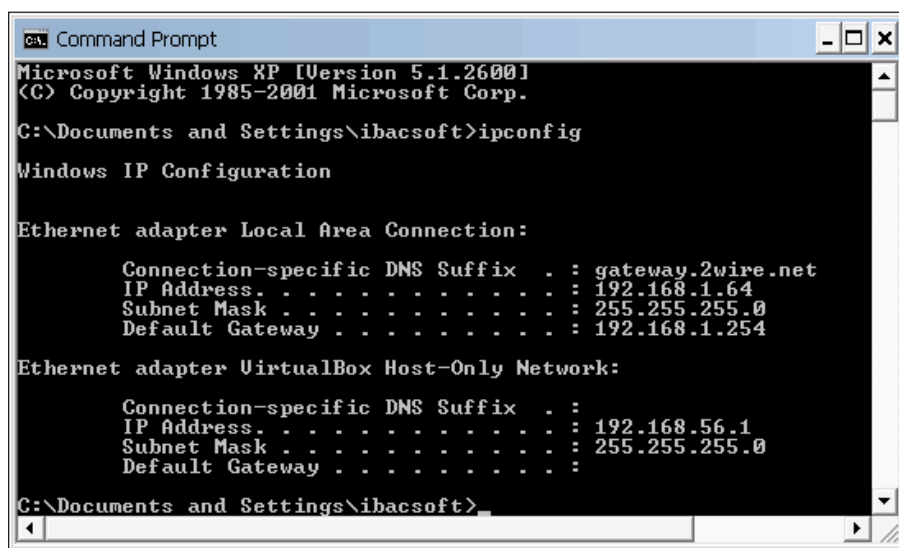
Testing the NAT mode

As I said before, the NAT mode is the one you'll be working in most of the time unless you need to use a specialized virtual machine to act as a web server, for example. This is probably the way your host PC is connected to the Internet: A modem/router firewall lets you surf the web, receive and send email, and do all your daily Internet chores. Let me show you what I mean...

Time for action – accessing the NAT mode in your VM

In this exercise, I'll show you how the NAT mode lets you do some basic networking tasks in your virtual machine as if you were working with a real PC behind a firewall or router. I'll also show you how to set up a web server in one of your virtual machines and test if you can access that web server from your host PC. In this exercise, I used a Windows XP system as the host PC and the UbuntuVB virtual machine we created in previous chapters.

1. If you're using a Windows host, open a **Command Prompt** window and type `ipconfig`, followed by *Enter*. If you're using a Linux host, open a terminal window and type `ifconfig`, followed by *Enter*. Your IP configuration will show up next, as in the following screenshot (Windows host):



```

C:\Documents and Settings\ibacsoft>ipconfig

Windows IP Configuration

Ethernet adapter Local Area Connection:

    Connection-specific DNS Suffix  . : gateway.2wire.net
    IP Address. . . . .               : 192.168.1.64
    Subnet Mask . . . . .             : 255.255.255.0
    Default Gateway . . . . .         : 192.168.1.254

Ethernet adapter VirtualBox Host-Only Network:

    Connection-specific DNS Suffix  . : 
    IP Address. . . . .               : 192.168.56.1
    Subnet Mask . . . . .             : 255.255.255.0
    Default Gateway . . . . .         : 

C:\Documents and Settings\ibacsoft>

```

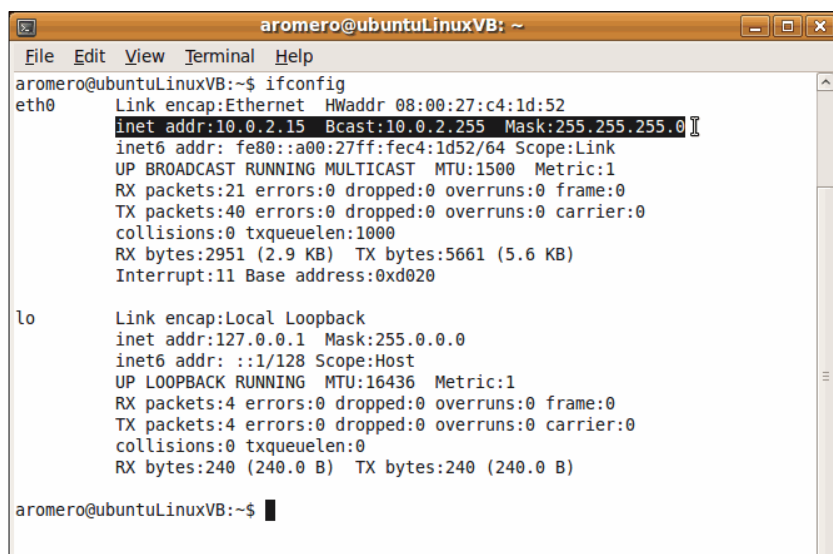
2. There will be an entry for each network adapter physically connected to your host PC and one entry for the VirtualBox Host-Only Network virtual adapter. In the previous screenshot, there are two entries, one for the real network adapter connected to the host PC and one for the VirtualBox Host-Only adapter.



Write down the IP address of each adapter because you'll need this information later.

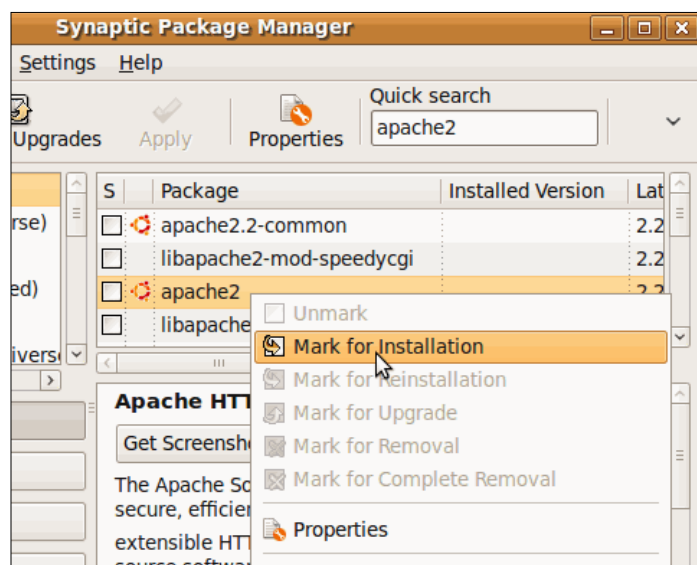
3. Close the **Command Prompt** or the terminal window, open VirtualBox, start your UbuntuVB virtual Machine, and login.

4. Open a terminal window (**Applications | Accessories | Terminal**), type `ifconfig`, and then press *Enter*. The following information will appear:

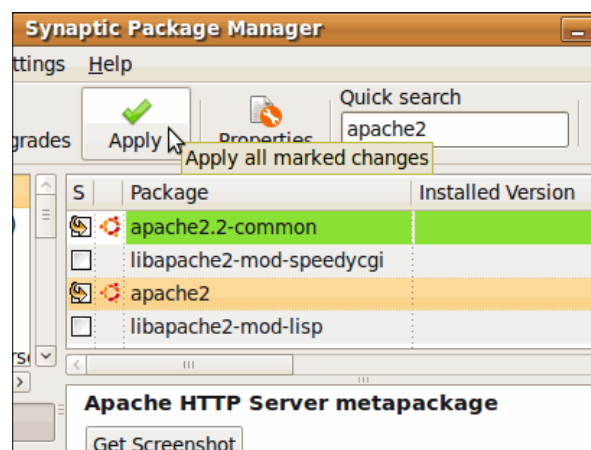


```
aromero@ubuntuLinuxVB: ~  
File Edit View Terminal Help  
aromero@ubuntuLinuxVB:~$ ifconfig  
eth0      Link encap:Ethernet  HWaddr 08:00:27:c4:1d:52  
          inet addr:10.0.2.15  Bcast:10.0.2.255  Mask:255.255.255.0  
          inet6 addr: fe80::a00:27ff:fec4:1d52/64 Scope:Link  
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1  
          RX packets:21 errors:0 dropped:0 overruns:0 frame:0  
          TX packets:40 errors:0 dropped:0 overruns:0 carrier:0  
          collisions:0 txqueuelen:1000  
          RX bytes:2951 (2.9 KB)  TX bytes:5661 (5.6 KB)  
          Interrupt:11 Base address:0xd020  
  
lo        Link encap:Local Loopback  
          inet addr:127.0.0.1  Mask:255.0.0.0  
          inet6 addr: ::1/128 Scope:Host  
          UP LOOPBACK RUNNING  MTU:16436  Metric:1  
          RX packets:4 errors:0 dropped:0 overruns:0 frame:0  
          TX packets:4 errors:0 dropped:0 overruns:0 carrier:0  
          collisions:0 txqueuelen:0  
          RX bytes:240 (240.0 B)  TX bytes:240 (240.0 B)  
  
aromero@ubuntuLinuxVB:~$
```

5. There are two interfaces shown in the previous screenshot: **eth0** and **lo**. The **eth0** interface is the Ethernet adapter VirtualBox assigns to the virtual machine. As you can see, the IP address (10.0.2.15) is completely different from the IP address of your host PC (192.168.1.64, in the screenshot from step 1).
6. Close the terminal window, and select **System | Administration | Synaptic Package Manager** to open the package manager. Type your `sudo` password when Ubuntu asks you for it.
7. Once the Synaptic Package Manager window opens, type `apache2` on the **Quick Search** field, right-click on the `apache2` package from the list, and select **Mark for Installation** from the pop-up menu:

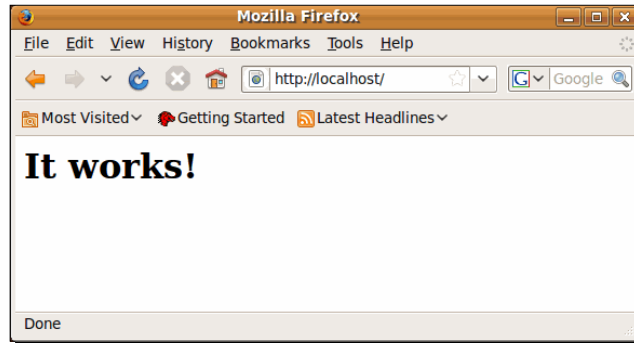


8. The **Mark additional required changes?** dialog will appear next. Click on **Mark** to add the required packages to install along with the `apache2` package. You'll then return to the **Synaptic Package Manager** main screen.
9. Click on the **Apply** button to apply the changes to your Ubuntu virtual machine:

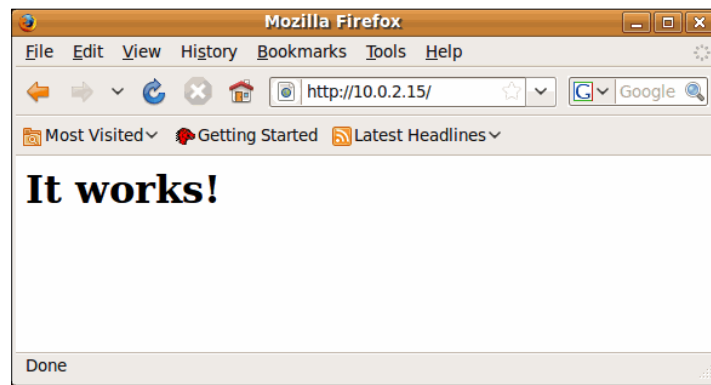


10. The **Apply the following changes** screen will appear next. Click on **Apply** to continue. The **Downloading Package Files** dialog will show up, followed by the **Applying Changes** dialog to indicate the `apache2` package is installing on your Ubuntu virtual machine.

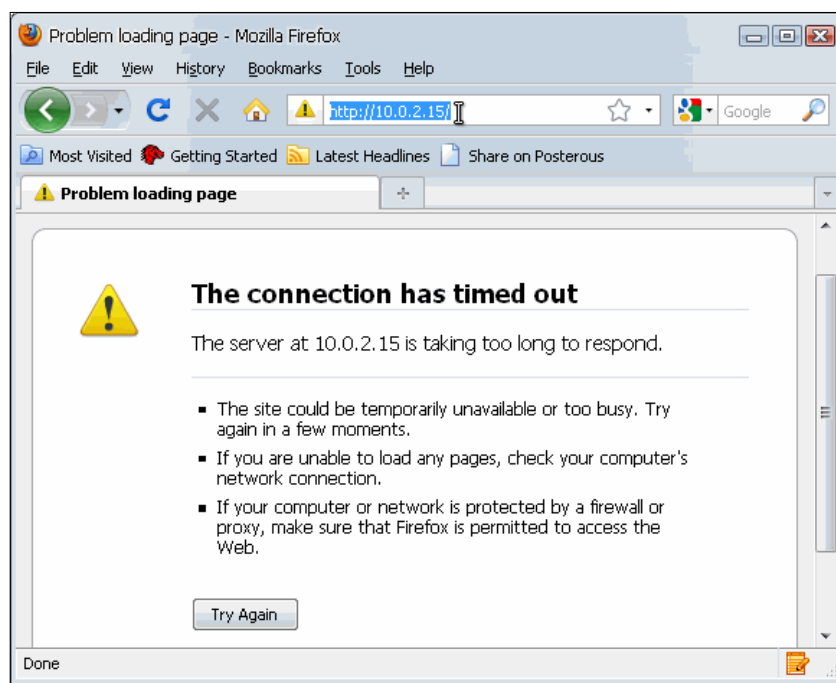
- 11.** Once the installation finishes, the **Changes applied** dialog will appear. Click on **Close**, and then select **File | Quit**, or press **Ctrl+Q** to exit the Synaptic Package Manager.
- 12.** Select **Applications | Internet | Firefox Web Browser** to open your virtual machine's web browser, and type `http://localhost` in the address bar. The following screen should appear to indicate the Apache web browser was successfully installed on your virtual machine:



- 13.** Now replace `localhost` with the IP address you wrote down from step 4 of this exercise (`10.0.2.15` in my case), type **Enter**, and the **It works!** success page will show up to indicate your Ubuntu virtual machine is acting like a web server:



- 14.** To prove your virtual machine can't receive incoming connections from other computers because it's using the NAT mode, open a web browser window in your host PC, and type the same IP address as before (`10.0.2.15` in my case). The following error page will show up:



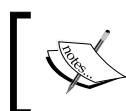
15. Close the web browsers in your host PC and in your virtual machine, and then close your Ubuntu VM.

What just happened?

This exercise showed you how the NAT mode lets your virtual machine access an external network and the Internet as if it were connected behind a firewall. It also showed you that a virtual machine is completely isolated from external inbound connections because if you have a web server installed inside a virtual machine, you can't access it from your host PC or from other computers through a LAN or the Internet.

So, if you want to host a web server in your virtual machine, you definitely cannot use the NAT networking mode because your web server will be completely isolated from the external world! But wait! There's a quick solution to this dilemma! The web server uses port 80, and if you open this port to the world, you can use port-forwarding with NAT, as we'll see in the following section.

Before we continue though, I'd like to say a word about IP addresses and the NAT networking mode. As you saw in the previous exercise, the IP address assigned to my UbuntuVB virtual machine was 10.0.2.15. If I open another virtual machine without shutting down UbuntuVB, the IP address assigned to it will also be 10.0.2.15. And why the same address? Because in reality, each virtual machine is inside its own "private network," so it can't see other virtual machines! But in order to connect to the Internet, the guest operating system inside the virtual machine must use an IP address, so VirtualBox assigns the 10.0.2.15 fake IP address to each VM, and the problem is solved!



VirtualBox assigns the 10.0.2.2 IP address to your host PC, so all your virtual machines can access the host's services through this IP address, known as the **gateway**.

Have a go hero – testing the NAT mode on Windows guests

It would be cool if you could set up a Windows XP/Vista or Windows 7 virtual machine, install the Apache web server, and try out the above exercise. You could also try it out with other Linux distributions such as Fedora, Debian, or SuSE. You could even try it out with OpenSolaris!

Using port-forwarding with the NAT mode

As I said before, even though the NAT mode isolates your virtual machine from external inbound connections, you can use the **port-forwarding** feature to open certain ports so that other PCs can access one or more services in your virtual machine—a web server, for example.

Time for action – enabling port-forwarding in NAT mode

In the following exercise, I'll show you how to use the port-forwarding feature available in the NAT networking mode of your UbuntuVB virtual machine, so that your host PC can access the web server from the previous Time for action—Accessing the NAT mode in your VM section.



When using the VBoxManage command on a Windows host, you need to add the full path to the VBoxManage command; another option is to add the VirtualBox installation folder—the default being "C:\Program Files\Sun\VirtualBox\"—to your system path.

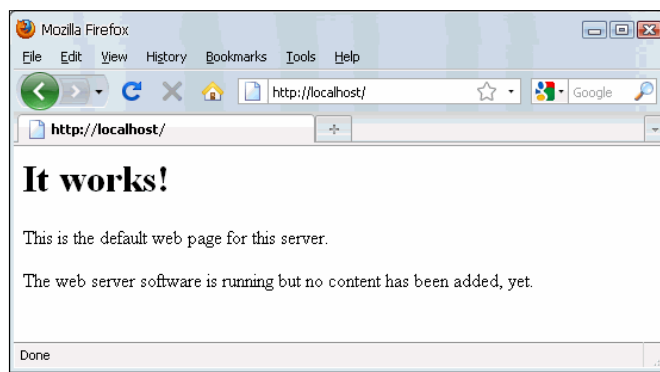
On Linux systems, you don't have to specify the VirtualBox path when using the VBoxManage command.

1. Open a Command Prompt and type the following lines, pressing *Enter* after each one:


```
"C:\Program Files\Sun\VirtualBox\VBXManage" setextradata
"UbuntuVB" "VBoxInternal/Devices/pcnet/0/LUN#0/Config/webserver/
Protocol" "TCP"

"C:\Program Files\Sun\VirtualBox\VBXManage" setextradata
"UbuntuVB" "VBoxInternal/Devices/pcnet/0/LUN#0/Config/webserver/
GuestPort" "80"

"C:\Program Files\Sun\VirtualBox\VBXManage" setextradata
"UbuntuVB" "VBoxInternal/Devices/pcnet/0/LUN#0/Config/webserver/
HostPort" "80"
```
2. Start your **UbuntuVB** virtual machine, and login.
3. Open a web browser window in your host PC, and go to the `http://localhost` URL. Now you'll be able to access your virtual machine's web server, thanks to the port-forwarding feature:



4. Close the web browser window and your **UbuntuVB** virtual machine.

What just happened?

Although the NAT mode is like putting your virtual machines behind a firewall, you can use the port-forwarding feature to open one or more specific ports to access services from your virtual machines, such as a file or web server.

To enable port forwarding, you need to use the `setextradata` command from the `VBXManage` command-line interface because this is an advanced feature you can't adjust within the standard VirtualBox GUI. You must adjust three values: the **protocol**, the **guest port**, and the **host port**. That's why you need to execute the `VBXManage setextradata` command three times for every service in your virtual machine that you want to open to your LAN or to the Internet.

Basically, the three commands executed in step 1 specify that all TCP connections to port 80 on your host PC will be forwarded to port 80 on your **UbuntuVB** virtual machine. Since a web server uses port 80, you must use this value as the guest port number, but you can change the host port value if you don't want to use port 80. However, if you use a different port number, you'll need to specify it in the URL. For example, if you use port 9000, you'll have to type `http://localhost:9000` in your web browser to access your virtual machine's web server, instead of just typing `http://localhost`.

The above exercise assumes you're using one of the PCNet virtual network adapters. If you're using one of the Intel Pro/1000 virtual adapters, just replace `pcnet` with `e1000`. Also, the `0` after `pcnet` represents the first interface instance (**Adapter 1** in the VirtualBox GUI). If you're using another interface instance, just replace the `0` with `1`, `2`, or `3`. And you can replace the `webserver` part with any name of your choice, so you can identify the forwarding configuration if there's more than one.

And in case you're wondering if it wouldn't just be easier to install the Apache web server on your host PC, let me tell you that there are several advantages to using a virtual machine to run a web server or another service through the port forwarding feature in the NAT mode:

- ◆ If there's a bug in the service or it crashes up, your host PC cannot be compromised.
- ◆ You can run the service in a different operating system than the host operating system.
- ◆ You don't have to install software in your host PC that could mess up your main work environment.

As good as it sounds, the NAT networking mode has some limitations to consider:

- ◆ Some network debugging tools such as ping or tracerouting may not work due to certain ICMP protocol limitations when using virtual machines.
- ◆ NetBios name resolution may not always work due to an unreliable UDP broadcast reception mechanism. This can be a problem when using shared folders from other PCs in a network, but you can overcome this obstacle if you use the IP address of the PC you're trying to access instead of its name: for example, `192.168.1.86\share` instead of `WindowsServer\share`.
- ◆ TCP and UDP are the only protocols supported. This may not be a problem for regular network use.
- ◆ You cannot forward host ports lower than 1024 on Unix-based hosts. This means you can't use port 80 to forward all connections to a web browser in your virtual machine; you'll have to use a port number greater than 1024 for your host.

Now that you know the advantages and disadvantages of using the NAT networking mode along with port forwarding, let me show you another networking mode that lets your virtual machine communicate with other PCs in a LAN without the need to do port forwarding...

Have a go hero – using port forwarding with the NAT networking mode

Suppose you need to access your Ubuntu virtual machine from other PCs in your LAN by means of a terminal window so that you can do basic administrative chores. First, you'll need to install the SSH package on your **UbuntuVB** virtual machine. Open a terminal window, and type `sudo apt-get install ssh` to install the open-ssh client and server packages. Then use the `VBoxManage setextradata` command to forward all connections to port 22 from your host PC to port 22 on your virtual machine (if you're using a Unix-based host like Ubuntu, you'll need to use a port number greater than 1024, so you can replace host port 22 with 2222). Finally, check if you can access your UbuntuVB virtual machine from your host PC using an SSH client program (on Windows hosts you can use the PuTTY SSH from <http://www.putty.org>).

Testing a server operating system in the bridged networking mode

The NAT mode lets you do all the common chores a regular user does, such as using a web browser, reading email, sharing files between host and guest, and so on. But what if you want to test a server operating system such as Windows 2003 or Ubuntu Server Edition? Or how about installing a web server on one of your virtual machines to test some web applications from your host PC, from other virtual machines, or from the Internet without having to mess up your host PC's configuration? That's where the bridged networking mode comes into action...

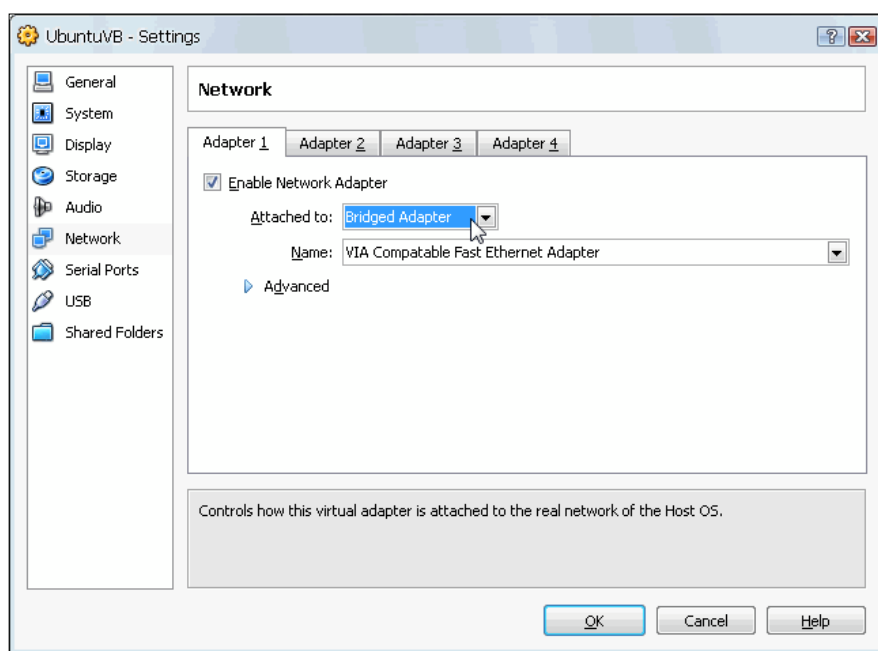
Accessing your VM's web server from your host PC

This is one of my favorite features VirtualBox has to offer because I constantly have to test new features on a website, and for that I need to install one or more virtual machines with different operating systems and configurations. Basically, I use one Ubuntu Linux or Windows 2003 Server virtual machine as a web server with the bridged networking mode activated so that I can access the web server from other virtual machines, from the host PC, from other PCs on a local LAN, or even from the Internet!

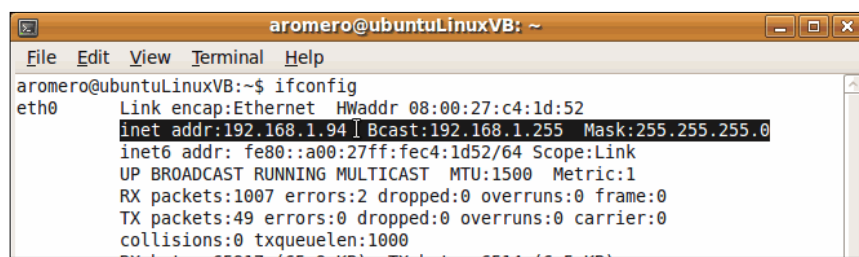
Time for action – changing your virtual machine to bridged networking mode

In this exercise, I'll show you how to change your virtual machine's networking mode from NAT to bridged so that you can access, from your host PC, the web server you installed on the **UbuntuVB** virtual machine in the previous exercise.

1. Select your **UbuntuVB** virtual machine, and click on the **Settings** button to go to the **UbuntuVB – Settings** page.
2. Select the **Network** category, click on the **Attached to:** list box, and select the **Bridged Adapter** option:



3. Click on **OK** to close the **UbuntuVB – Settings** dialog and return to VirtualBox, then start your **UbuntuVB** virtual machine, and login with your administrator account.
4. Open a terminal window (**Applications | Accessories | Terminal**), and type `ifconfig`, followed by `Enter`. Now you'll see something like this:

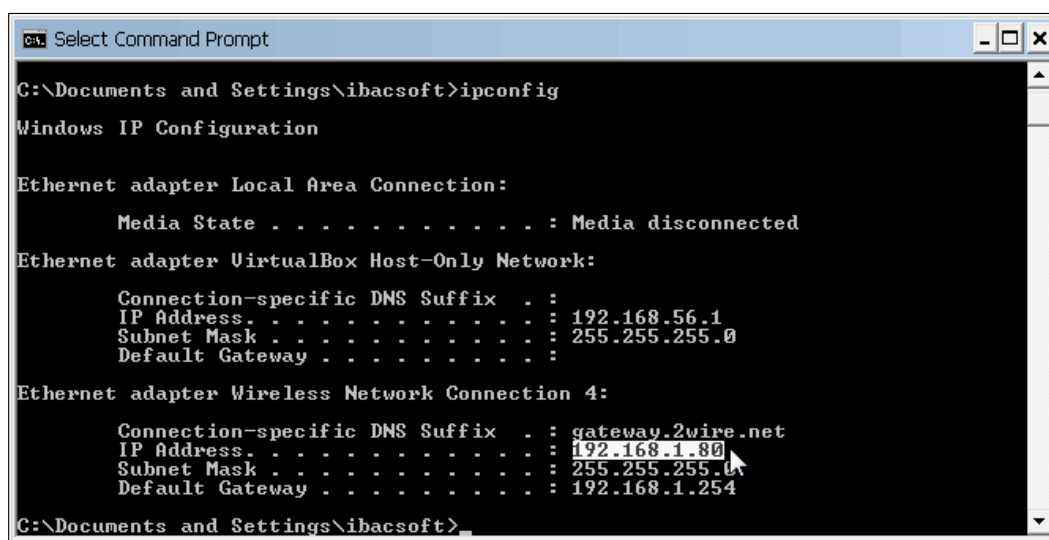


```

aromero@ubuntuLinuxVB:~$ ifconfig
eth0      Link encap:Ethernet  HWaddr 08:00:27:c4:1d:52
          inet addr:192.168.1.94  Bcast:192.168.1.255  Mask:255.255.255.0
          inet6 addr: fe80::a00:27ff:fec4:1d52/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:1007 errors:2 dropped:0 overruns:0 frame:0
          TX packets:49 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000

```

5. If you compare the IP address of your virtual machine in step 4 of the Time for Action – Accessing the NAT mode in your VM exercise (10.0.2.15) and the IP you obtained in this exercise (192.168.1.94), you'll notice they're different. That's because in this exercise, your virtual machine gets its IP address from the same network that your host PC does.
6. If you're using a Windows host, open a Command Prompt window, and type `ipconfig` followed by *Enter*. If you're using a Linux host, open a terminal window, and type `ifconfig` followed by *Enter*. The following screenshot shows the result of executing the `ipconfig` command on a Windows XP host:



```

C:\Documents and Settings\ibacsoft>ipconfig

Windows IP Configuration

Ethernet adapter Local Area Connection:

    Media State . . . . . : Media disconnected

Ethernet adapter VirtualBox Host-Only Network:

    Connection-specific DNS Suffix  . : 
    IP Address. . . . . : 192.168.56.1
    Subnet Mask . . . . . : 255.255.255.0
    Default Gateway . . . . . : 

Ethernet adapter Wireless Network Connection 4:

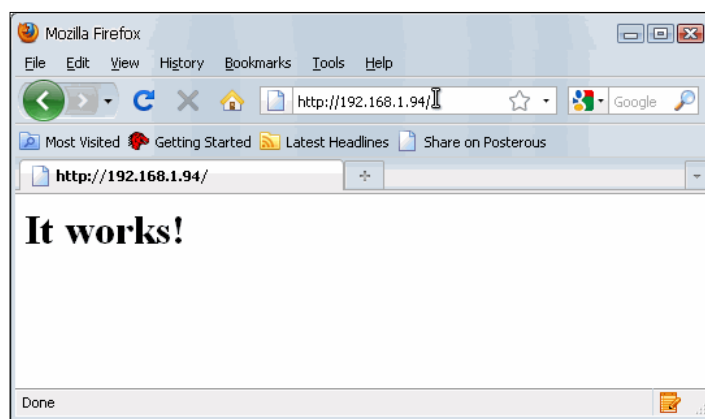
    Connection-specific DNS Suffix  . : gateway.2wire.net
    IP Address. . . . . : 192.168.1.80
    Subnet Mask . . . . . : 255.255.255.0
    Default Gateway . . . . . : 192.168.1.254

C:\Documents and Settings\ibacsoft>

```

7. In this case, there's a wireless adapter connected to the host, and the IP address assigned to it is 192.168.1.80. It's in the same range as the IP address assigned to your UbuntuVB virtual machine (192.168.1.94) in step 4 of this exercise.

8. Open a web browser window on your host PC, and type your virtual machine's IP address. Now you will be able to see the **It works!** page from your virtual machine's web server:



9. You can close your host PC's web browser now, but leave your UbuntuVB virtual machine running for the next exercise.

What just happened?

In this short exercise, you saw how to use the bridged networking mode in one of your virtual machines to access its web server from your host PC. When using the bridged mode, your virtual machine gets an IP address from the same range as your host PC and the other PCs on your LAN (local area network), in case you are inside one. This means you can access the UbuntuVB virtual machine web server from any computer connected to your host PC through a LAN. And if you're connected to the Internet through a router or a firewall, you can open port 80 and use your virtual machine as a web server!

Before we continue with the other networking modes, I want to talk a little about IP addresses. When you have several PCs interconnected in a LAN, you need to use a device called DHCP server. The **DHCP** acronym stands for **Dynamic Host Configuration Protocol**. This device is in charge of assigning an IP address to each PC connected to the LAN.

Every PC in the LAN must have an IP address of the same range. In the previous exercise, we used the 192.168.1.94 IP address for the **UbuntuVB** virtual machine and the 192.168.1.80 IP address for the Windows host PC. Both IP addresses are in the same range because the first 3 pairs of numbers (192.168.1) are the same; consequently, we can assume every address between 192.168.1.0 and 192.168.1.255 is in the same range.

When a virtual machine runs in the bridged networking mode, it uses the same DHCP server as the other PCs in your LAN. That's why it always has an IP address in the same range as your host PC. Later on we'll see how VirtualBox can provide a DHCP server to assign IP addresses to its virtual machines so that they can be isolated from your host PC or the other PCs on your LAN.

Accessing your VM's web server from another VM

Ok, now you know how to install and configure a web server in one of your virtual machines, and you know how to access it from your host PC or any other computer connected to your LAN. But what if you need to test your virtual machine's web server from another virtual machine?

Time for action – accessing your VM's web sever from another VM

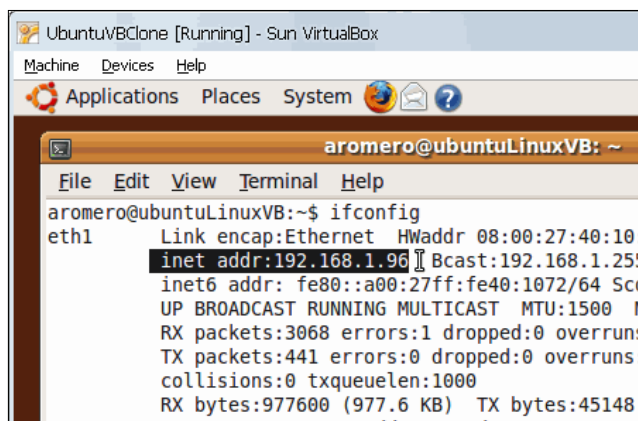
In this exercise, I'll show you how you can open another virtual machine and use the bridged networking mode to access the web server we configured in the previous exercise.



For the next exercise, you'll need at least 2 GB of RAM on your host PC because each virtual machine will occupy at least 384 MB of RAM. Since you're going to have two VMs open, you'll need 764 MB of RAM just for them. VirtualBox warns you about using more than half of your total RAM for virtual machines because the host system could behave erratically.

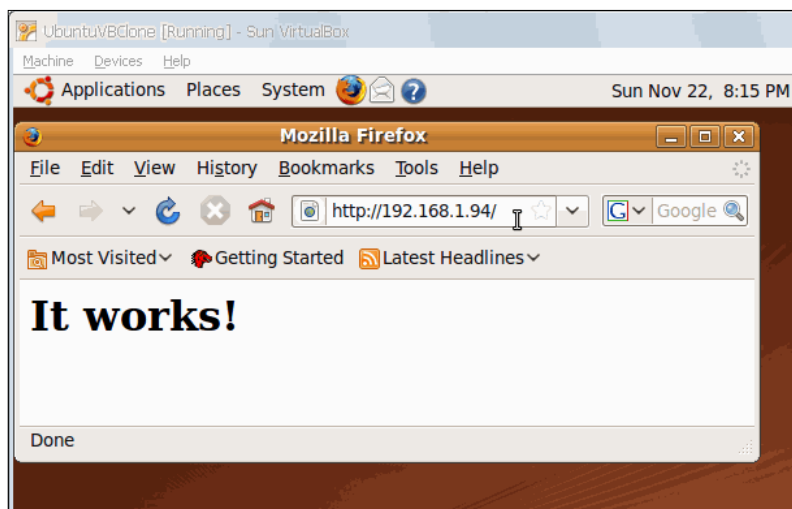
1. With your `UbuntuVB` virtual machine running, select your `UbuntuVBClone` virtual machine, and click on the **Settings** button to go to the **UbuntuVBClone – Settings** page.
2. Select the **Network** category, click on the **Attached to:** list box, and select the `Bridged Adapter` option.
3. Click on **OK** to close the **UbuntuVBClone – Settings** dialog and return to VirtualBox; then start your `UbuntuVBClone` virtual machine, and login with your administrator account.

4. Open a terminal window (**Applications | Accessories | Terminal**), and type `ifconfig`, followed by `Enter`. You'll get similar results as in step 4 of the previous exercise, only this time it will be a different IP address:



```
aromero@ubuntuLinuxVB: ~$ ifconfig
eth1      Link encap:Ethernet  HWaddr 08:00:27:40:10:
          inet addr:192.168.1.96  Bcast:192.168.1.255
          inet6 addr: fe80::a00:27ff:fe40:1072/64 Sc
          UP BROADCAST RUNNING MULTICAST  MTU:1500  M
          RX packets:3068 errors:1 dropped:0 overruns:
          TX packets:441 errors:0 dropped:0 overruns:
          collisions:0 txqueuelen:1000
          RX bytes:977600 (977.6 KB)  TX bytes:45148
```

5. Close the terminal window in your UbuntuVBClone virtual machine, open a web browser window (**Applications | Internet | Firefox Web Browser**), and type the IP address of your UbuntuVB virtual machine (192.168.1.94 in the previous example) to access the web server. You should get the **It Works!** success page:



6. You can close both virtual machines now.



Remember to shut them down properly.

What just happened?

This little exercise demonstrates how you can use one virtual machine as a web server and make it available through your host's LAN, along with other virtual machines on your host. And what about opening your VM's web server to the Internet world? No problem! If you connect to the Internet through a DSL or Cable modem/router, you need to simply open port 80 according to your modem/router's user manual, get a dynamic host account on <http://www.no-ip.org> or <http://www.dyndns.org>, and start your own blog, e-shop, or forum!

As I explained in the What just happened? section of the previous exercise, when using the bridged networking mode, your virtual machines get their IP addresses from the same DHCP server as your host PC or the other PCs in your LAN. This way, you can have several virtual machines running, and each one will behave exactly as if it were a real PC on your LAN!

Now let's see how to isolate your virtual machines from a network connection...

Have a go hero – testing the bridged mode on Windows guests

How about testing the bridged networking mode out on a Windows XP/Vista or Windows 7 virtual machine? And how about trying the previous exercises with other Linux distributions such as Fedora, Debian, or SuSE? You can also try using an OpenSolaris virtual machine!

Pop quiz – using the bridged networking mode

1. The computer department at work wants to set up a file server so that everybody in the office can share files and documents. The department needs to:
 - a. Hire a professional computer consultant to install the database server along with the required network connections.
 - b. Use a virtual machine to set up the file server, and then set it up with the bridged networking mode so it can be seen by all the PCs in your office's LAN.
 - c. Set up a virtual machine as a file server, and use the NAT networking mode so every PC in your LAN can communicate with it.
2. What would be the best networking mode for a web server?
 - a. The NAT mode.
 - b. The internal networking mode.
 - c. The bridged networking mode.

3. If you are running a virtual machine and your host PC is connected to a LAN:
 - a. The NAT networking mode lets your VM communicate with the other PCs in your LAN but not with the host PC.
 - b. The Host-Only networking mode lets your VM surf the web.
 - c. The bridged networking mode lets your virtual machine communicate with your host PC.

Using the 'Not Attached' mode

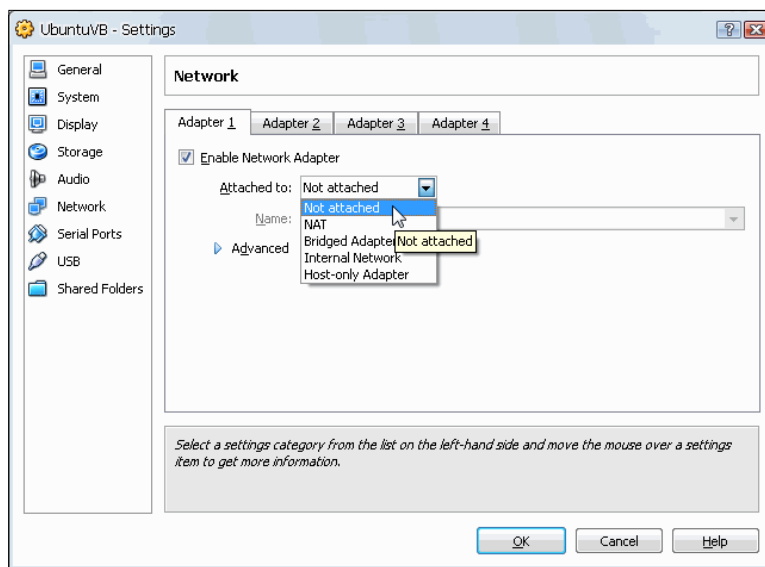
The "Not Attached" networking mode makes your virtual machine think that there's no cable connected to its network adapter; this mode is for those times when you want your virtual machine to be completely isolated from your host PC, the Internet, other PC's in your LAN, and even from other virtual machines.

"But I could just uncheck the **Enable Network Adapter** checkbox in the **Settings** page, and it would be the same thing!" was the first thing that came into my mind when I read about this mode. Although you can achieve the same objective because your virtual machine will be disconnected from any network available to your host PC, sometimes you need to start your virtual machine without a network connection, like if you need to force an operating system reconfiguration, for example. In short, if you don't plan to use a network connection with your virtual machine, you can disable the network adapter completely, but if you just need to start your VM temporarily disconnected from the network, use the "Not Attached" mode instead.

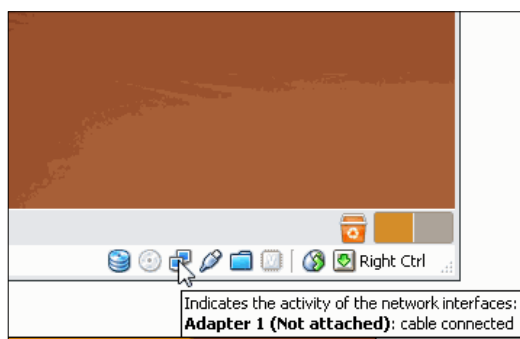
Time for action – isolating a VM with the 'Not Attached' mode

In this exercise, I'll show you how to isolate the web server in your `UbuntuVB` virtual machine from the outside world.

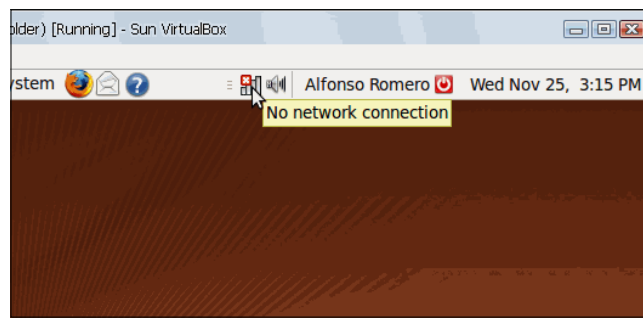
1. Select your `UbuntuVB` machine on VirtualBox's main screen, and click on the **Settings** button to go to the **UbuntuVB – Settings** page.
2. Go to the **Network** category, and then select the `Not attached` mode on the **Attached to** list box:



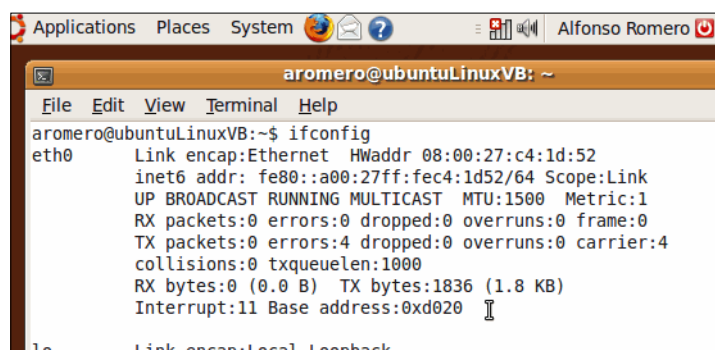
3. Click on **OK** to return to the VirtualBox main screen. Then start your **UbuntuVB** virtual machine and login.
4. If you look at the bottom-right part of your virtual machine window, you'll see that the status bar indicates that your network cable is connected, but there's no adapter attached to your virtual machine:



5. And if you look at your **UbuntuVB** virtual machine's menu bar, you'll see that there's no network connection:



6. Now open a terminal window in your **UbuntuVB** virtual machine, and type `ifconfig` followed by *Enter* to see your network configuration:



7. As you can see, the **eth0** interface is present but has no IP address assigned because your virtual machine thinks there's no network connection.
8. Close your **UbuntuVB** virtual machine for the next exercise.

What just happened?

This last exercise showed you what happens when you select the Not Attached networking mode in a virtual machine. It will be completely isolated from external connections, and if you need to reconnect your network, you'll have to shut down your virtual machine, change the networking mode from Not Attached to NAT or some other mode, and restart your VM again.

But what if you need to disconnect your network adapter when your virtual machine is running? Well, then the "Not Attached" mode is not the best choice because you can't change it when your VM's running. In the next section, we'll address this issue with a concise example.

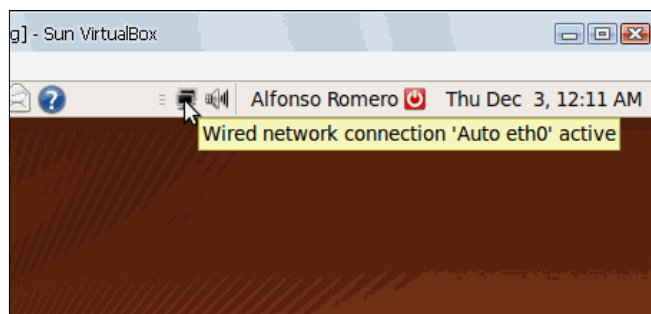
Disconnecting your virtual machine from the network without shutting it down

The Not Attached mode lets you run your virtual machine without any network connections, but you can't enable/disable the network when it's running. That's why I'm going to show you how to connect/disconnect your virtual machine from the network without having to shut it down, change the network settings, and start it up again.

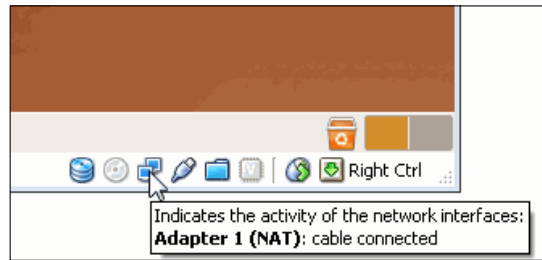
Time for action – connecting/disconnecting your VM from the network

In this exercise, you'll see how easy it is to connect/disconnect your virtual machine from the network while it's running. I'm going to use the NAT mode, but you can also do this in any other networking mode.

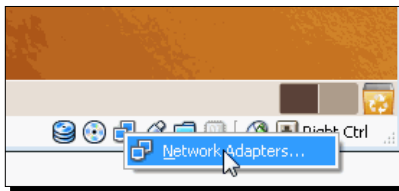
1. Select your **UbuntuVB** machine on VirtualBox's main screen, and click on the **Settings** button to go to the **UbuntuVB – Settings** page.
2. Go to the **Network** category, and then select the **NAT** mode on the **Attached to** list box.
3. Click on **OK** to return to the VirtualBox main screen, start your **UbuntuVB** virtual machine, and login with your administrator account.
4. Once you're logged in, move your mouse over the network icon on your Ubuntu virtual machine's menu bar to verify that your network connection is active:



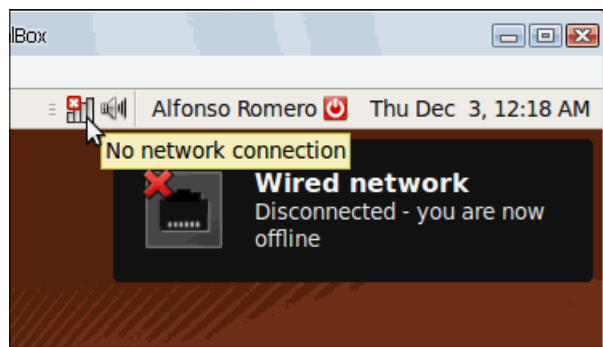
5. Now move your mouse over the network adapters icon located at the bottom-right corner of your UbuntuVB virtual machine's screen:



6. To disconnect your virtual machine's network cable, right-click on the network adapters icon, and select the **Network Adapters...** option:



7. The **Network Adapters** dialog will appear next. Click on the **Cable connected** checkbox to 'unplug' the network cable from your virtual adapter. Right after doing that, the network icon in your Ubuntu virtual machine's menu bar will indicate that the network cable is disconnected:



8. You can connect back your virtual machine's network cable by enabling the **Cable connected** checkbox from the **Network Adapters** dialog. Close your **UbuntuVB** virtual machine when you're finished.

What just happened?

Ok, now you know how to connect/disconnect the network cable from your virtual machines while they're running! Although I've never used this feature before, it's good to know you can disconnect your virtual machine's network cable whenever you need to without having to shut down your VM, change your network settings, and start it up again, don't you think?

Now let's go and see the rest of the networking modes VirtualBox has to offer...

Have a go hero – don't forget to experiment with Windows guests

If you have Windows virtual machines, go and try out the previous exercise on them so you can experience for yourself the differences between guest operating systems. Try to use other Linux distributions too, and even an OpenSolaris guest.

Using the Internal Networking mode

Suppose you need to test a new web application on a virtual machine, and you want to test it from another virtual machine, but you don't want it to be open to the other PCs on your LAN. The bridged networking mode will let you run a web server on a virtual machine, but it will be open to the other PCs on your LAN and to the Internet, in case you open up port 80 in your modem/router's firewall. This is where the internal networking mode comes into play: your virtual machines can communicate between each other, but you can't access them from your host PC or from another PC on your LAN! And they are isolated from the Internet, too!

This little magic trick has some requirements, though: you must enable a DHCP server to assign your virtual machines an IP address automatically each time you start them because they must not be on the same IP address range of your host PC or the other PCs on your LAN. Now let's see how to do that...

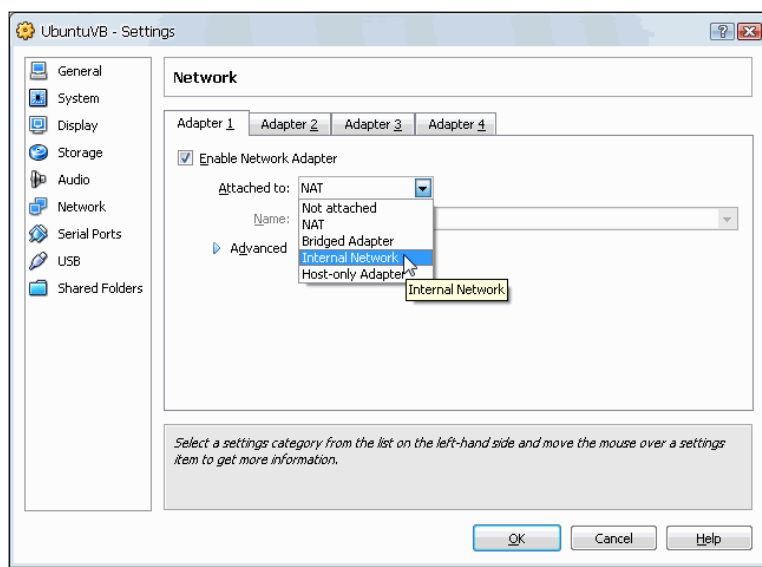
Time for action – communicating between VMs only

In this exercise, I'll show you how to isolate your virtual machines from the rest of the PCs in your LAN or from inbound Internet connections to create a private network for your VM's only. I'm going to use a Windows XP host, but you can use any host allowed by VirtualBox, like Ubuntu Linux.

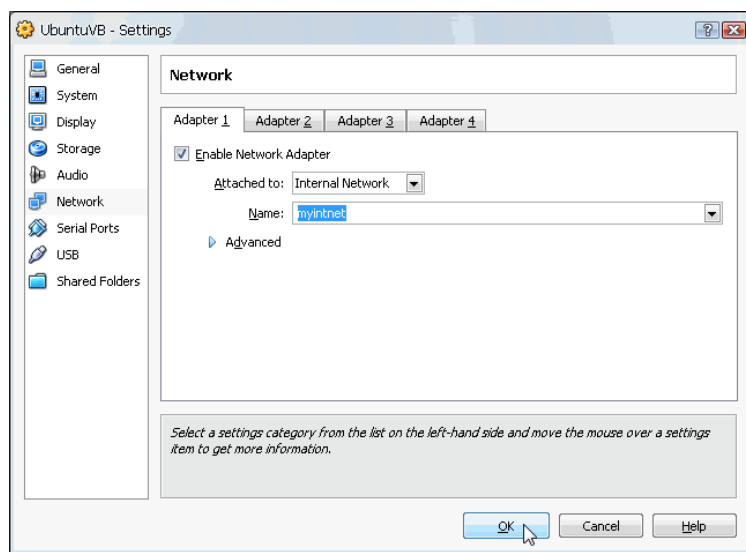


Just be sure to adjust the command in step 4 to suit your operating system.

1. Select your **UbuntuVB** machine on VirtualBox's main screen, and click on the **Settings** button to go to the **UbuntuVB – Settings** page.
2. Go to the **Network** category, and then select the **Internal Network** mode on the **Attached to** list box:



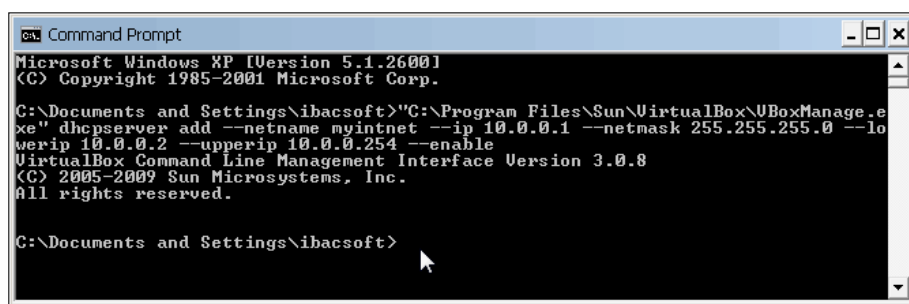
3. The **Name** field will show the default `intnet` name for your internal network. Replace that value with `myintnet`, and click on **OK** to continue:



4. Open a command prompt or a terminal window on your host PC. Then type the following command (don't forget to press *Enter* after typing it):

```
"C:\Program Files\Sun\VirtualBox\VBoxManage.exe" dhcpserver add
--netname myintnet --ip 10.0.0.1 --netmask 255.255.255.0
--lowerip 10.0.0.2 --upperip 10.0.0.254 --enable
```

5. The VirtualBox Command Line Management Interface will respond with the following message to indicate the DHCP server started successfully:

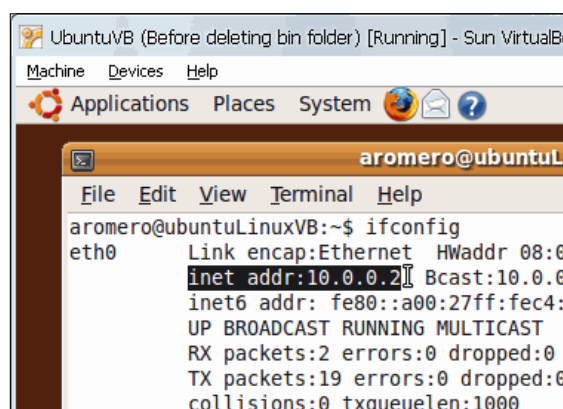


```
Microsoft Windows XP [Version 5.1.2600]
(C) Copyright 1985-2001 Microsoft Corp.

C:\Documents and Settings\ibacsoft>"C:\Program Files\Sun\VirtualBox\VBoxManage.exe" dhcpserver add --netname myintnet --ip 10.0.0.1 --netmask 255.255.255.0 --lowerip 10.0.0.2 --upperip 10.0.0.254 --enable
VirtualBox Command Line Management Interface Version 3.0.8
(C) 2005-2009 Sun Microsystems, Inc.
All rights reserved.

C:\Documents and Settings\ibacsoft>
```

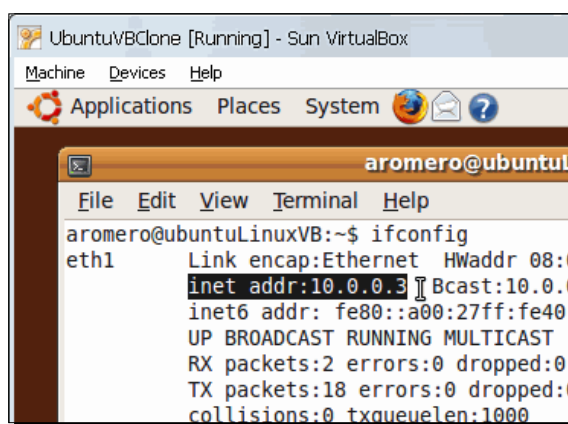
6. Now start your UbuntuVB virtual machine and login.
7. Open a terminal window (**Applications | Accessories | Terminal**), and type `ifconfig` followed by *Enter* to see your network configuration:



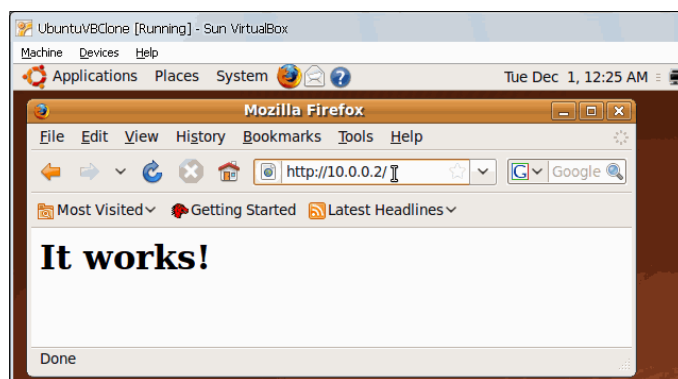
```
aromero@ubuntuLinuxVB:~$ ifconfig
eth0      Link encap:Ethernet  HWaddr 08:00:06:00:00:00
          inet addr:10.0.0.2  Bcast:10.0.0.255  Mask:255.255.255.0
          inet6 addr: fe80::a00:27ff:fec4::1  Prefixlen 64
          UP BROADCAST RUNNING MULTICAST
          RX packets:2 errors:0 dropped:0
          TX packets:19 errors:0 dropped:0
          collisions:0 txqueuelen:1000
```

8. The **eth0** interface has the inet address 10.0.0.2, thanks to the DHCP server we started in step 4 of this exercise.
9. Now go to the VirtualBox main screen, select the **UbuntuVBClone** virtual machine, and click on the **Settings** button to open the **UbuntuVBClone – Settings** window.

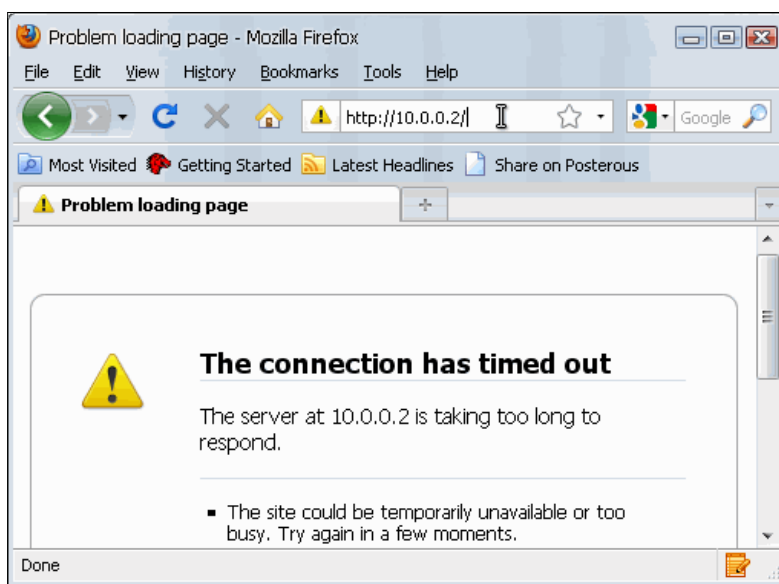
- 10.** Go to the **Network** category, and then select the **Not attached** mode on the **Attached to** list box.
- 11.** If the **Name** field shows the default `intnet` name for your internal network, just replace that value with `myintnet` like in step 3 of this exercise.
- 12.** Click on **OK** to return to the VirtualBox main screen, start your **UbuntuVBClone** virtual machine, and login.
- 13.** Open a terminal window (**Applications | Accessories | Terminal**), and type `ifconfig` followed by *Enter* to see your network configuration:



- 14.** The **eth0** interface of your UbuntuVBClone virtual machine has the inet address `10.0.0.3`, thanks to the DHCP server we started in step 4 of this exercise.
- 15.** Open a web browser on your UbuntuVBClone virtual machine, and type `http://10.0.0.2` (remember to use the IP address assigned to your UbuntuVB virtual machine) to see if you can access the web server in UbuntuVB:



- 16.** Now open a web browser in your host PC, and type `http://10.0.0.2`—or the IP address assigned to your **UbuntuVB** virtual machine—to see if you can access the web server in **UbuntuVB**:



- 17.** The **connection timed out** error message indicates that your host PC cannot access your UbuntuVB virtual machine's web server. You can shut down your **UbuntuVB** and **UbuntuVBClone** virtual machines now.

What just happened?

This exercise showed you how the internal networking mode allows network communication between virtual machines only. Your host PC is unable to access your virtual machines through the network interface, and if you're connected to a LAN, the other PCs connected to it will also be unable to access your virtual machines because they are completely isolated from external networks, including the Internet. That also means you can't browse the web, check your email, or do anything related to the Internet from your virtual machines, so keep that in mind when selecting this networking mode.

When using the internal networking mode, you need to define a name for your internal network. In step 3, you used the name `myintnet`, and for each virtual machine you plan to use on this internal network you need to type `myintnet` in the **Name** field. Then, in step 4, you executed the `VBoxManage dhcpserver` command along with some parameters to start a DHCP server for your `myintnet` internal network:

Parameter	Definition	Example
--netname	The name of the internal network.	--netname myintnet
--ip	The IP address of the DHCP server itself.	--ip 10.0.0.1
--netmask	The netmask is a four-part number that masks out the network part of an IP address to identify each individual virtual machine on the network. The 255 value hides that portion, and the 0 value makes it visible. A 255.255.255.0 value means that each virtual machine will be identified with the last portion of the IP address.	--netmask 255.255.255.0
--lowerip	The lowest IP address in the valid range used by the DHCP server to assign IP addresses.	--lowerip 10.0.0.2
--upperip	The highest IP address in the valid range used by the DHCP server to assign IP addresses.	--upperip 10.0.0.254
--enable	Creates the DHCP server in the enable state; that is, it starts immediately after executing the <code>VBoxManage dhcpserver</code> command.	--enable

Those are the required parameters you need to include on the `VBoxManage dhcpserver` command each time you want to create a DHCP server for an internal network. In the above exercise, I used the 10.0.0.2 – 10.0.0.254 address range for assigning IP addresses to virtual machines because the DHCP server needs one IP address in the same range, so I reserved the 10.0.0.1 IP address for it.

You can use another IP address range like 192.168.1.1 through 192.168.1.254, for example. The trick is to use the same numbers for the first three parts of the IP address. Then you need to reserve just one IP address for the DHCP server, and you can have the rest of the range for your virtual machines.



Just remember, you can't use the last IP address in the range, 192.168.1.255 or 10.0.0.255, and so on.

One last thing: the `VBoxManage` command has a lot of uses besides starting DHCP servers. In later chapters, we'll see some other cases where you can use this powerful command. Now let me show you the last networking mode you can use in VirtualBox.

Have a go hero – testing the Internal Network mode on Windows guests

As in the other sections, don't forget to try out the exercises with Windows virtual machines, with other Linux distributions, and with an Open Solaris guest, too. You can also try to run two or more virtual machines at the same time with a different operating system on each one, to see if they can interact together without any strange behavior.

Pop quiz – using the internal networking mode

1. You need to isolate two virtual machines for testing purposes. What can you do?
 - a. Use the NAT networking mode.
 - b. Use the internal networking mode.
 - c. Use the bridged networking mode.
2. When using the internal networking mode, your virtual machine is:
 - a. Visible to other virtual machines on the same internal network.
 - b. Connected to the Internet.
 - c. Visible to other PCs on your LAN.
3. If Harry wants to access the web server on a virtual machine in your host from his brand-new MacBook Pro, you need to:
 - a. Use the bridged networking mode in your virtual machine.
 - b. Use the internal networking mode in your virtual machine.
 - c. Install VirtualBox on his MacBook Pro, and create an Ubuntu virtual machine.
4. You want to test a new web application but don't want to mess up your Windows XP main computer configuration by installing an Apache web server. What would be the best thing to do?
 - a. Buy another computer, and set it up as a web server.
 - b. Hire a hosting service provider like Godaddy.
 - c. Create a virtual machine, install the Apache web server on it, and configure it with the internal network mode to keep it isolated from your host PC and the other PCs in your LAN.

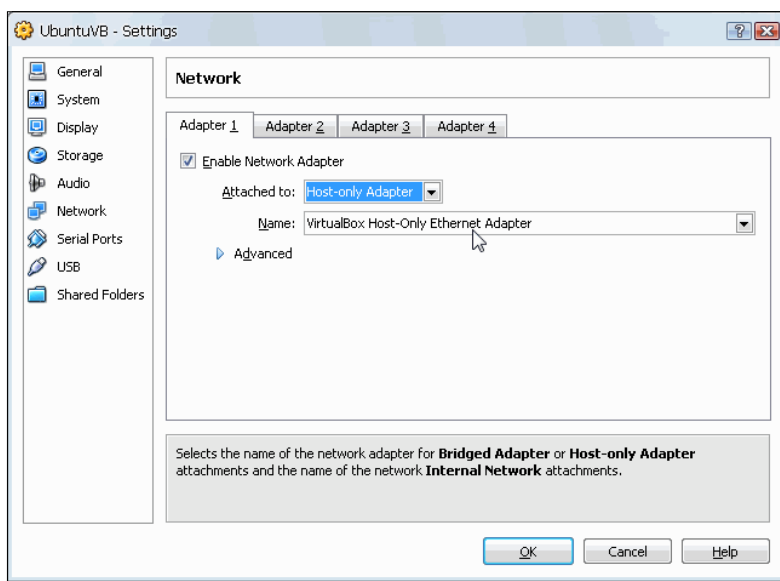
Using the Host-Only Networking mode

In this networking mode, your virtual machines can communicate between each other and with the host PC but not with other PCs on your LAN or with the Internet. It's almost the same as the internal network mode, only this time your virtual machines can talk to the host PC.

Time for action – communicating between VMs and your host PC only

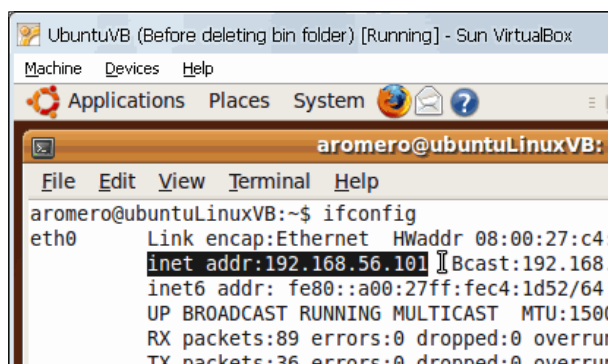
In this exercise, I'll show you how to use the host-only networking mode to allow communication between your `UbuntuVB` and `UbuntuVBClone` virtual machines and your host PC.

1. Select your `UbuntuVB` machine on the VirtualBox main screen, and click on the **Settings** button to go to the **UbuntuVB – Settings** page.
2. Go to the **Network** category, and then select the **Host-only Adapter** mode on the **Attached to** list box.
3. **VirtualBox Host-Only Ethernet Adapter** will be the selected option in the **Name** field, as shown below:



4. Click on **OK** to return to the VirtualBox main screen, start your **UbuntuVB** virtual machine, and login.

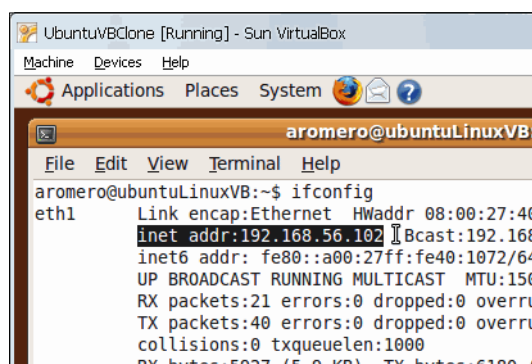
5. Open a terminal window (**Applications | Accessories | Terminal**), and type `ifconfig` followed by *Enter* to see your network configuration:



```

aromero@ubuntuLinuxVB:~$ ifconfig
eth0      Link encap:Ethernet  HWaddr 08:00:27:c4:
          inet addr:192.168.56.101  Bcast:192.168.
          inet6 addr: fe80::a00:27ff:fec4:1d52/64
          UP BROADCAST RUNNING MULTICAST  MTU:1500
          RX packets:89 errors:0 dropped:0 overru
          TX packets:36 errors:0 dropped:0 overru
  
```

6. In this case, the **eth0** interface has the inet address 192.168.56.101, thanks to the built-in DHCP server that's created automatically when you install VirtualBox.
7. Now go to the VirtualBox main screen, select the **UbuntuVBClone** virtual machine, and click on the **Settings** button to open the **UbuntuVBClone – Settings** window.
8. Go to the **Network** category, select the **Host-Only Adapter** mode on the **Attached to** list box, and click on **OK** to return to the VirtualBox main screen.
9. Start your **UbuntuVBClone** virtual machine, and login.
10. Open a terminal window (**Applications | Accessories | Terminal**), and type `ifconfig` followed by *Enter* to see your network configuration:

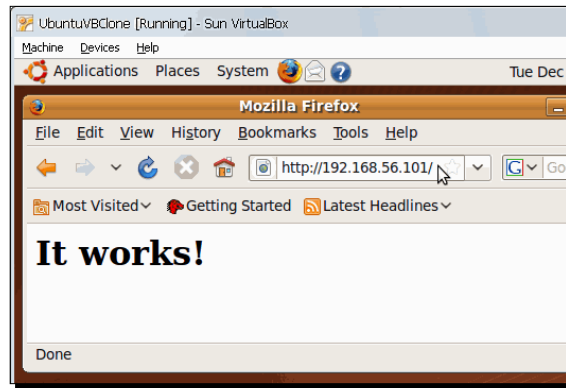


```

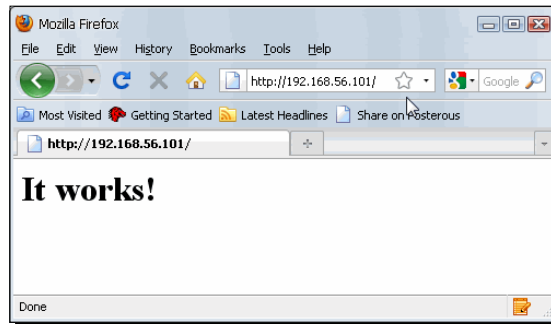
aromero@ubuntuLinuxVB:~$ ifconfig
eth1      Link encap:Ethernet  HWaddr 08:00:27:40:
          inet addr:192.168.56.102  Bcast:192.168
          inet6 addr: fe80::a00:27ff:fe40:1072/64
          UP BROADCAST RUNNING MULTICAST  MTU:1500
          RX packets:21 errors:0 dropped:0 overru
          TX packets:40 errors:0 dropped:0 overru
          collisions:0 txqueuelen:1000
          RX bytes:5027 (5.0 KB)  TX bytes:6100
  
```

11. Your **UbuntuVBClone** virtual machine will be assigned the 192.168.56.102 inet address through the same DHCP server that assigned the 192.168.56.101 IP address to your **UbuntuVB** virtual machine.

- 12.** Now open a web browser on your **UbuntuVBClone** virtual machine, and type `http://192.168.56.101` (remember to use the IP address assigned to your **UbuntuVB** virtual machine) to see if you can access the web server in **UbuntuVB**:



- 13.** Finally, open a web browser in your host PC, and type `http://192.168.56.101`—or the IP address assigned to your **UbuntuVB** virtual machine—to see if you can access the web server in **UbuntuVB**:



- 14.** The last screenshot indicates you can access your **UbuntuVB** virtual machine's web server from your host PC, too.
- 15.** You can close your virtual machines now.

What just happened?

You've just learned how to create a private network between your virtual machines and your host PC! This configuration is perfect for those times when you need to test some web application and when you don't want it to be publicly accessible yet. Then, when all's been tested and your web application works as expected, you can change the networking mode in your virtual machine to Bridged and make your web application accessible to everyone!

Have a go hero – testing the Host-Only mode on Windows guests

As in the other sections, don't forget to try out the exercises with Windows virtual machines, with other Linux distributions, and with an OpenSolaris guest, too. You can also try to run two or more virtual machines at the same time with a different operating system on each one, to see if they can interact together without any strange behavior.

Pop quiz – virtual networking

1. What is an IP address?
 - a. A piece of hardware in your host PC.
 - b. The number assigned to your virtual machine's network adapter so it can connect to the Internet and to other computers in a LAN in case it's using the bridged mode.
 - c. The number assigned to each hard drive in your virtual machine.
2. If you're running two virtual machines at the same time:
 - a. The internal networking mode lets them communicate between each other and your host PC.
 - b. The NAT mode lets them communicate between each other.
 - c. The Host-Only networking mode lets your virtual machines communicate between each other and your host PC.
3. What applies to both the Host-Only and internal networking modes?
 - a. Your virtual machines can't communicate between each other.
 - b. Your virtual machines can't communicate with your host PC and the other PCs in your LAN.
 - c. Your virtual machines can't connect to the Internet.

Have a go hero – combining networking modes in several VMs

Now that you've seen how to work with the different networking modes available in VirtualBox, it would be cool if you could enable more than one network adapter in a virtual machine and experiment with two or more networking modes at the same time. For example, open your `UbuntuVB` virtual machine **Settings** window, go to the **Network** category, and enable **Network Adapters 2** through **4**; then select a different networking mode (except the **Not Attached** mode) on each adapter, and see what happens!

This is the building block for creating more complex network settings, such as using a web server in one virtual machine, a database server on another virtual machine and connect them together in a private network through the internal network mode, and then enable a second network adapter on the web server virtual machine with the bridged mode to let incoming connections from a LAN or the Internet, for example. Experiment all you want with the different networking modes until you get the hang of it, and then try using two or more virtual machines at the same time if you have the RAM to spare!

Summary

VirtualBox has several networking modes you can choose for your virtual machines, and each one serves a specific purpose. In this chapter, we covered the five networking modes available using several concise exercises to show you the differences and specific features of each networking mode and how you can choose the best networking mode in any given situation:

1. **Network Address Translation (NAT) mode:** This is the default networking mode when you create a virtual machine. It's like having your virtual machine behind a firewall: you can surf the web, download files, read your email, use instant messaging, and do all the things a regular web user does. You can even run a web server via the port forwarding feature, but due to several limitations, it's not recommended for a production server.
2. **Bridged Networking mode:** This is the mode you need for advanced networking, like running a web/FTP server or any other kind of dedicated server. Your virtual machine acts just like any other real PC in your LAN.
3. **'Not Attached' mode:** This mode makes your virtual machine think that there's no cable connected to its network adapter; this mode is for those times when you want your virtual machine to be completely isolated from your host PC, the Internet, other PC's in your LAN, and even from other virtual machines.
4. **Internal-Networking mode:** This mode allows for your virtual machines to communicate between each other, but you can't access them from your host PC or from another PC on your LAN! And they are isolated from the Internet, too!

5. **Host-Only Networking mode:** This mode shows that your virtual machines can communicate between each other and with the host PC but not with other PCs on your LAN or to the Internet. It's almost the same as the internal networking mode, only this time your virtual machines can talk to the host PC.

Specifically, we covered:

- How to connect a virtual machine to the Internet and a local LAN through the default NAT mode
- The differences between using the NAT mode and the bridged networking mode
- How to use a virtual machine as a web server and what would be the best networking mode for it
- How to use the internal networking mode to isolate one or several virtual machines in a private LAN, out of reach from the host computer, from other LANs, or from the Internet
- How to use the host-only networking mode to restrict outside access for certain virtual machines

Now that you have a good idea of how to configure your virtual machine's networking settings depending on the environment in which you plan to use them, let's go to Chapter 7 so you can start working with virtual appliances virtual machines that come pre-configured with all the necessary software applications to serve a specific purpose (including the operating system) such as a web development server, a file server, a database server, and so on.

7

Using Virtual Appliances

During the last six chapters, you saw how to create virtual machines from scratch. You also learned to work with them as if they were real computers. This process is quite time consuming, so fortunately there are several options where you can get virtual machines ready to run with some of the most popular operating systems and software applications pre-installed, and you need to just configure your virtual machines to suit your particular environment.

In this chapter you shall:

- ◆ Learn what virtual appliances are and how you can use them to reduce the installation, configuration, and maintenance times associated with creating virtual machines from scratch
- ◆ Learn how to run virtual appliances on VirtualBox
- ◆ Learn how to import and export virtual machines
- ◆ Learn how to install and use virtual appliances from several providers such as VirtualBox® Images, BitNami, and Turnkey Linux

Setting up preconfigured virtual machines in a flash

Up until now we've been working with the `UbuntuVB` and `UbuntuVBClone` virtual machines we created from scratch in previous chapters. This is the most common way of working with VirtualBox, and it's like having several full-fledged PCs to work with, only they're all inside your host!

The only drawback is that, as with any other regular PC, you need to install an operating system and then the software applications you want to work with, and that can take a lot of time and effort. But fortunately for us, there are several communities around the world that work very hard to offer **virtual appliances** to people who need them.

And what is a virtual appliance, you may ask? Well, to keep it simple, let's just say that a virtual appliance is a virtual machine that you can download, configure, and use without having to install the operating system first and then the software applications you need to run.

For example, imagine that you need a virtual machine with Apache, MySQL, PHP, and Linux (a collection of software applications usually known as **LAMP** in the world of programmers) to work on your new website. You'd have to open VirtualBox, create a new virtual machine, install Ubuntu or some other Linux distribution, wait for a half hour or so until Linux gets completely installed in your VM, and then proceed to install and configure Apache, PHP, and MySQL. Not a simple chore, right?

And what would you think if I told you there's a virtual appliance from Turnkey Linux called LAMP Stack Appliance (<http://www.turnkeylinux.org/lamp>) that can be downloaded for free, and it's ready to run on VirtualBox? Once you download it, it will only take you about 20 minutes or so to create a new virtual machine and install this LAMP appliance! And the best of all, you won't have to mess up your main PC with an Apache, PHP, and MySQL installation.

There are a lot of virtual appliances available on Turnkey Linux and on several other websites such as BitNami (<http://www.bitnami.org>), VirtualBox Images (<http://virtualbox.wordpress.com>), VirtualBoxImages.com (<http://www.virtualboximages.com>), and so on. Stay with me, and I'll teach you how to get these fabulous virtual appliances and create your own virtual machines with Wordpress, Joomla, Apache, Java, and all the most popular open source software you can get your hands on! And as I said before, you won't have to bother with all the clumsy details of installing operating systems or any of the software you'll be running in your virtual machines! Now let's get started...

Importing a virtual appliance

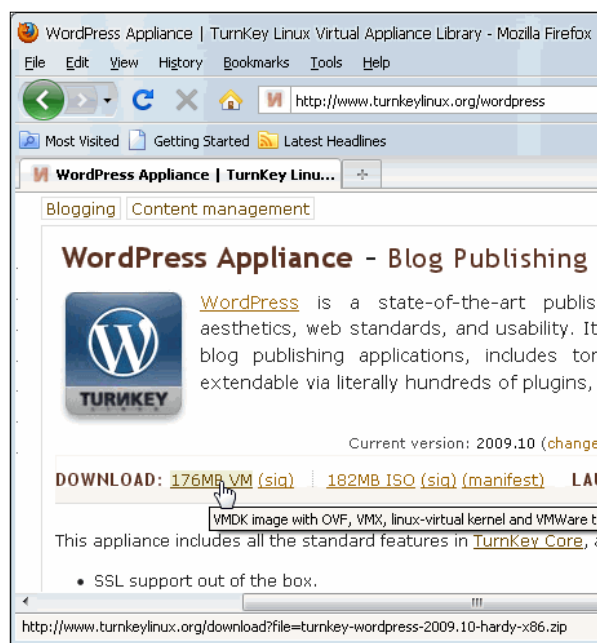
One quick way of getting your hands on a virtual appliance is by means of importing it in your VirtualBox installation. At the present time, there are several sites where you can download virtual appliances and import them into VirtualBox. I chose the Turnkey Linux website (<http://www.turnkeylinux.org>) for the following exercise because it offers excellent virtual appliances ready to run as standalone servers or inside a virtual machine, and they're almost too good to be free!

To import or export a virtual appliance in VirtualBox, you must use the **Open Virtualization Format (OVF)** standard. This means you can import/export virtual appliances from every virtualization product that supports the **OVF** standard, like VMware. In short, not only can VirtualBox work with its own VDI file format, but it also can work with the extremely popular VMDK file format used on VMware and with the VHD format used on the Microsoft's Virtual PC product, too!

Time for action – using the TurnKey Wordpress virtual appliance

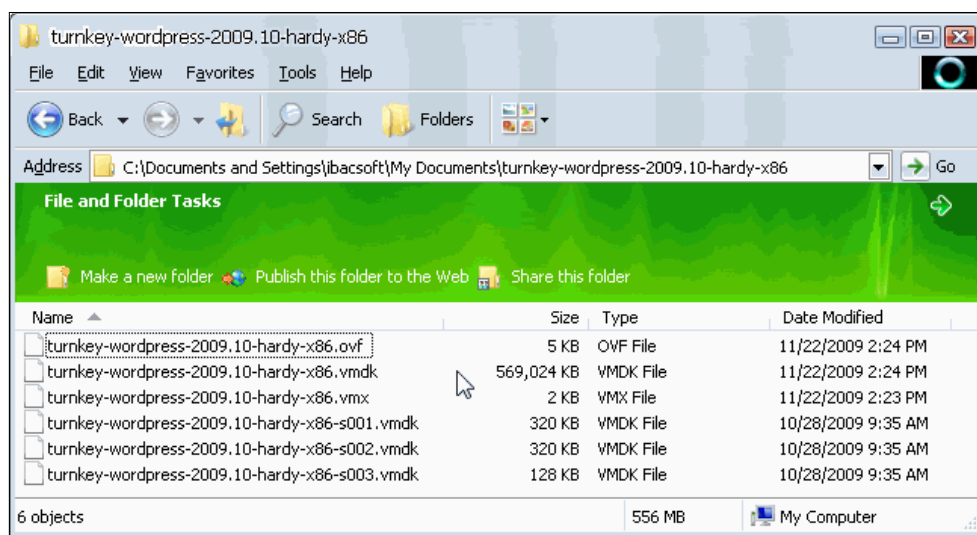
In this exercise, I'll show you how to download, import, and use the TurnKey Wordpress virtual appliance in your own virtual machine.

1. Open a web browser window, go to <http://www.turnkeylinux.org/wordpress>, and scroll down until you locate the **VM** link. Click on it:

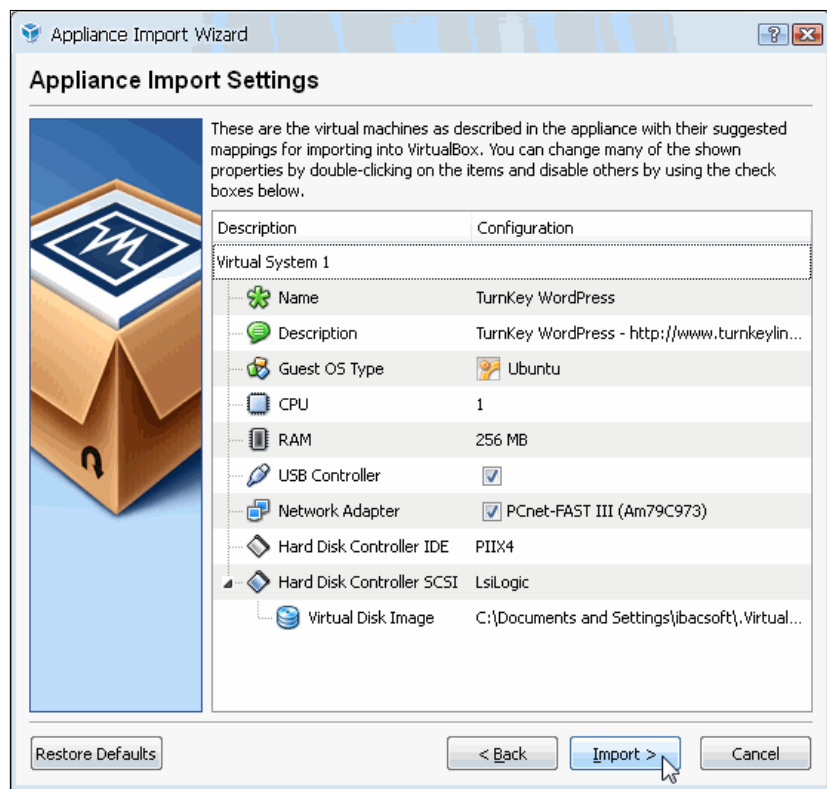


2. After a few seconds, you will be taken to the Sourceforge website, and a file download dialog will show up. Select the **Save** option to save the Wordpress virtual appliance file to your computer, and wait for it to download through your web browser.

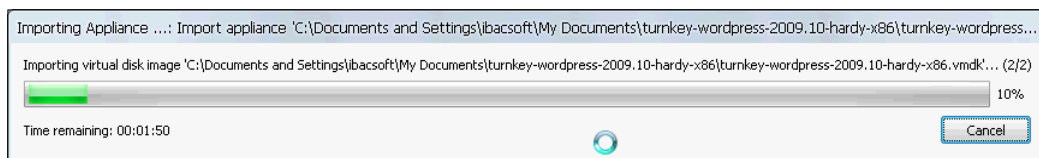
3. Once the download completes, go to the directory where you downloaded the Wordpress file, and unzip it. The following screenshot shows the file's contents after unzipping it:



4. As you can see, the Wordpress virtual appliance consists of six files: one .ovf file, several .vmdk files, and one .vmx file. Now open VirtualBox to continue.
5. Once the VirtualBox main screen shows up, select **File | Import Appliance** from the main menu.
6. The **Appliance Import Wizard** will appear next. Click on the **Choose...** button to continue.
7. Now the **Select an appliance to import** dialog will show up. Navigate to the directory where you unzipped the Wordpress virtual appliance, select the .ovf file, and click on **Open** to continue.
8. You will be taken back to the **Appliance Import Wizard**. Click on **Next** to continue.
9. The **Appliance Import Settings** screen will appear to show the predefined settings for your new Wordpress virtual machine. Click on **Import>** to continue:

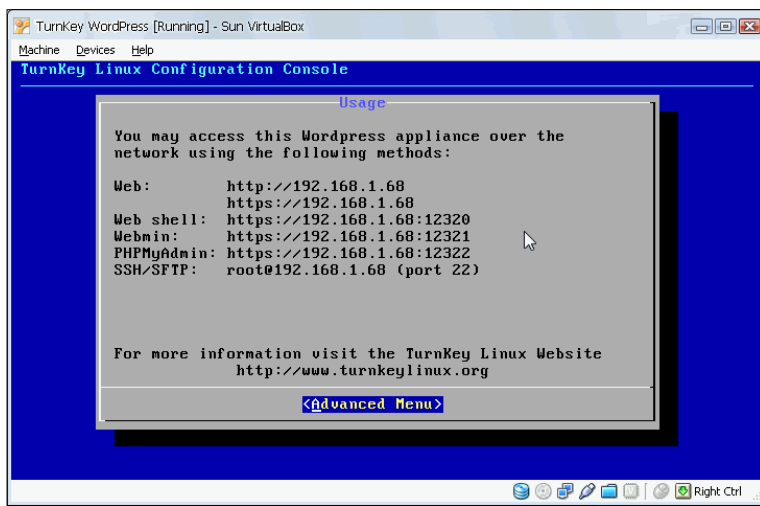


- 10.** The **Importing Appliance** dialog will appear to indicate that VirtualBox is importing the Wordpress virtual appliance in your new virtual machine:

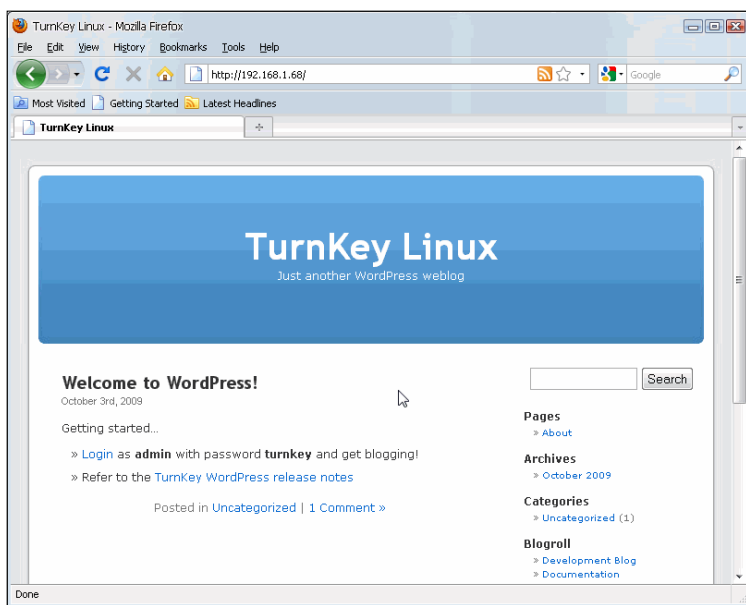


- 11.** Wait until the wizard finishes importing your new virtual machine. Once the importing process is finished, you'll be taken back to the VirtualBox main screen. This time your new **Turnkey WordPress** virtual machine will show up. Click on the **Start** button to start it up.

- 12.** Wait until your new virtual machine finishes booting up. When your machine is ready, the following screen will show up:



- 13.** To access your new Wordpress virtual appliance, open a web browser window on your host PC, and type the URL shown in the **Web:** line of your TurnKey Wordpress window (in the above example it's `http://192.168.1.68`, but your IP address may be different). The TurnKey Linux Wordpress homepage should appear next:



- 14.** Now your Turnkey Linux Wordpress virtual appliance is ready to roll! Don't shut it down yet because you'll need to have it running for the next exercise.

What just happened?

How about that? You've just installed and configured a Wordpress blog server in just a few steps! And without even having to worry about installing the Apache web server, the MySQL database server, and the Linux operating system! That's the real beauty of virtual appliances: they're ready to run once you import them into a blank virtual machine!

In this particular example, we created a Wordpress virtual machine by downloading and importing the Turnkey Linux Wordpress virtual appliance. The virtual appliance consists of six files, as we saw before:

File(s)	Definition
turnkey-wordpress-2009.10-hardy-x86.ovf	The OVF (Open Virtualization Format) file contains an XML description of the virtual machine.
turnkey-wordpress-2009.10-hardy-x86.vmdk	The VMDK (Virtual Machine Disk) files contain the hard disk image of your virtual appliance.
turnkey-wordpress-2009.10-hardy-x86-s001.vmdk	
turnkey-wordpress-2009.10-hardy-x86-s002.vmdk	
turnkey-wordpress-2009.10-hardy-x86-s003.vmdk	The configuration file used on the VMware virtualization software, VirtualBox uses the OVF configuration file instead.
turnkey-wordpress-2009.10-hardy-x86.vmx	

All Turnkey Linux virtual appliances conform to the OVF standard. They contain an `.ovf` file and at least one `.vmdk` file because they were created using VMware, but you can also use them in VirtualBox. Virtual appliances created in VirtualBox contain an `.ovf` file and a `.vdi` file because VDI is the default format VirtualBox uses for hard disk images. All in all, the basic importing process is the same: the virtual appliance you want to import must have an `.ovf` file.

In step 9 of the previous exercise you saw the **Appliance Import Settings** screen. In this screen, you can change the virtual machine settings to suit your own needs by double-clicking on the description items. For example, if you want to assign more RAM to your new virtual appliance, just double-click on the RAM item, and select the new memory amount. I always try to run the virtual appliance without any changes for the first time because later on, I can always go to the **Settings** page of my virtual machine and change whatever setting I need to change.

Now that you know how to import a virtual appliance, let me show you the opposite process: exporting virtual appliances to use them on other hosts.



When importing a virtual appliance with a Windows guest operating system, you may experience a **Blue Screen of Death (BSOD)** error after starting it from VirtualBox. The most probable cause is having a different virtual disk controller type than the one the virtual appliance was created with. To solve this problem, you can change the disk controller type in the **Storage** category of your imported virtual machine settings window.

Pop quiz – importing virtual appliances

1. What file do you need to open to import a virtual appliance in VirtualBox?
 - a. A .doc file.
 - b. A .vmdk file.
 - c. An .ovf file.
2. A virtual appliance is
 - a. A very powerful host PC.
 - b. A very powerful vacuum cleaner.
 - c. A preconfigured virtual machine.
3. How can you import a virtual appliance into VirtualBox?
 - a. You need to create another extra virtual hard disk.
 - b. You need to create a new virtual machine, and then adjust the RAM setting.
 - c. Using the Appliance Import Wizard.

Exporting a virtual appliance

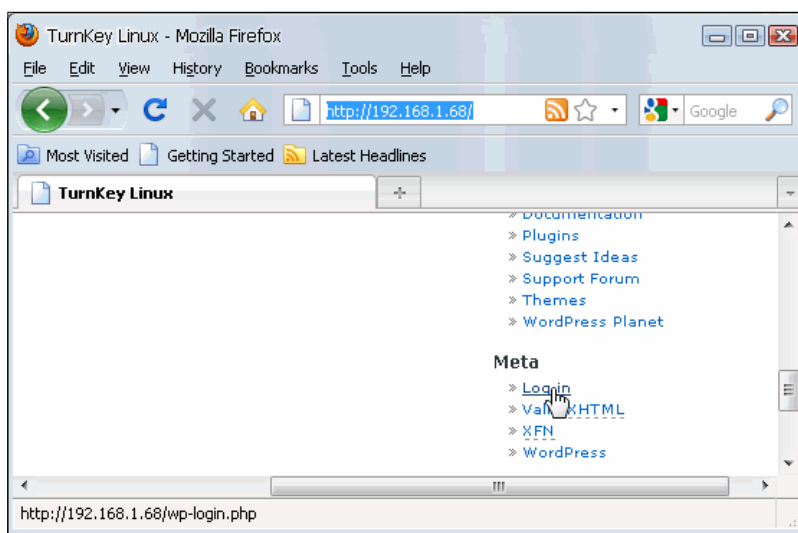
Previously, you imported a Wordpress virtual appliance with the default options. In this section, I'll show you how to customize your Wordpress blog to your likings and export it as a customized virtual appliance so that you can run it on other hosts.

This is a very easy way to distribute ready-to-use software packages: you configure a virtual machine with all the desired software—including the operating system—and then distribute it throughout a group of users to save time and labor costs.

Time for action – exporting your customized Wordpress virtual appliance

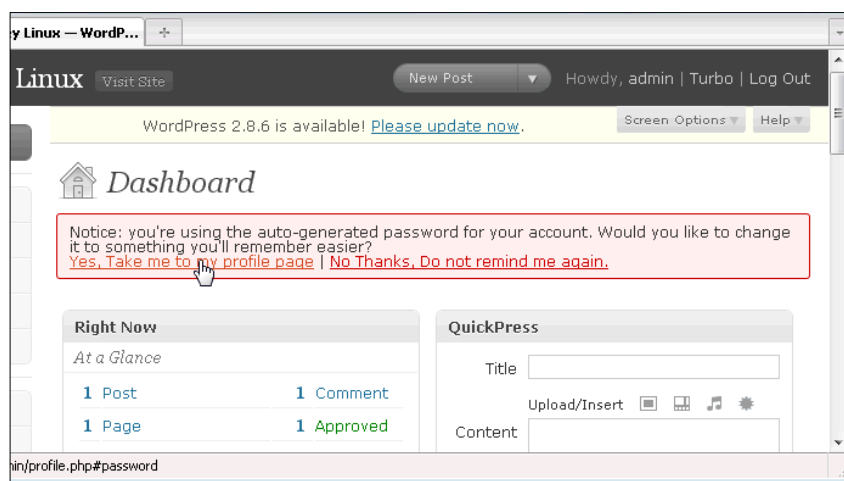
Now it's time to customize your Wordpress virtual machine and export it as an OVF virtual appliance so that you can use it on other hosts that have VirtualBox installed.

1. Go to your Wordpress blog home page, scroll down until you locate the **Meta** section at the bottom-right part of the screen, and click on the **Login** link to log into your Wordpress blog:

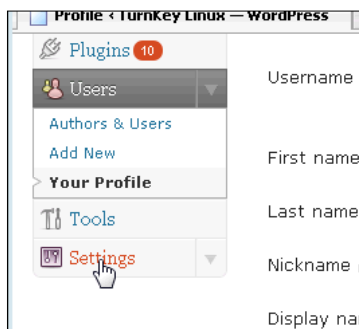


2. Type **admin** in the **Username** field and **turnkey** in the **Password** field. Then click on the **Login** button to continue.

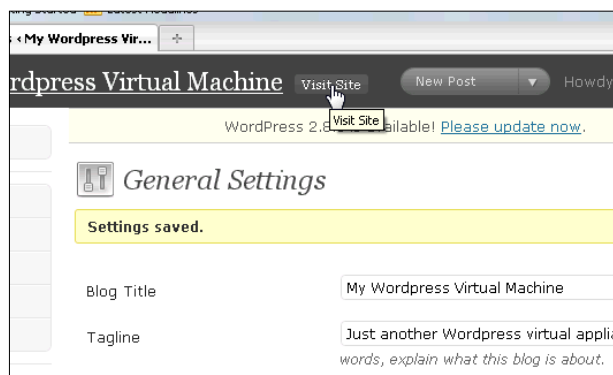
3. The **Wordpress Dashboard** page will appear next. The first thing you need to do is change your account's password because all Turnkey Linux Wordpress appliances have the same auto-generated password. The first time you log into Wordpress, a **Notice** box appears to warn you about changing your password. Select the **Yes, Take me to the profile page** link to continue:



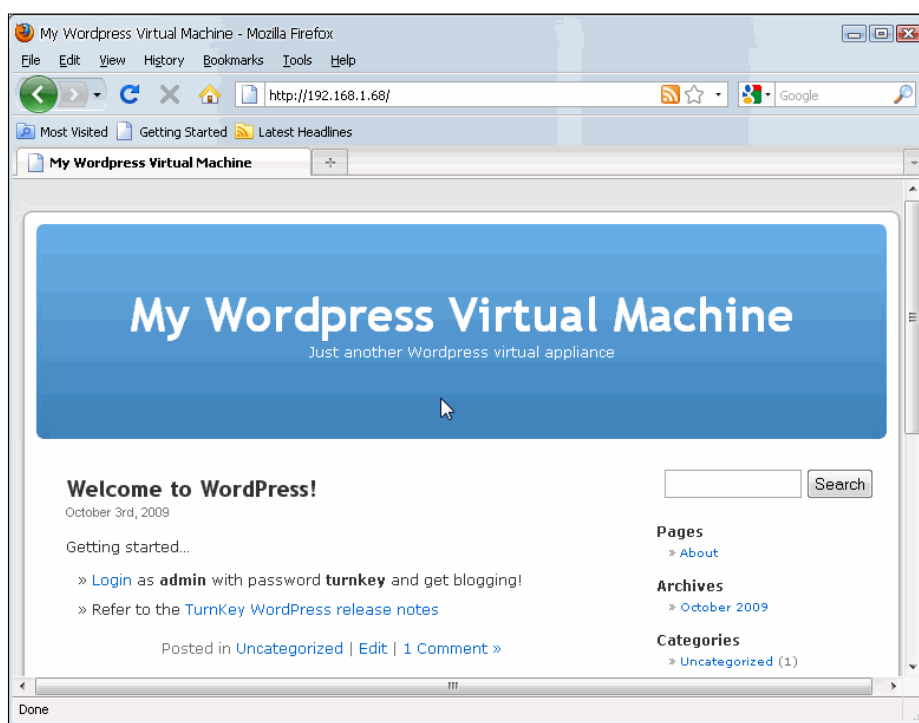
4. Next, you'll be taken to your profile page. Scroll down until you locate the **New password** section, type your new password on the two blank fields, and then click on **Update Profile**.
5. Wordpress will update your user profile. Now scroll up until you locate the **Settings** section at the left part of the screen, and click on it:



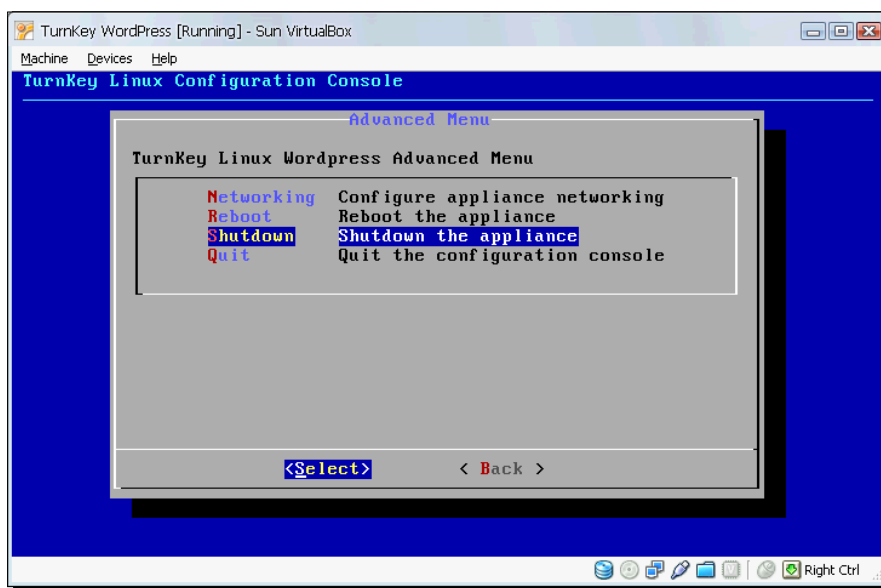
6. You'll be taken to the **General Settings** page next. Type My Wordpress Virtual Machine in the **Blog Title** field and Just another Wordpress virtual appliance in the **Tagline** field. Type your email in the **E-mail address** field, select the correct time zone to reflect your location, and click on the **Save Changes** button when finished. Once your settings are saved, click on the **Visit Site** link at the uppermost part of the settings page to continue:



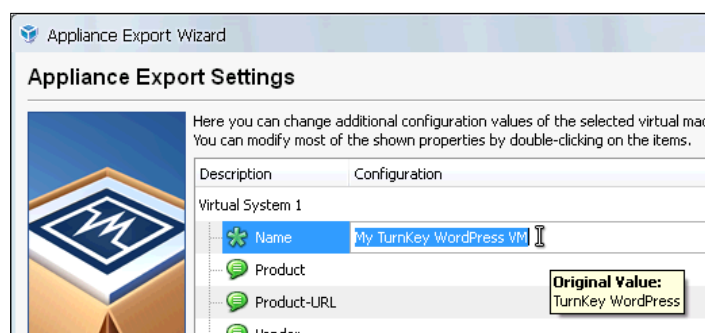
7. Your new Wordpress home page will show up with the changes you made to it:



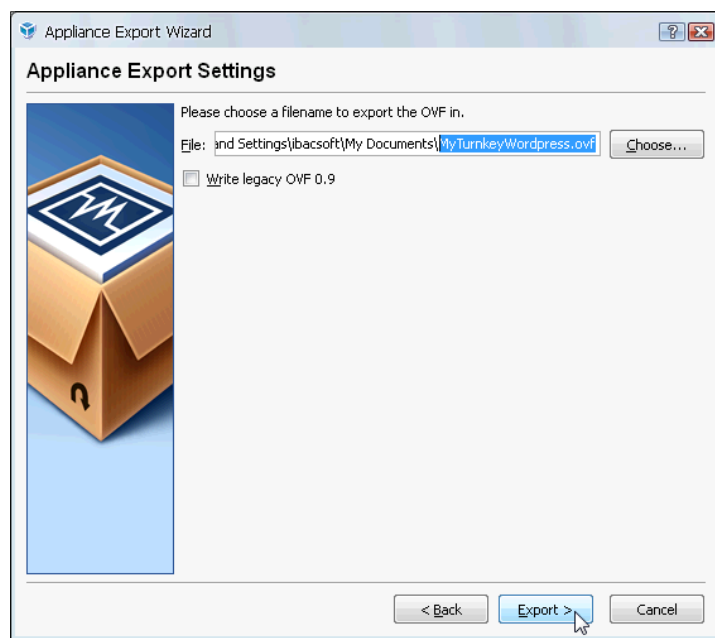
8. Now close your web browser window, and select your Wordpress virtual appliance screen. Hit *Enter* to select the **Advanced Menu** option, then select the **Shutdown** option on the **TurnKey Linux Wordpress Advanced Menu** screen, and hit *Enter* again to shut down your virtual appliance gracefully:



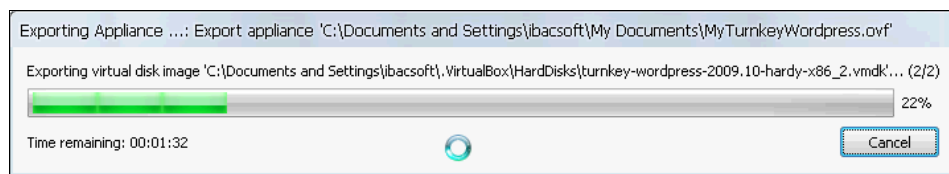
9. Once your virtual machine finishes shutting down, go to the VirtualBox main screen, and select **File | Export Appliance** from the menu.
10. The **Welcome to the Appliance Export Wizard** will appear next. Select the **TurnKey Wordpress** appliance, and click on **Next** to continue.
11. The **Appliance Export Settings** will show up next. Double click on the **Name** field, and type **My Turnkey Wordpress VM** to replace the old value:



- 12.** Click on **Next** to continue. Replace the default value on the **File** field with `MyTurnkeyWordpress.ovf`, and click on **Export** to continue:



- 13.** The **Appliance Export Wizard** will start exporting your customized virtual appliance, and the **Exporting Appliance** dialog will appear to show a progress bar:



- 14.** When finished, the **Appliance Export Wizard** will take you back to the VirtualBox main screen. Go to the directory where you exported your virtual machine, and you'll see two files: `turnkey-wordpress-2009.10-hardy-x86_2.vmdk` and `MyTurnkeyWordpress.ovf`. These are the files you'll need when importing your Wordpress customized virtual appliances to another host PC with VirtualBox.

What just happened?

Pretty neat, huh? You can customize a virtual machine to your likings, and if later you need to replace your host PC with another one, you need only to backup all your virtual appliances, install VirtualBox on your new host PC, and then import your virtual appliances to your new setup! This is another excellent feature of virtual machines: **portability**.

Because the Appliance Export Wizard does all the dirty work for you, it's really easy to create and tune up a virtual machine on your host PC, export it, and then distribute it to other users in your home, office, or school environment. That's why virtual appliances are becoming so popular nowadays: you can concentrate on using the software application you need without having to spend part of your precious time learning how to configure a computer and all the required software just to run it.



If you're in a hurry, most of the time you can save time and effort if you clone the virtual hard disk of a virtual machine with the `VBoxManage clonehd` command and create a new virtual machine with the same settings, instead of going through the lengthy process of exporting/importing a virtual appliance. However, if you want to distribute your virtual machine to other users so that they can import your virtual appliance without having to do anything extra, you need to stick with the exporting/importing process.

Have a go hero – testing your new customized virtual appliance

Now that you've created your first customized virtual appliance, compress the `.ovf` and `.vdi` files into a `zip` file, copy this file to another PC with VirtualBox, and try to import your customized virtual appliance into it. If you don't have access to another PC where you can install VirtualBox and make this test, don't worry. Just delete the Turnkey Linux Wordpress virtual machine from your VirtualBox installation, unzip the file with your customized virtual appliance, and try to import it back again. And remember, if you run into any trouble, you can always reach me at questions@packtpub.com.

Have a go hero – exporting your UbuntuVB virtual machine as a virtual appliance

How about exporting the **UbuntuVB** virtual machine we've been working with throughout the book? You can then distribute it to a friend or colleague on another host PC. Try to import it on a different host operating system than the one you're using to see if you can spot any differences in your UbuntuVB virtual appliance.

Have a go hero – importing and exporting more virtual appliances

You already know how to import and export the Turnkey Linux Wordpress virtual appliance, right? Well, it would be cool if you could also practice with the other virtual appliances available from Turnkey Linux at <http://www.turnkeylinux.org>. All of the virtual appliances are based on Ubuntu Linux 8.04 Server Edition and some of the best open source software applications such as Apache, PHP, Tomcat, and MySQL.

And how about creating a virtual appliance from scratch? You can use Ubuntu 8.04 Hardy Heron or OpenSolaris to create a virtual machine, customize it to your likings, and then export it as a virtual appliance!

Pop quiz – exporting virtual appliances

1. You need to train ten people from the sales team so that they can use Wordpress to promote your company's products. You can:
 - a. Let them experiment with your company's Wordpress live blog.
 - b. Install VirtualBox on their PCs and let them import a customized Wordpress virtual appliance so that you can teach them how to use it before starting to post on the company's live blog.
 - c. Quit your job and join the Salvation Army.
2. If you need to change your host PC and don't want to lose all the work you've been doing in your **UbuntuVB** virtual machine, you can:
 - a. Install VirtualBox on the new host PC, and then create a new Ubuntu virtual machine from scratch.
 - b. Export your UbuntuVB virtual machine as a virtual appliance, then install VirtualBox on your new host PC, and import your UbuntuVB virtual machine.
 - c. Install Ubuntu on your new host PC.
3. The Appliance Export Wizard lets you:
 - a. Add virtual hard disks to your virtual machines.
 - b. Export your virtual machines to use them on other host PCs.
 - c. Change the networking mode of your virtual machine.

Working with virtual appliances

In the previous section, you learned how to import/export virtual appliances to your host PC running VirtualBox. In this section, I'll show you how to work with virtual appliances from three very popular sources: VirtualBox® Images, BitNami, and Turnkey Linux.

Before we roll up our sleeves, I'd like to say something about virtual appliances. Sometimes a virtual appliance can consist of an operating system ready to run, like the PuppyLinux virtual appliance we'll see next, and sometimes a virtual appliance can consist of an operating system and several open source software applications like the BitNami Drupal and the Turnkey Linux File Server virtual appliances we'll see later.

What I want you to understand is that a virtual appliance is meant to ease up the task of using an operating system or software application; for example, the PuppyLinux virtual appliance you're about to see lets you run the PuppyLinux operating system in just a few minutes, without your having to worry about all the details involved with the installation process.

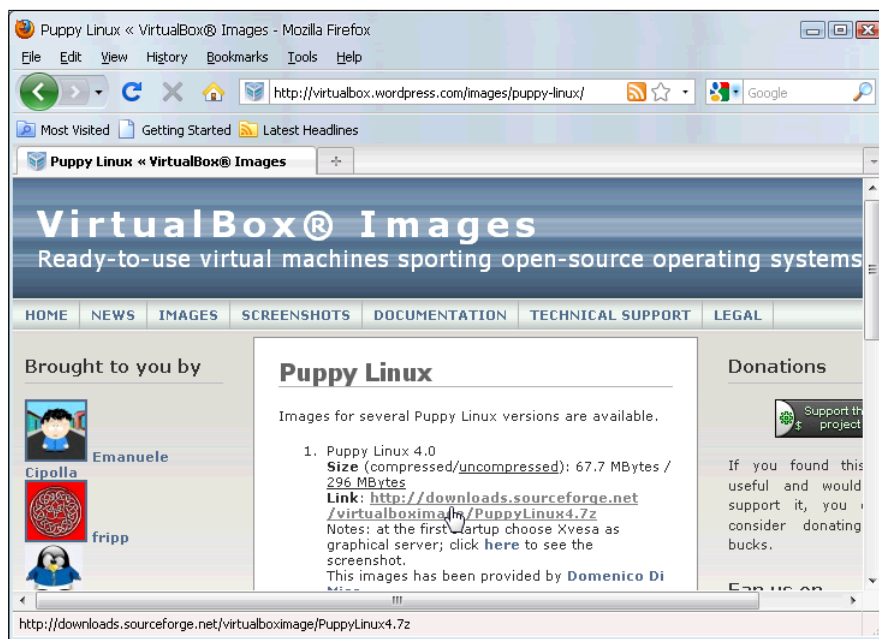
Using virtual images from VirtualBox® Images

The VirtualBox® Images website provides virtual images that use Linux, OpenSolaris, and other popular open source or free operating systems. In this case, the VirtualBox Image contains a virtual appliance because the operating system and all the software packages needed are ready to run! You just need to adjust the basic settings of your virtual machine and voila! To better understand this concept, let's do an exercise where you'll download, install, and configure the PuppyLinux virtual image from the VirtualBox Images website (<http://virtualbox.wordpress.com>).

Time for action – using a PuppyLinux VM in VirtualBox

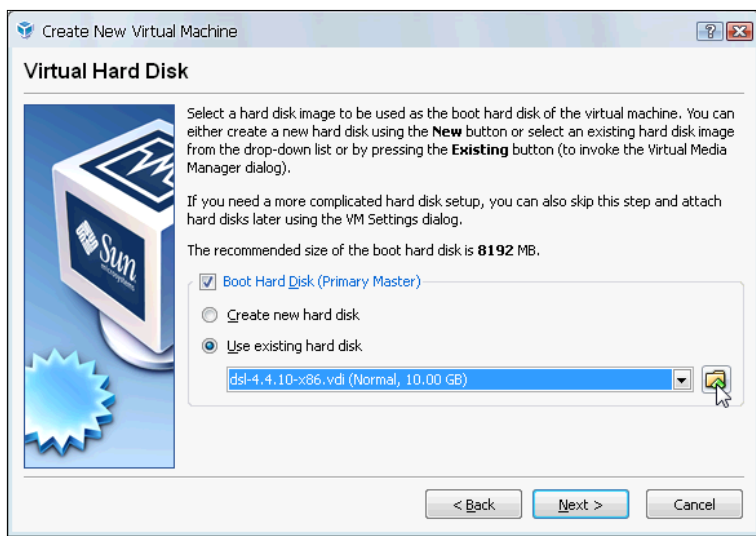
In this exercise, you'll download the PuppyLinux virtual image, and then create a virtual machine in VirtualBox to use the PuppyLinux operating system.

1. Open a web browser window in your host PC, go to <http://virtualbox.wordpress.com/images/puppy-linux>, and click on the **Puppy Linux 4.0** download link:

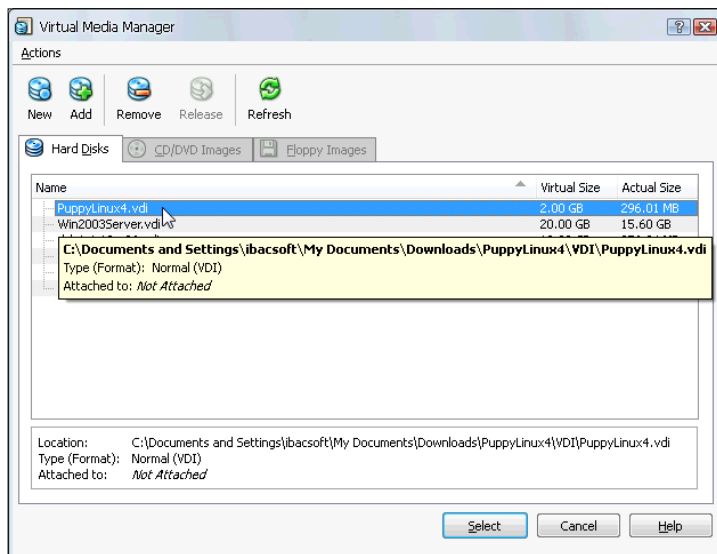


2. You'll be taken to the Sourceforge.net website. Wait for the open/download dialog box to appear, and save the `PuppyLinux4.7z` file to your hard drive.
3. Once the file finishes downloading, go to the directory where you saved it, and extract its contents. You can use the 7-Zip free application to uncompress the zipped file (<http://www.7-zip.org>). The `PuppyLinux4.7z` contains an image file named `PuppyLinux4.vdi`. You'll find this file inside the `VDI` directory.
4. Open VirtualBox, and create a new virtual machine named `PuppyLinuxVB`. When the **VM Name and OS Type** screen shows up, select **Linux** and **Linux 2.6** in the **Version** and **Operating System** fields, respectively, and click on **Next** to continue.

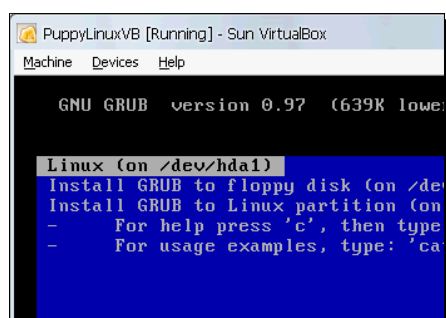
5. Leave the default value in the **Memory** screen, and click on **Next** to continue. The **Virtual Hard Disk** screen will appear next. Select the **Use existing hard disk** radio button, and click on the **Virtual File Manager** icon to continue:



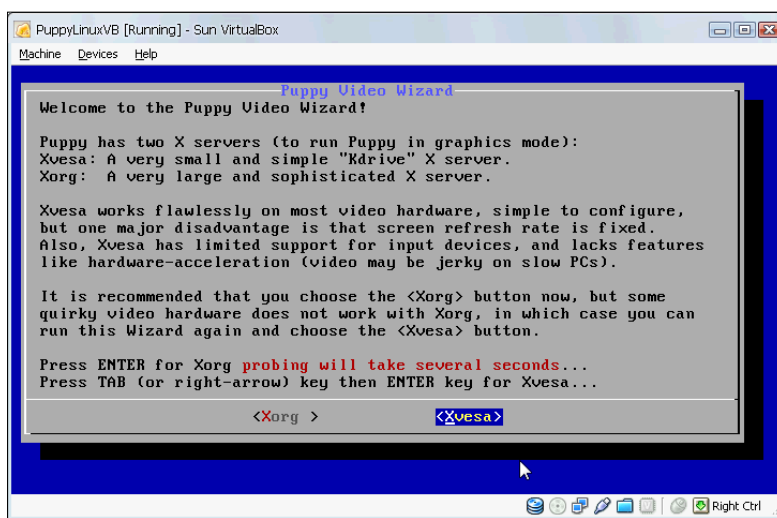
6. The **Virtual Media Manager** window will show up. Click on the **Add** button to open the **Select a hard disk image file** dialog, navigate to the directory where the `PuppyLinux4.vdi` file resides, and click on **Open** to add it to the Virtual Media Manager list, as shown below:



7. Select the `PuppyLinux4.vdi` image, and click on the **Select** button to select this image file as the hard disk for your new `PuppyLinuxVB` virtual machine. You'll return to the **Create New Virtual Machine** wizard. Click on **Next** to continue.
8. The wizard will show you a summary of all the options you selected. Click on **Finish** to continue and return to the VirtualBox main screen.
9. Make sure your `PuppyLinuxVB` virtual machine is selected, and click on the **Start** button to start it up. The **GNU GRUB** screen will appear next. Make sure the **Linux (on /dev/hda1)** option is selected, and hit *Enter* to continue:



10. The PuppyLinux operating system will start to boot up. Eventually, the **Select the keyboard layout** screen will show up. Hit *Enter* to leave the default US keyboard value and continue.
11. The **Puppy Video Wizard** screen will appear next. Select the **Xvesa** option, and hit *Enter* to continue:





There have been some issues reported when choosing the Xorg server with VirtualBox, and the system apparently hangs up. That's why I chose the Xvesa server in this exercise instead. However, I found a possible solution in the PuppyLinux forum: <http://208.109.22.214/puppy/viewtopic.php?p=380019&sid=f76ba083439023ba80d8e5485d8abf19>.

- 12.** After a few seconds, the **Xvesa Video Wizard** screen will show up. Select the **800x600x24** screen resolution, and click on **CHANGE** to continue.
- 13.** The Xvesa Video Wizard will change your virtual machine's resolution. If everything looks okay, click on the **OKAY** button to save your changes. Now you have a PuppyLinux virtual appliance ready to roll!

What just happened?

In this exercise, we saw how to download and configure a virtual appliance from the <http://virtualbox.wordpress.com> website, where you'll find virtual images from all the popular Linux distributions and other open source operating systems, like OpenSolaris. The PuppyLinux virtual image comes ready to use since you need only to configure your virtual machine's basic settings, such as memory and operating system type. Instead of creating a new hard disk for your VM, you use the VDI image downloaded from virtualbox.wordpress.com. There are a lot of ready-to-run Linux distributions available on this website.

Have a go hero – installing Guest Additions on PuppyLinux

Now that you have a PuppyLinux virtual machine running, it would be cool to install the VirtualBox Guest Additions on it to enjoy all the advantages such as file sharing, mouse and keyboard integration, and so on. Just open the SeaMonkey web browser in your PuppyLinux virtual machine, and type <http://www.murga-linux.com/puppy/viewtopic.php?mode=attach&id=11689&sid=ef8f142e6dd1f77174a712e74b5bca68> on the address bar. Choose the **Open it with the default application (petget)** option; then click on **OK**, and the **PETget package manager** window will show up next. Click on **INSTALL PACKAGE** to install the Guest Additions package in your PuppyLinux virtual machine. Now just wait until the **PETget package manager** shows a dialog to indicate the Guest Additions were installed successfully. Click on the **OKAY** button of this dialog, close all windows on your PuppyLinux virtual machine, and reboot it. When your PuppyLinux virtual machine finishes booting up, you'll be able to use all the Guest Additions features!

Using virtual appliances from BitNami

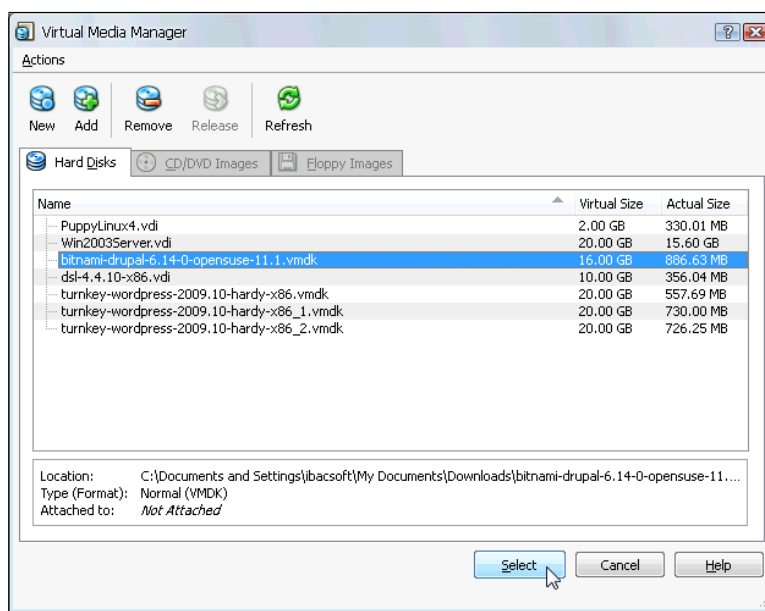
This is another cool place where you can get excellent virtual appliances for your specific needs. There are a lot of cool web applications you can download and start using almost immediately in your virtual machines such as the Drupal Content Management System, the Apache Roller Blog Server, the Moodle E-learning System, and the phpBB forum to name a few. You can check out all the virtual appliances available at <http://www.bitnami.org>.

Time for action – using the BitNami Drupal virtual appliance

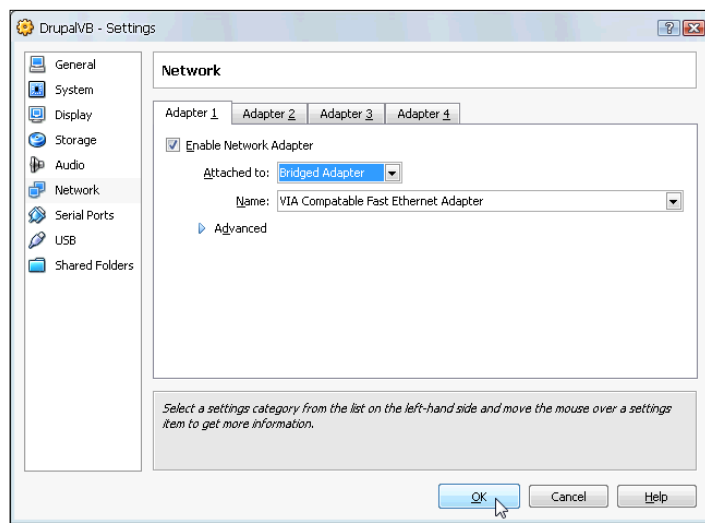
In this exercise, you'll learn how to download, install, and use the Drupal virtual appliance from <http://www.bitnami.org> in your VirtualBox host.

1. Open a web browser window on your host, go to <http://www.bitnami.org/stack/drupal>, scroll down until you locate the **VM** section, and click on the **Download** link.
2. Wait for the open/download dialog box to appear, and save the Drupal VM file (`bitnami-drupal-6.14-0-opensuse-11.1.zip` at the time of this writing) to your hard drive.
3. Once the file finishes downloading, go to the directory where you saved it, and extract its contents. The `.zip` file contains three files: `bitnami-drupal-6.X-X-opensuse-11.X.vmdk`, `bitnami-drupal-6.X-0-opensuse-11.X.vmx`, and `README.txt`. To create a Drupal virtual appliance in VirtualBox, you'll only need the `.vmdk` file.
4. Open VirtualBox, and click on the **Start** button to create a new virtual machine. Type `DrupalVB` in the **Name** field, select `Linux` as the **Operating System** and `OpenSUSE` as the **Version** on the **VM Name and OS Type** screen, and then click on **Next** to continue.
5. Leave the default value in the **Memory** screen, and click on **Next** to continue. On the **Virtual Hard Disk** screen, select the **Use existing hard disk** option, and click on the **Virtual Media Manager** button to open the **Virtual Media Manager** screen.
6. Then click on the **Add** button to open the **Select a hard disk image file** dialog, and navigate to the directory where you extracted the Drupal `.vmdk` file.

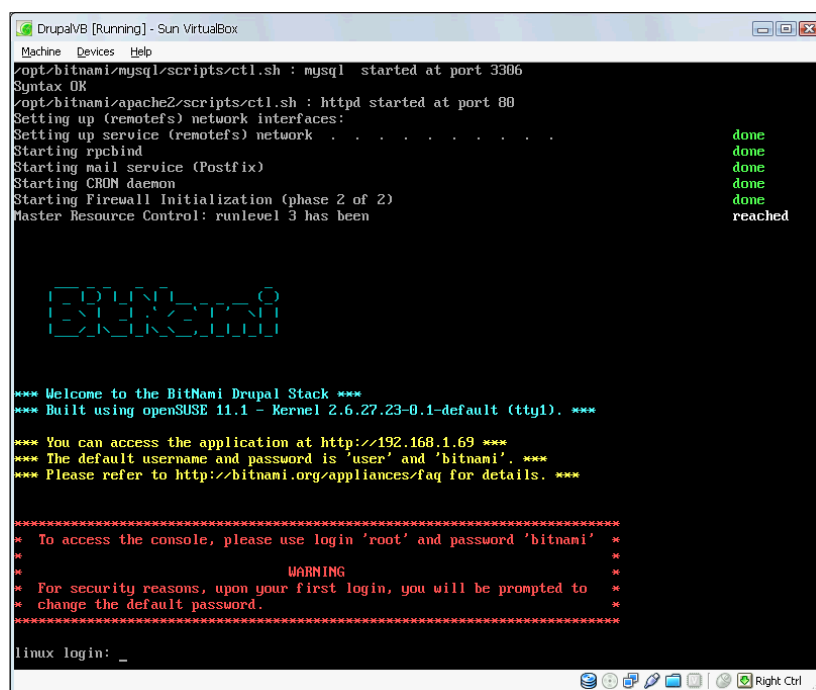
7. Select this file, and click on **Open** to close the dialog and return to the **Virtual Media Manager** screen. Make sure the Drupal .vmdk file is selected, and click on **Select** to continue:



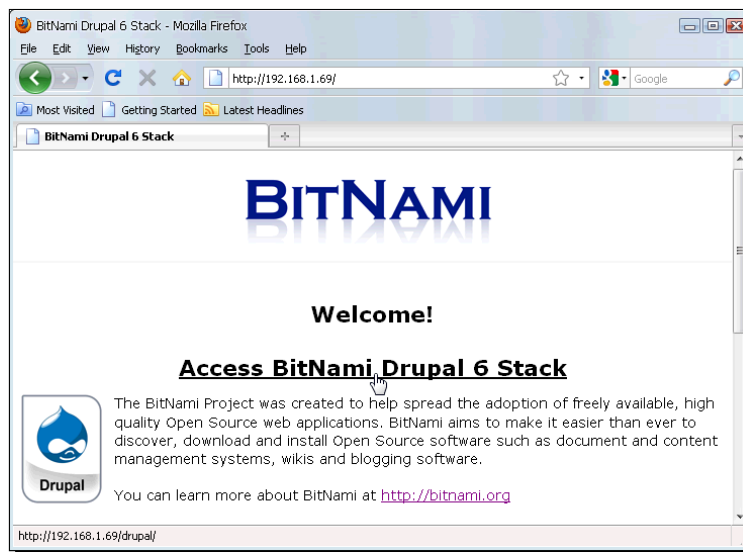
8. You'll be taken back to the **Virtual Hard Disk** screen. Click on **Next** to continue; then click on **Finish** to exit the **Create New Virtual Machine** wizard and return to the VirtualBox main screen. Your new DrupalVB virtual machine will appear selected on the VirtualBox virtual machines list.
9. Click on the **Settings** button to open the **DrupalVB – Settings** screen, and go to the **Network** category. Select the **Bridged Adapter** option from the **Attached to** field, and click on **OK** to continue:



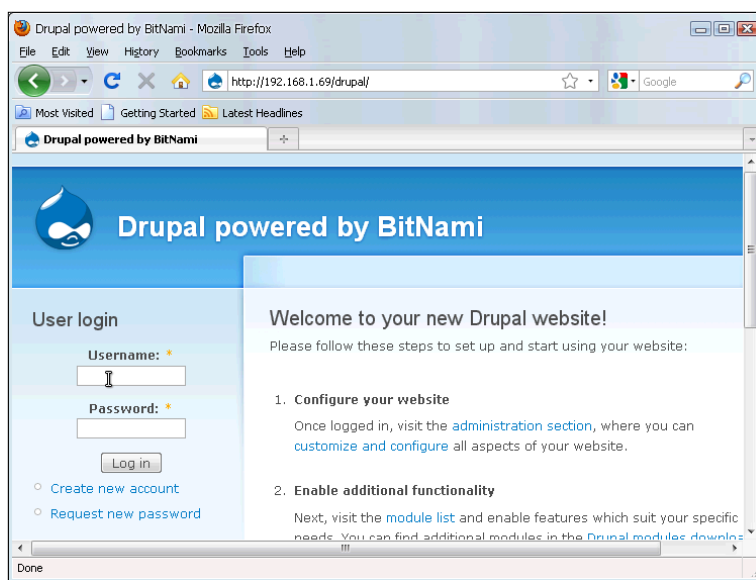
10. You'll return to the VirtualBox main screen. Now click on the **Start** button to start your new Drupal virtual machine.
11. Once your Drupal VM finishes booting up, the following screen will show up:



- 12.** The *** **You can access the application at http:** line in the above screen will indicate the IP address of your Drupal virtual machine. Take note of the IP address because you'll need it to access your Drupal VM from the other PCs in your LAN, including your host PC.
- 13.** Now open a web browser window in your host, and type the IP address of your Drupal VM (in my case, the IP address is `http://192.168.1.69`, but you'll need to replace it with your own IP address). The **BitNami Drupal 6 Stack** screen will appear, as shown below:



- 14.** To access the Drupal website, click on the **Access BitNami Drupal 6 Stack** link, and the **Welcome to Drupal** screen will appear:



- 15.** The first time you log into Drupal, you'll need to type in the default username (user) and password (bitnami), but after that, I definitely recommend that you change those values to keep hackers away from your Drupal website!

What just happened?

Well, I bet you never thought that having your own Drupal website would be so easy, right? That's what virtual appliances are for! And you can get a lot of them in the bitnami.org website! All you have to do is create a new virtual machine, but instead of using a blank hard disk, you add the pre-configured `.vmdk` image with all the required software applications such as Apache, MySQL, and PHP already installed and configured along with Drupal so that you can start using your Drupal installation right away! Oh, and don't forget to change the networking mode from NAT to bridged on your virtual machine so that you can access your Drupal virtual appliance from your host PC and from any other PC connected to your LAN!

Have a go hero – trying out other virtual appliances from Bitnami.org

Hey, now that you can run your own Drupal website, maybe you'd like to test other virtual appliances from the <http://www.bitnami.org> website, huh? You can choose from a wide variety of virtual appliances, and the process of downloading and installing them in a VirtualBox virtual machine is exactly the same as with the Drupal virtual appliance you've just installed. So go on and experiment with all the open source software you always wanted to try but never had the chance until now. VirtualBox makes it so easy for you!

Pop quiz – working with virtual appliances

1. You've been working on your company's Drupal website for a month now, thanks to the VirtualBox book and the BitNami Drupal appliance. Suddenly, one morning you find out the server hosting the Drupal website stopped working. Fortunately, you backed up your virtual appliance's hard disk image! What would you need to do to get your Drupal website back in business?
 - a. Download the BitNami Drupal virtual appliance again, and install it on a new computer.
 - b. Get a new computer, install VirtualBox, copy your Drupal virtual appliance's hard disk image to your new computer, create a new virtual machine, and select the hard disk image you've just copied to have your Drupal website running back again.
 - c. Go to the nearest bridge, and throw yourself off it!
2. What's the difference between a virtual image and a virtual appliance?
 - a. A virtual appliance is a preconfigured virtual machine, and a virtual image is the storage area—or virtual disk—of a virtual appliance.
 - b. A virtual image is a `.vmdk` file, and a virtual appliance is a `.vdi` file.
 - c. A virtual image is for Windows-based hosts, and a virtual appliance is for Unix-based hosts.
3. What formats does VirtualBox support for virtual images?
 - a. DVD and CD.
 - b. `vmdk`, `.vdi`, and `.vhd`.
 - c. Zip files.

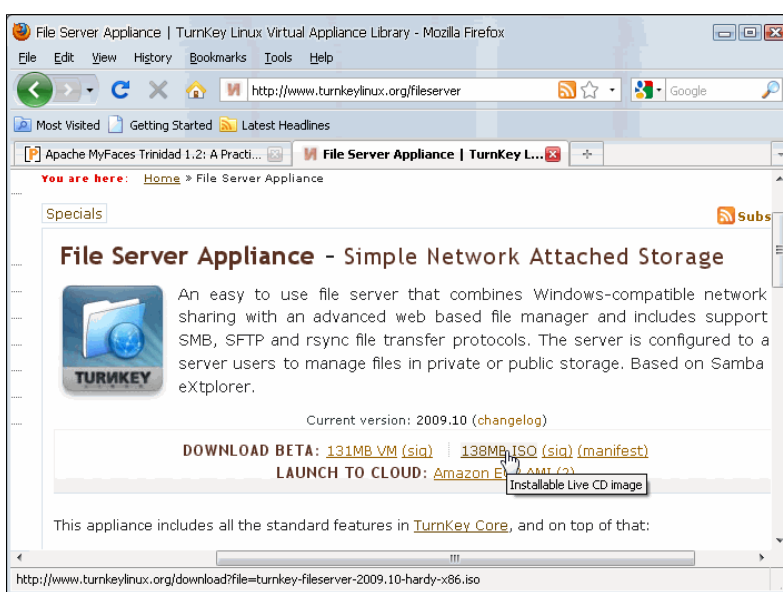
Using the Turnkey Linux File Server appliance

I must say that one of my favorite virtual appliances websites is <http://www.turnkeylinux.org>. One of the reasons is that I've been an Ubuntu Linux user for a long time, and they use it as their base operating system. I already showed you how to import a virtual appliance using the Turnkey Linux Wordpress appliance as an example. Now I'll show you how to use the Turnkey Linux File Server appliance in a slightly different but equally effective way.

Time for action – using the Turnkey Linux File Server appliance

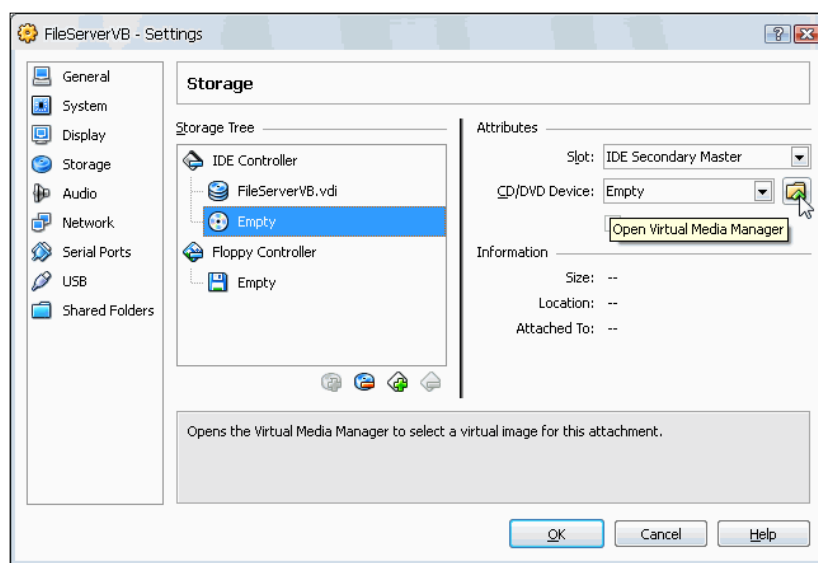
In this exercise, you'll learn how to download and use the Turnkey Linux File Server virtual appliance from <http://www.turnkeylinux.org>.

1. Open a web browser window on your host, go to <http://www.turnkeylinux.org/fileserver>, scroll down until you locate the **DOWNLOAD BETA** section, and click on the **Installable Live CD image** link (at the time of this writing, it was labeled as **138MB ISO**, but maybe you'll see a different size):

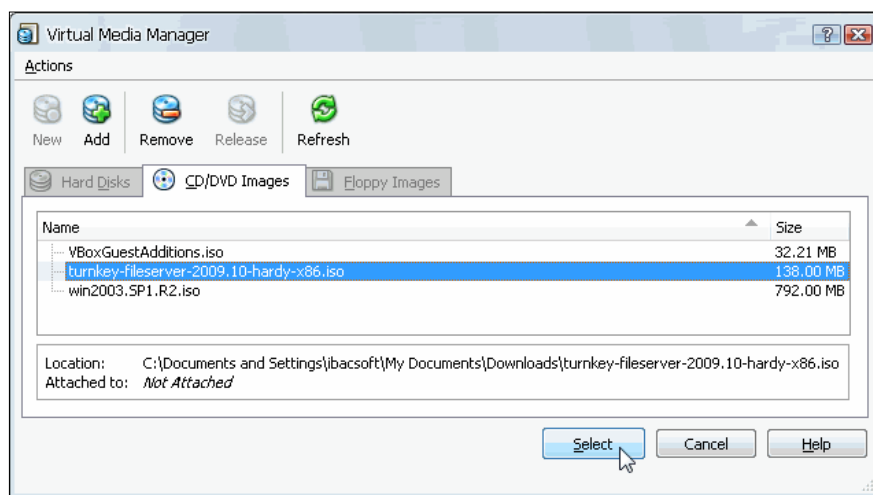


2. Wait for the open/download file dialog to appear, and save the ISO file to your hard drive.
3. Once the file is downloaded, open VirtualBox, and click on the **New** button to create a new virtual machine. The **New Virtual Machine Wizard** will appear. Click on **Next** to continue.

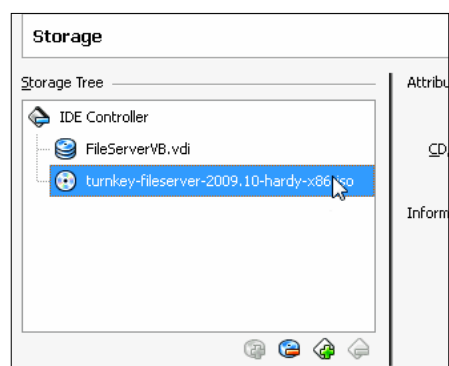
4. The **VM Name and OS Type** screen will show up next. Type **FileServerVB** in the **Name** field, select **Linux** as the **Operating System** and **Ubuntu** as the **Version**, and click on **Next** to continue.
5. Leave the default **384 MB** value in the **Memory** dialog box, and click on **Next** to continue.
6. Now the Virtual Hard Disk screen will show up. Leave the default **Create new hard disk** option, and click on **Next** twice.
7. Leave the default **Dynamically Expanding Storage** option in the **Hard Disk Storage Type** dialog, and click on **Next** to continue.
8. Leave the default values in the **Virtual Disk Location and Size** dialog, and click on **Next** to continue.
9. Click on **Finish** twice to exit both **Summary** screens and return to the VirtualBox main screen.
10. Make sure your **FileServerVB** virtual machine is selected, and click on the **Settings** button to go to the **FileServerVB – Settings** dialog. Now go to the **Storage** section, select the **Empty** item under the **Storage Tree** section, and click on the **Open Virtual Media Manager** button under the **Attributes** section to open the **Virtual Media Manager** window:



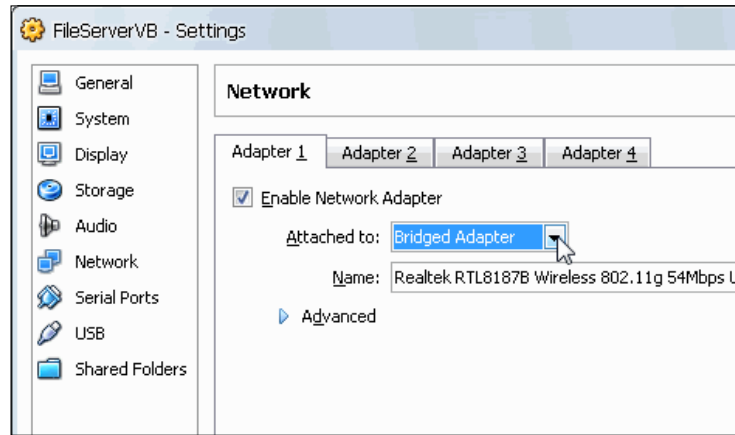
- 11.** Once the **Virtual Media Manager** window shows up, click on the **Add** button, and go to the directory where you downloaded the Turnkey File Server ISO image. Then double-click on that file to add it to the Virtual Media Manager list, and click on **Select** to use it with your `FileServerVB` virtual machine:



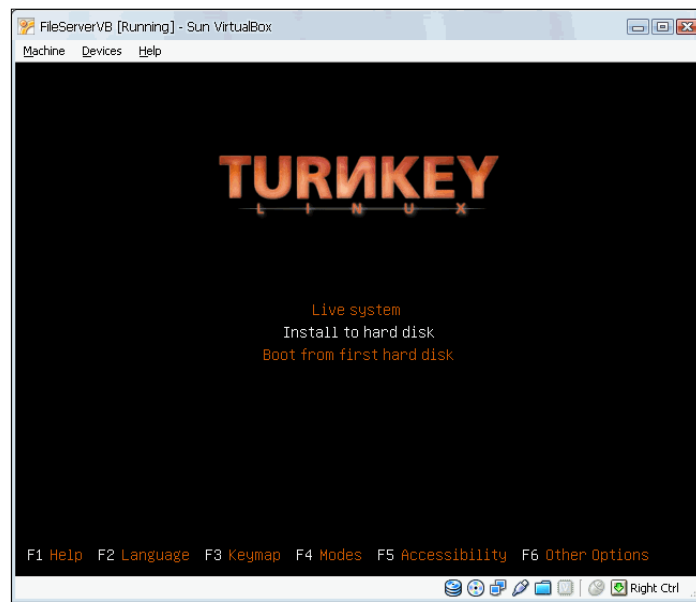
- 12.** Next you'll be taken back to the **FileServerVB – Settings** dialog, and the Turnkey File Server ISO image will appear under the **Storage Tree** section:



- 13.** Now go to the **Network** category, and select **Bridged Adapter** on the **Attached to** field:



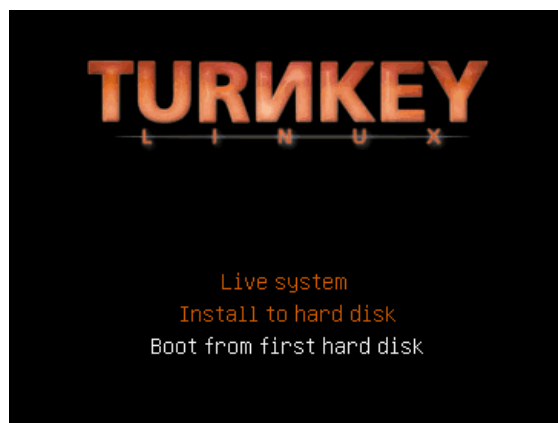
- 14.** Click on OK to close the **FileServerVB – Settings** window and return to the VirtualBox main screen. Now click on the **Start** button to start your FileServerVB virtual machine.
- 15.** The **Turnkey Linux** screen will appear next. Select the **Install to hard disk** option, and hit *Enter* to continue:



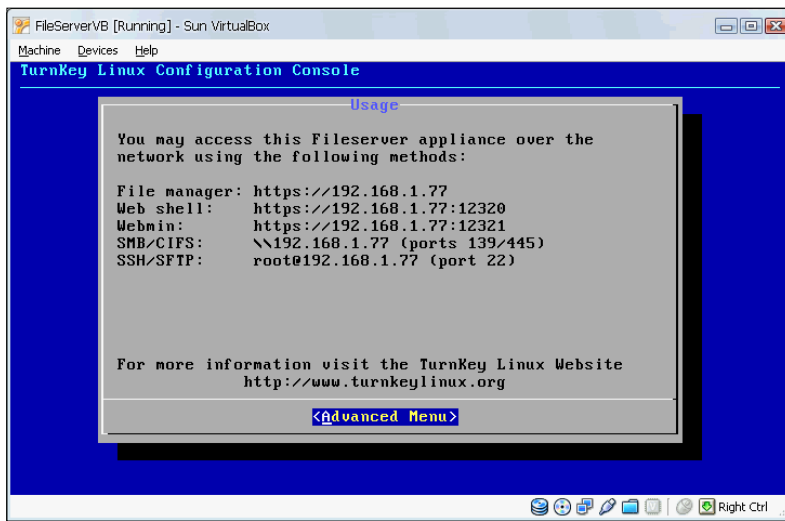
- 16.** Wait until the **Debian Installer Live** screen shows up, select the **Guided – use entire disk option**, and hit *Enter* to continue:



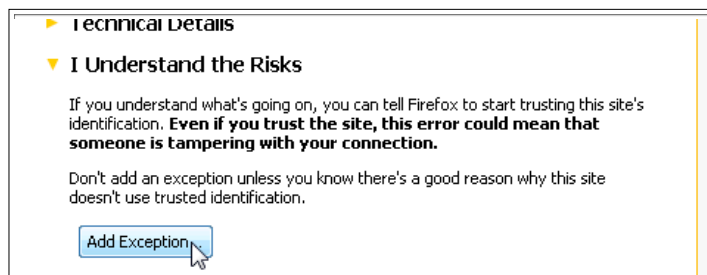
- 17.** Next, the Debian Installer will ask if you want to write the changes to disk. Select **Yes**, and hit *Enter* to continue.
- 18.** Wait until the Debian Installer formats your virtual machine's hard drive, copies the required files, and installs the system on it. When finished, it will ask you to set a password for the root user. You'll need to type it and hit *Enter* twice to continue.
- 19.** Eventually, the Debian Installer will show a message to indicate that the installation is complete and will ask if you want to restart the system. Select **Yes**, and hit *Enter* to continue.
- 20.** Wait until your virtual machine reboots and the Turnkey Linux screen shows up. This time, select the **Boot from first hard disk** option, and hit *Enter* to continue:



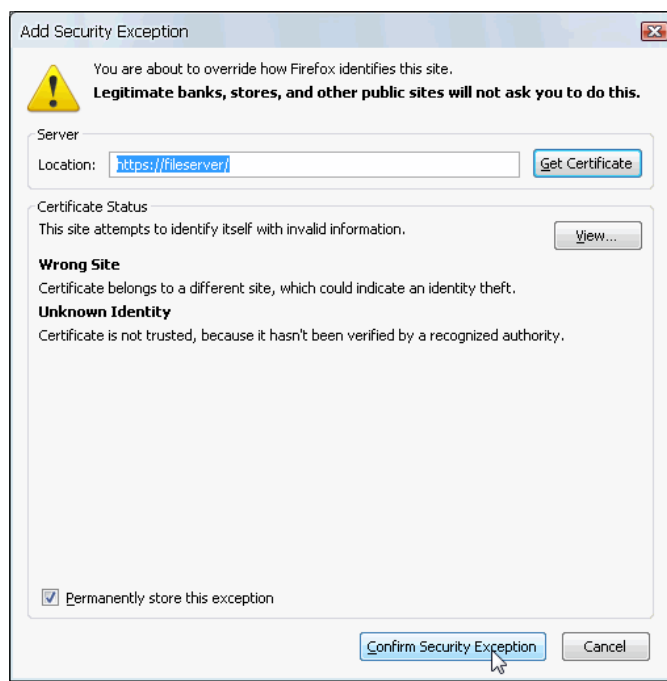
- 21.** Now just wait until the **TurnKey Linux Configuration Console** screens shows up:



- 22.** The above screen will show the IP address of your **FileServerVB** virtual machine. You need it to access its contents, add/modify/delete users or files, and so on. The Turnkey File Server also comes preconfigured so you can use the `https://fileserver` URL instead of the IP address.
- 23.** Open a web browser window in your host PC, and type `https://fileserver` to enter the File manager interface (remember to use **https** instead of **http**). If you're using Mozilla Firefox, the **Untrusted Connection** page will show up.
- 24.** Don't worry about this error page. It's completely normal due to the fact that you need to use an https connection, and you don't have a security certificate. If you were running a production server, you'd need to create a certificate, and this error page wouldn't show up. For now, just click on the **I Understand the Risks** link and then on **Add Exception...** to continue:



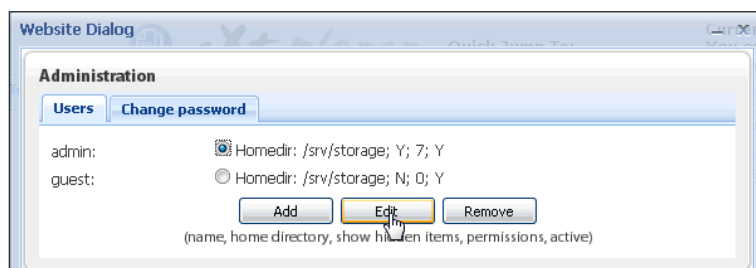
- 25.** The **Add Security Exception** dialog will appear next. Leave the default values, and click on the **Confirm Security Exception** button to continue:



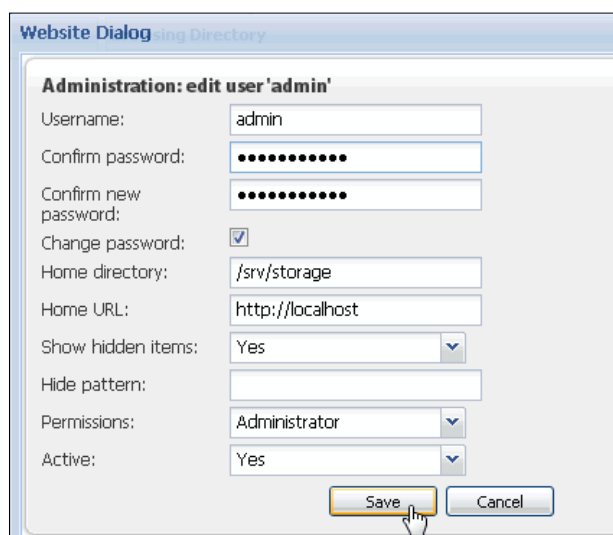
- 26.** Now the **eXtplorer** login screen will show up. Type **admin** in the **Username** field and **turnkey** in the **Password** field (the default values). Then click on the **Login** button to continue.
- 27.** The first time you log into eXtplorer, a warning dialog shows up to remind you to change your password. Click on **OK** to continue:



- 28.** The **Website Dialog** window will open up next. Select the admin user, and click on **Edit** to continue:

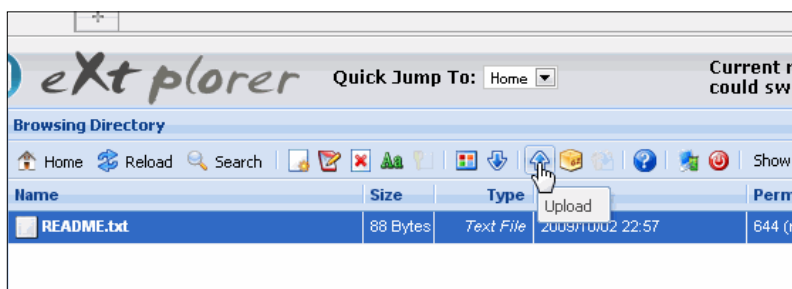


- 29.** Type the new password in the **Confirm password** and **Confirm new password** fields; then enable the **Change password** check box, and click on **Save** to continue:

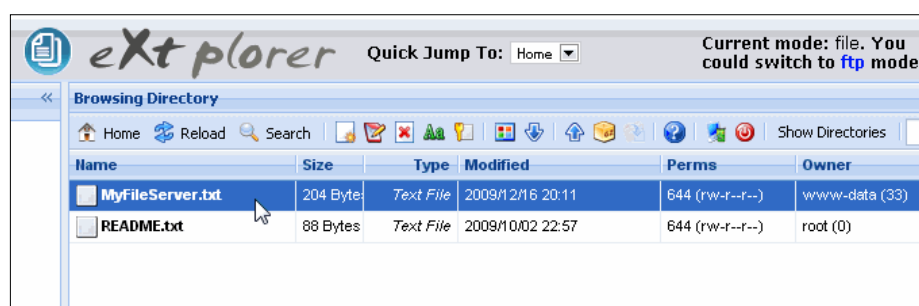


- 30.** Close the **Website Dialog** window to return to the eXtplorer home page. The `README.txt` file will appear on your user files list.
- 31.** Now let's try out eXtplorer's file uploading capabilities. Open a text editor in your host PC, and type the following text:
- ```
This is my first file server virtual appliance!
Now I'll be able to share files with other users in
a LAN, or if I open port 80 on my firewall, I'll be
able to share my files with other Internet users!
```

32. Name the file `MyFileServer.txt`, and save it in your hard drive. Then go to the eXtplorer home page, and click on the **Upload** button:



33. The **Upload file(s)** screen will show up next. Click on the **Browse...** button of the **File 1** field to open the **File Upload** dialog. Go to the directory where you saved the `MyFileServer.txt` file, and double-click on it to add its path to the **File 1** field and return to the **Upload file(s)** window. Then click on **Save** to begin the uploading process.
34. You'll be taken back to the eXtplorer home page, and the `MyFileServer.txt` file you've just uploaded will appear above the `README.txt` file:



35. Now you have your own virtual file server appliance!

### ***What just happened?***

Turnkey Linux appliances offer you an excellent way to create preconfigured virtual machines—or virtual appliances—with just a few clicks. A virtual file server appliance can be very useful in several scenarios: at home, at school, or at the office. Since most families nowadays have more than one PC or laptop at home, having a common file server is a very handy way of sharing photos, files, and all kinds of documents, don't you think?



As I said before, in the first exercise in this chapter, I taught you how to download and install a Turnkey Linux Wordpress virtual appliance in a slightly different way. That's because I wanted you to see that a Turnkey Linux virtual appliance lets you download a VMDK image or an ISO image, and it's good to try them both on VirtualBox. Now that you know how to use these excellent virtual appliances in your VirtualBox installation, you're free to choose the best way for you to install them.

### **Have a go hero – exploring your virtual file server appliance**

Now that you have your own virtual file server, it would be a good idea to explore all the features available, don't you think? You can learn more about the eXtplorer web interface at <http://extplorer.sourceforge.net/>, and you can also learn about Samba, the popular open source software suite that provides file and print sharing services, at [http://www.samba.org/samba/what\\_is\\_samba.html](http://www.samba.org/samba/what_is_samba.html).

### **Have a go hero – exploring the other Turnkey Linux virtual appliances**

Don't forget to explore all the other virtual appliances Turnkey Linux has for you! The <http://www.turnkeylinux.org/> web page has all the information you need to start wandering through the vast universe of virtual appliances available, and you'll also find valuable advice on the FAQ and BLOG sections. And always remember that practice makes perfect virtual appliances!

### **Pop quiz – virtual appliances**

1. Your company wants to try out a new web-based collaboration suite so that all of its employees can communicate efficiently. There are three products (Zimbra, phpBB, and ejabberd) that your department must evaluate so it can then choose the most appropriate for your company. What would be the best thing to do?
  - a. Call a professional consultant to help with the evaluation process.
  - b. Go to the Turnkey Linux website, download the three virtual appliances, and test them in a VirtualBox host.
  - c. Install Linux, Apache, PHP, and MySQL on three separate computers, and then evaluate Zimbra, phpBB, and ejabberd on each one of them.
2. What would be the best way to learn to use a Wordpress blog?
  - a. Use the Turnkey Linux Wordpress virtual appliance.
  - b. Hire a hosting service, and pay a monthly bill to host your Wordpress blog.
  - c. Use the BitNami Drupal appliance.

## Summary

In this chapter, we saw how virtual appliances can help make your life easier when using virtual machines and VirtualBox.

Specifically, we covered:

- ◆ What virtual appliances are, and how you can use them to reduce the installation, configuration, and maintenance times associated with creating virtual machines from scratch
- ◆ How to run virtual appliances on VirtualBox
- ◆ How to import and export virtual machines using the OVF standard
- ◆ How to install and use virtual appliances from several providers such as VirtualBox®, Images, BitNami, and Turnkey Linux

Now that you have a good grasp on virtual appliances and how to use them in your VirtualBox installation, let's go to Chapter 8 and find out about using remote virtual machines and alternative frontends for VirtualBox so that you can run headless virtual servers and run virtual machines from a remote PC.



# 8

## Managing your Virtual Machines from a Remote Computer

*Ok, you've mastered most of the basic stuff you need to work with VirtualBox and virtual machines in your home, office, or school environments. Now I'm going to teach you a pretty cool and awesome feature of VirtualBox: managing your virtual machines from a remote computer!*

*First I'll introduce you to the alternative frontends available for running VirtualBox and controlling your virtual machines, and then I'll take you on a step by step tour where you'll learn to install, configure, and run your own VirtualBox headless server from another PC.*

In this chapter you shall:

- ◆ Learn the basics about VBoxManage, the alternative frontend you can use to control and run your virtual machines from your host's command-line interface
- ◆ Learn the basics about VBoxSDL, a simple graphical frontend you can use instead of the classic VirtualBox frontend to control your virtual machines
- ◆ Learn the basics about VBoxHeadless, the VRDP server interface built into VirtualBox and how to setup a 'headless' VirtualBox server running on top of the Ubuntu Linux Server operating system
- ◆ Learn how to create, manage, and run virtual machines in your headless server from a remote PC

## Managing virtual machines from alternative front-ends

I know you're very happy with the way you've been working with VirtualBox and your virtual machines until now. Nevertheless, I need to show you the alternative frontends you can use instead of **VirtualBox**, the main GUI interface.

Specifically, the VirtualBox virtualization software includes four frontends you can use to manage your virtual machines:

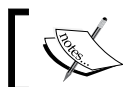
- ◆ **VirtualBox**: This is the main graphical interface you can use to interact with your virtual machines. There are some advanced features you can't use with this interface, but it's the ideal interface when you begin to delve into the virtualization world.
- ◆ **VBoxManage**: This is the command-line interface; this means it has no fancy graphics to interact with. However, it's the most complete interface in VirtualBox, and it offers a very detailed control of all the features available. You can use it to setup remote virtual machines and to manage your virtual machines through scripts.
- ◆ **VBoxSDL**: This is a simple graphical frontend you can use instead of the VirtualBox GUI in cases where users of the virtual machines must not be able to control the features offered by the main GUI interface. For example, you can use VBoxSDL to let other limited users run virtual machines, and then you can control each virtual machine with the VBoxManage interface.
- ◆ **VBoxHeadless**: This is the interface used when you plan to use virtual machines remotely. That is, you have a headless server where you have all the virtual machines installed, and then you run each virtual machine remotely through **VRDP (VirtualBox Remote Desktop Protocol)**.

Throughout the exercises in this chapter, you'll learn how to use the three alternative frontends for VirtualBox, along with a detailed step-by-step tutorial on how to set up an Ubuntu headless server and install VirtualBox on it to manage remote virtual machines from a Windows XP desktop client. And now, let the show begin!

## Using the VBoxManage frontend

Up until now we've been using the VirtualBox main graphical interface to manage our virtual machines. As you may have noticed, the VirtualBox GUI has all you need to create, configure, and run virtual machines from your host PC. Now let me tell you about the VBoxManage interface, the command-line alternative frontend for managing your virtual machines without having to interact with the VirtualBox main GUI.

From this interface, you can practically do anything related to VirtualBox and your virtual machines. In fact, everything we've seen in this book can be done with `VBoxManage`... and then some! For example, you could create a script and add the `VBoxManage startvm "UbuntuVB"` command to run a virtual machine named "UbuntuVB" automatically every time you turned on your Windows XP desktop PC! And now, let me give you a little taste of `VBoxManage` with the following exercise...



For the next exercise, you'll need the `PuppyLinuxVB` virtual machine created in Chapter 7.

### Time for action – using `VBoxManage` to start a virtual machine

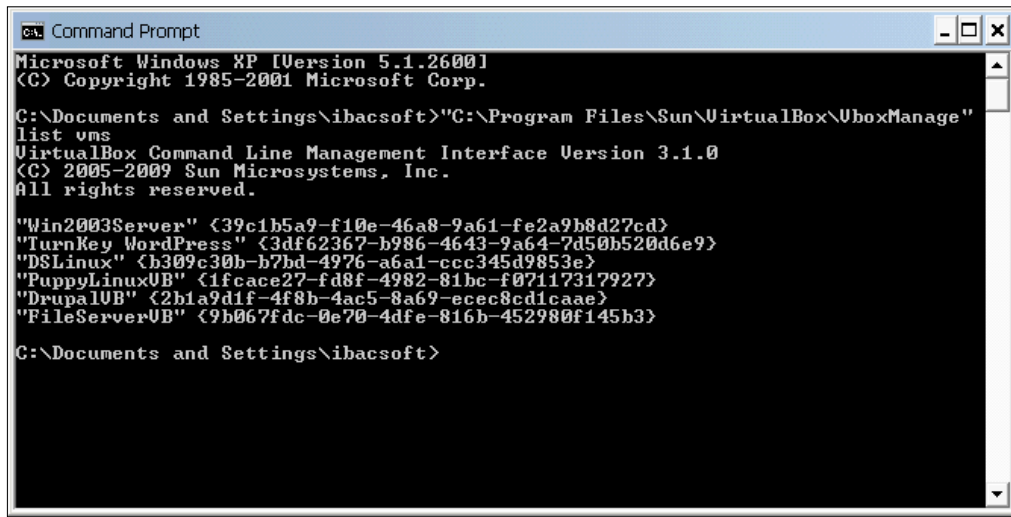
In this exercise, I'll show you how easy it is to use the `VBoxManage` interface to start a virtual machine without having to start the VirtualBox GUI. In this exercise, I'm using a Windows XP for the host PC and `PuppyLinux` as the guest virtual machine.



When using the `VBoxManage` command on a Windows host, you need to add the full path to the `VBoxManage` command. Another option is to add the VirtualBox installation folder—the default being `"C:\Program Files\Sun\VirtualBox\"`—to your system path.

On Linux systems, you don't have to specify the VirtualBox path when using the `VBoxManage` command.

1. Make sure VirtualBox is closed and, then open a Command Prompt window.
2. Type "C:\Program Files\Sun\VirtualBox\VBoxManage" list vms, and hit *Enter* to see a list of all the machines currently registered with your VirtualBox host. VBoxManage will respond with the following message (it's very likely you'll see a different list, depending on the virtual machines you've created):



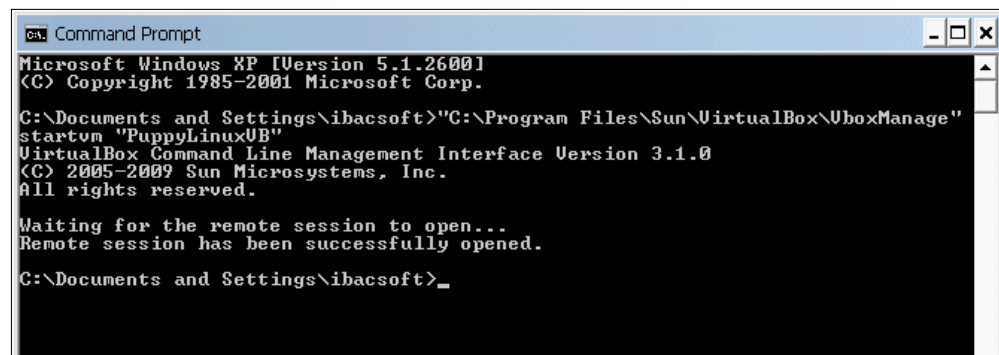
```
Microsoft Windows XP [Version 5.1.2600]
(C) Copyright 1985-2001 Microsoft Corp.

C:\Documents and Settings\ibacsoft>"C:\Program Files\Sun\VirtualBox\VBoxManage"
list vms
VirtualBox Command Line Management Interface Version 3.1.0
(C) 2005-2009 Sun Microsystems, Inc.
All rights reserved.

"Win2003Server" {39c1b5a9-f10e-46a8-9a61-fe2a9b8d27cd}
"TurnKey WordPress" {3df62367-b986-4643-9a64-7d50b520d6e9}
"DSLlinux" {b309c30b-b7bd-4976-a6a1-ccc345d9853e}
"PuppyLinuxVB" {1fcace27-fd8f-4982-81bc-f07117317927}
"DrupalVB" {2b1a9d1f-4f8b-4ac5-8a69-ec8cd1caae}
"FileServerVB" {9b067fdc-0e70-4dfe-816b-452980f145b3}

C:\Documents and Settings\ibacsoft>
```

3. The PuppyLinuxVB virtual machine you created in the previous chapter will show up on the list. Now type "C:\Program Files\Sun\VirtualBox\VboxManage" startvm "PuppyLinuxVB" and hit *Enter*. VBoxManage will respond with the following message:



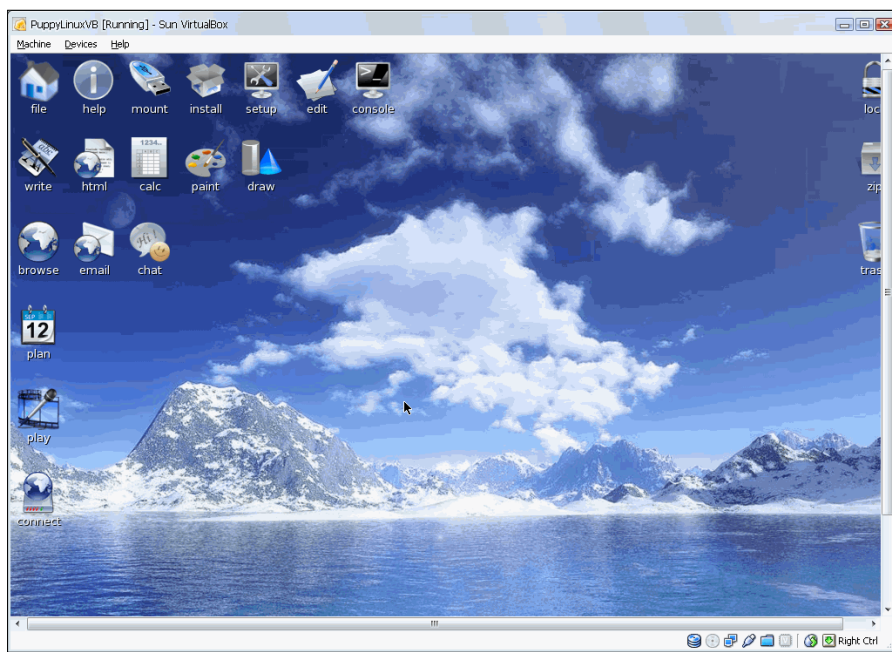
```
Microsoft Windows XP [Version 5.1.2600]
(C) Copyright 1985-2001 Microsoft Corp.

C:\Documents and Settings\ibacsoft>"C:\Program Files\Sun\VirtualBox\VBoxManage"
startvm "PuppyLinuxVB"
VirtualBox Command Line Management Interface Version 3.1.0
(C) 2005-2009 Sun Microsystems, Inc.
All rights reserved.

Waiting for the remote session to open...
Remote session has been successfully opened.

C:\Documents and Settings\ibacsoft>_
```

4. Then the PuppyLinuxVB virtual machine window will open up, as if you had used the VirtualBox GUI to open it:



5. Close your PuppyLinuxVB virtual machine, go to the Command Prompt or terminal window you opened in your host before, type "C:\Program Files\Sun\VirtualBox\VboxManage" showvminfo "PuppyLinuxVB" and hit *Enter*. The following screenshot shows a fragment of the output produced by VBoxManage:

```

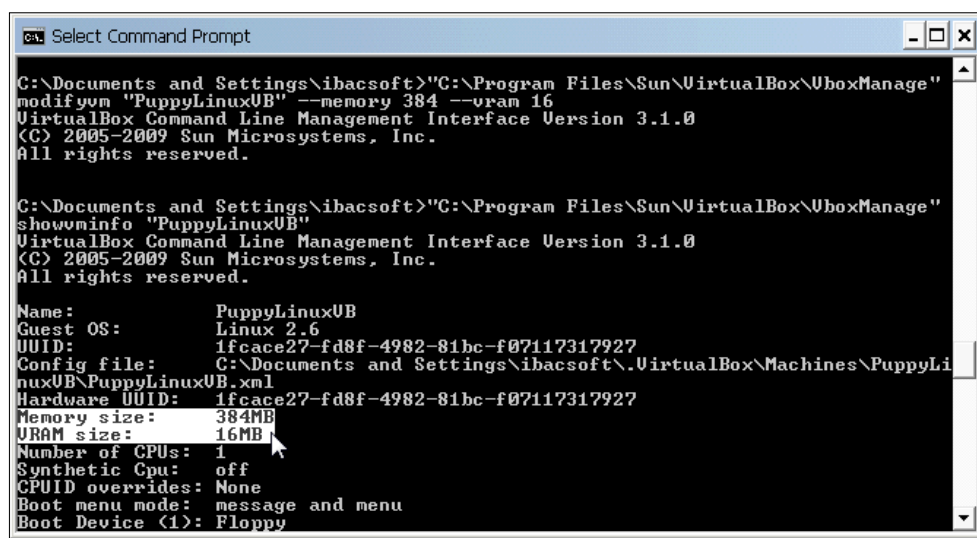
C:\Documents and Settings\ibacsoft>"C:\Program Files\Sun\VirtualBox\VboxManage"
showvminfo "PuppyLinuxVB"
VirtualBox Command Line Management Interface Version 3.1.0
(C) 2005-2009 Sun Microsystems, Inc.
All rights reserved.

Name: PuppyLinuxVB
Guest OS: Linux 2.6
UUID: 1fca27-fd8f-4982-81bc-f07117317927
Config file: C:\Documents and Settings\ibacsoft\VirtualBox\Machines\PuppyLi
nuxVB\PuppyLinuxVB.xml
Hardware UUID: 1fca27-fd8f-4982-81bc-f07117317927
Memory size: 256MB
VRAM size: 5MB
Number of CPUs: 1
Synthetic Cpu: off
CPUID overrides: None
Boot menu mode: message and menu
Boot Device (1): Floppy
Boot Device (2): DVD
Boot Device (3): HardDisk
Boot Device (4): Not Assigned
ACPI: on
IOAPIC: off

```



6. Take a look at the **Memory size** and **VRAM size** settings. Now type "C:\Program Files\Sun\VirtualBox\VboxManage" modifyvm "PuppyLinuxVB" --memory 384 --vram 16, and hit *Enter* to change your PuppyLinuxVB virtual machine's memory setting. VBoxManage will respond with a copyright notice to indicate the command was completed successfully.
7. Now type "C:\Program Files\Sun\VirtualBox\VboxManage" showvminfo "PuppyLinuxVB" and hit *Enter*. VirtualBox will show the following output:



```

C:\Documents and Settings\ibacsoft>"C:\Program Files\Sun\VirtualBox\VboxManage"
modifyvm "PuppyLinuxVB" --memory 384 --vram 16
VirtualBox Command Line Management Interface Version 3.1.0
(C) 2005-2009 Sun Microsystems, Inc.
All rights reserved.

C:\Documents and Settings\ibacsoft>"C:\Program Files\Sun\VirtualBox\VboxManage"
showvminfo "PuppyLinuxVB"
VirtualBox Command Line Management Interface Version 3.1.0
(C) 2005-2009 Sun Microsystems, Inc.
All rights reserved.

Name: PuppyLinuxVB
Guest OS: Linux 2.6
UUID: 1fcace27-fd8f-4982-81bc-f07117317927
Config file: C:\Documents and Settings\ibacsoft\VirtualBox\Machines\PuppyLi
nuxVB\PuppyLinuxVB.xml
Hardware UUID: 1fcace27-fd8f-4982-81bc-f07117317927
Memory size: 384MB
VRAM size: 16MB
Number of CPUs: 1
Synthetic Cpu: off
CPUID overrides: None
Boot menu mode: message and menu
Boot Device (1): Floppy
```

8. As you can see, the previous command modified the **Memory size** and **VRAM size** settings of your PuppyLinuxVB virtual machine.

## What just happened?

This little exercise showed you just some of the few tricks you can use with the VBoxManage interface. You can use it to see a list of all the virtual machines available, you can modify any parameter of a virtual machine (amount of RAM and video RAM, type of network card, and so on), you can create or modify a shared folder, and you can even change some of these settings when your virtual machine is running! Later on I'll show you how to set up your own VirtualBox headless server, and you'll get to use the VBoxManage interface to create, modify, and start a virtual machine, and you'll even get to create a shared folder.

## Controlling your virtual machines through VBoxManage

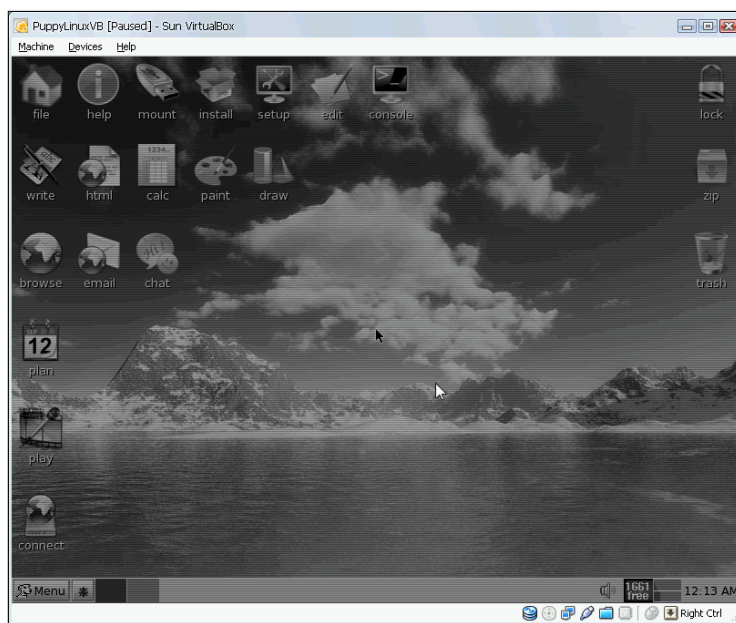
The VBoxManage interface has everything you need to control your virtual machines. Maybe you're used to working with the GUI, but once you sweep through the next exercise, you'll see why a lot of administrators and power users prefer to use text-based consoles to manage their computing environment.

### Time for action – pausing, resuming, and saving your virtual machine's state

In this exercise, I'll show you how to use the `controlvm` command to pause, resume, and save the state of your Puppy Linux virtual machine.

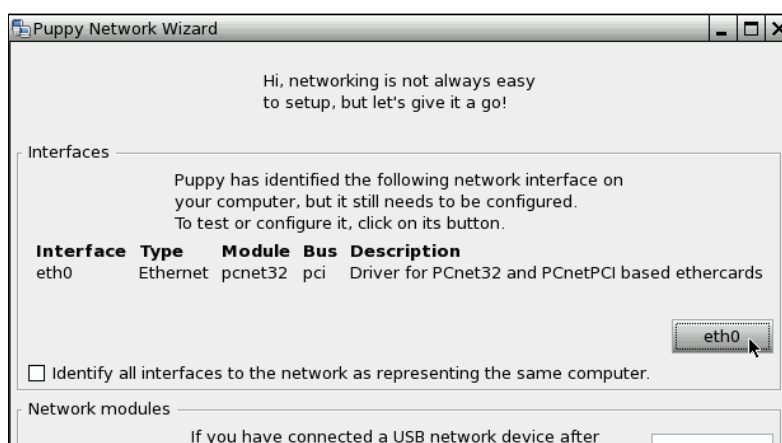
1. Start your **PuppyLinuxVB** virtual machine with VBoxManage `startvm`.
2. Once the Puppy Linux virtual machine is running, execute the following command from the command prompt:  

```
"C:\Program Files\Sun\VirtualBox\VBoxManage" controlvm
"PuppyLinux"
pause
```
3. Your **PuppyLinuxVB** virtual machine window will be grayed out, and the word **[Paused]** will appear next to the virtual machine's name in the Title bar to indicate that the VM is paused:



4. Now execute the following command from the command prompt:  

```
"C:\Program Files\Sun\VirtualBox\VBoxManage" controlvm
"PuppyLinux" resume
```
5. Your PuppyLinuxVB virtual machine will be active again. Click on the **setup** icon to open the **Puppy Setup** dialog, and then click on the **Connect to Internet by network interface** icon.
6. The **Puppy Network Wizard** window will pop up next. Your virtual machine's **eth0** interface will appear on the **Interfaces** section. Click on the **eth0** button to continue:



7. The **Configure network interface eth0** window will appear next. Click on the **Auto DHCP** button, and wait until Puppy Linux configures your network interface automatically. When it is finished, the **NETWORK CONFIGURATION of eth0 SUCCESSFUL** dialog will appear.
8. Click on **Yes** to keep the configuration, and then click on **Done** to exit the network configuration wizard. You can exit the **Puppy Setup** dialog now.
9. Click on the **browse** icon to open a browser window, and go to <http://www.virtualbox.org>.
10. Now go to the command prompt window on your host PC, and execute the following command:  

```
"C:\Program Files\Sun\VirtualBox\VBoxManage" controlvm
"PuppyLinux" savestate
```

11. VBoxManage will save your virtual machine's state and shut it down afterwards.
12. To start your virtual machine from the state it was in when you issued the `savestate` command, type `"C:\Program Files\Sun\VirtualBox\VBoxManage" startvm "PuppyLinux"` and press *Enter*. VBoxManage will restore your PuppyLinuxVB virtual machine to the state it was in right before issuing the `savestate` command.

### Have a go hero – using VBoxManage with your UbuntuVB virtual machine

Now that you've got the hang of it, use the VBoxManage interface to open your **UbuntuVB** virtual machine. Try pausing, resuming, and saving your **UbuntuVB** virtual machine's state. Then create a shared folder named `mysharedubuntu` in your host PC, and use the `VBoxManage sharedfolder add "UbuntuVB" --name "mysharedubuntu" --hostpath "C:\mysharedubuntu"` command to create a shared folder.

Remember that your virtual machine must be shut down to be able to create a shared folder, and you need to mount the shared folder in your UbuntuVB virtual machine with the `mount -t vboxsf` command. Once you're done, test the shared folder creating a text file in your **UbuntuVB** virtual machine, and then see if you can access it from your host PC.

### Pop quiz – using the VBoxManage interface

1. Do you need to open the VirtualBox GUI if you only want to see a list of all the virtual machines in your host PC?
  - a. Yep.
  - b. Nope, you can use the `VBoxManage list vms` command.
  - c. No, you can use the `VBoxManage showvms` command.
2. The VBoxManage command to create a shared folder is:
  - a. `VBoxManage createshared`
  - b. `VBoxManage sharedcreate`
  - c. `VBoxManage sharedfolder add`
3. If you try to start a virtual machine without opening the VirtualBox GUI:
  - a. Everybody will think you're crazy!
  - b. You'll need to use the `VBoxManage startvm` command.
  - c. You'll need to use the `VBoxManage createvm` command.

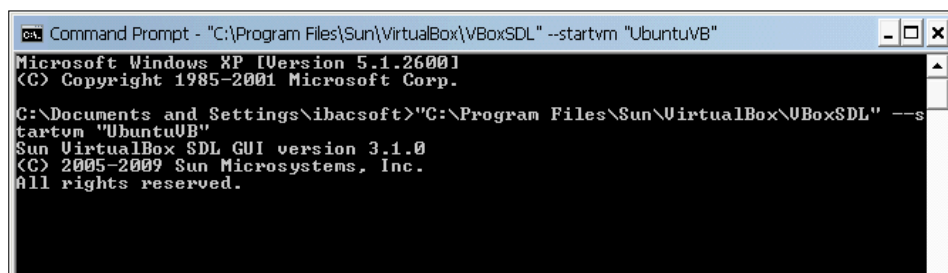
## Using the VBoxSDL simplified interface

VBoxSDL is the most simple graphics interface you can use in VirtualBox. The big difference with the main GUI is that you won't find the **Machine**, **Devices**, and **Help** menus, nor the status bar that shows information about hard disks, CD-ROM drives, network interfaces, USB devices, and so on. Basically, you can use this interface when you just want to let a user run a virtual machine without his/her being able to modify any of its settings, such as adding a USB device, saving snapshots, or adding a shared folder.

### Time for action – using VBoxSDL to start a virtual machine

In this exercise, I'll show you how to run a virtual machine through the VBoxSDL interface on a Windows XP host. If you want to use a Linux host like Ubuntu, remember to omit the `C:\Program Files\Sun\VirtualBox` part.

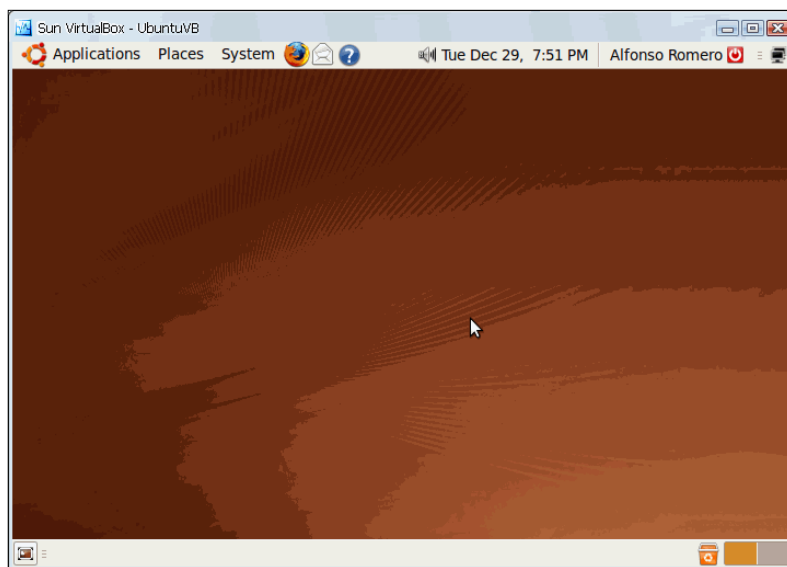
1. Make sure VirtualBox is closed, and open a Command Prompt window.
2. Type `"C:\Program Files\Sun\VirtualBox\VBoxSDL" --startvm "UbuntuVB"` and hit *Enter*. VBoxSDL will show the following output:



```
Command Prompt - "C:\Program Files\Sun\VirtualBox\VBoxSDL" --startvm "UbuntuVB"
Microsoft Windows XP [Version 5.1.2600]
(C) Copyright 1985-2001 Microsoft Corp.

C:\Documents and Settings\ibacsoft>"C:\Program Files\Sun\VirtualBox\VBoxSDL" --s
tartvm "UbuntuVB"
Sun VirtualBox SDL GUI version 3.1.0
(C) 2005-2009 Sun Microsystems, Inc.
All rights reserved.
```

3. At the same time, the `UbuntuVB` virtual machine window will open up. Wait until the login screen appears, and log into your Ubuntu virtual machine. If the window is too big, you can resize it, and the Ubuntu desktop will adjust its size automatically:



4. As you can see, the VBoxSDL screen doesn't include a menu or a status bar; that means the virtual machine user can't change any of the virtual machine's settings. You can close your UbuntuVB virtual machine now.

### ***What just happened?***

It's nice to know you can use an alternative GUI for your virtual machines, right? I mean, it would be an excellent alternative if you need to install several virtual machines in one or more host PCs and the users aren't familiarized with VirtualBox because they just want to use the virtual machines. You could create a desktop shortcut for each virtual machine so the user could just double-click on the shortcut, and the VBoxSDL interface would pop up with the virtual machine screen. And you wouldn't have to worry about the user messing up the virtual machines. How about that?

### **Have a go hero – experiment with the VBoxSDL interface**

How about running virtual machines with the VBoxSDL interface? Try creating one or more desktop shortcuts for your virtual machines using the VBoxSDL command. And remember, if you run into any trouble, you can always reach me at [questions@packtpub.com](mailto:questions@packtpub.com).

## Setting up your very own VirtualBox headless server

Ok, up until now, we've seen two of the alternative frontends available for VirtualBox. Now we are in the most interesting part of the chapter, I think. Here you'll learn to set up your very own VirtualBox headless server, which you can use to create and run virtual machines remotely! There are several step-by-step exercises I designed to get you through all the details required to have a functional VirtualBox headless server. I decided to use the Ubuntu Server 8.04 LTS operating system because it's one of the most stable and versatile Linux versions I've ever used, and I want to share that experience with you.



Before we delve into the next several exercises, let me get something straight: you will need an additional computer because it will be used as a dedicated headless server.

The hardware requirements for running an Ubuntu server are 128 MB of RAM and a decent CPU, and the hardware requirements for running a VirtualBox headless server depend on the virtual machines you plan to use.

In my case, I used an AMD 1.8 GHz Sempron processor with 2 GB of RAM and a 160 GB hard disk, so I guess you can use any decent hardware with all the RAM you can get your hands on because the amount of RAM on your headless server will define the kind of virtual machines your headless server can run. The virtual machine you'll create in the following exercises needs 384 MB of RAM, and your server must have at least 128 MB of available RAM to work, so to play it safe, I'd recommend you to have at least 1 GB of RAM on your headless server.

Oh, and if you don't know what a **headless server** is, let me explain the concept for you: a headless server is a computer acting as an ordinary server, but it doesn't need to have a monitor or any input device connected directly to it. But wait! Don't start unplugging stuff yet! You first need to install the server operating system, as I'll show you in the following subsection...

### Setting up Ubuntu Server 8.04 LTS

The first step involved in setting up a headless server is installing the Ubuntu Server 8.04 LTS operating system in your PC. You'll need to download the ISO image from the <http://www.ubuntu.com> website, and then use a CD-burning application to burn that ISO image into a CD.

## Time for action – downloading and installing Ubuntu Server 8.04 LTS

In this exercise, you'll download and install the Ubuntu Server 8.04 LTS operating system in a PC that will act as a headless server.

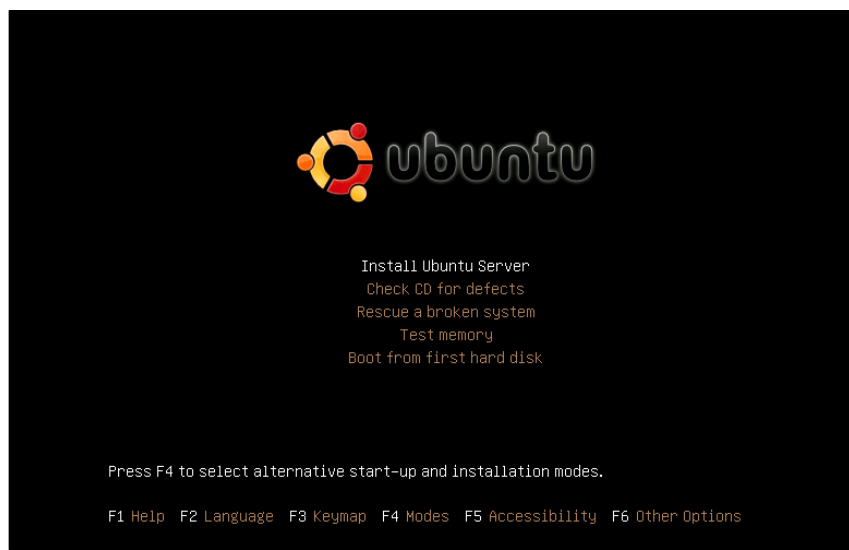
1. Open a web browser window, and go to <http://www.ubuntu.com/getubuntu/download-server>. The **Download Ubuntu Server** page will show up next. Click on the **Alternative download options** link to continue:



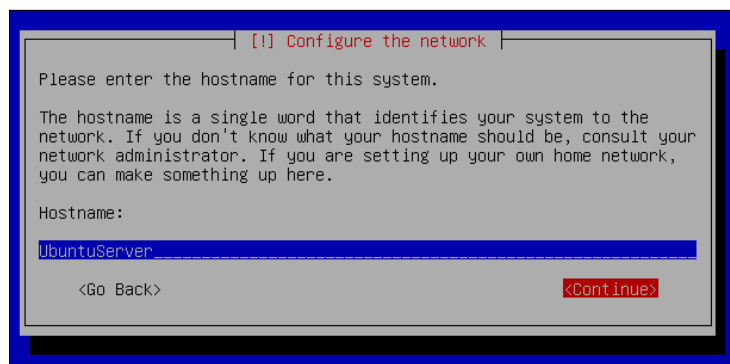
2. Select the **Download Ubuntu 8.04 LTS Server** and the **32-bit version** options, and then click on the **Begin download** button to start the download process.
3. Save the `ubuntu-8.04.3-server-i386.iso` file in your hard disk, and wait for the download to finish (approximately 1 hour on a 4 Mbps DSL connection). Once the file finishes downloading, burn it onto a CD with your favorite burning program.



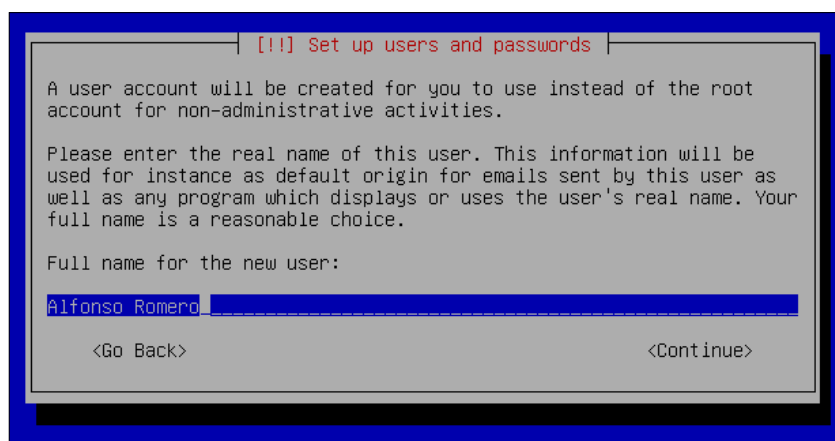
4. Insert the Ubuntu Server CD in the PC you plan to use as the VirtualBox headless server, turn it on, and wait for the Ubuntu installation menu to appear. Select the **English** language option, then select **Install Ubuntu Server**. Hit *Enter* to continue:



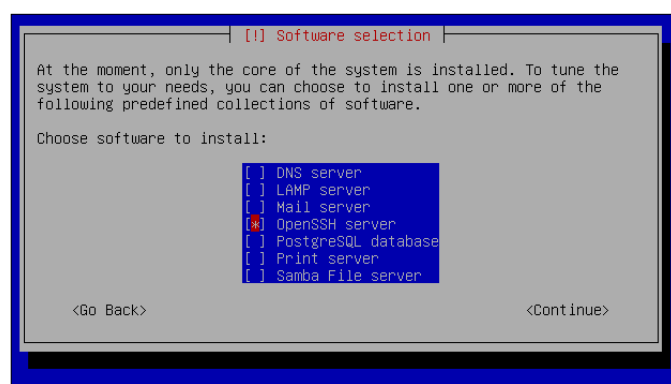
5. Eventually, the **Choose language** screen will show up. Select the **English** language option, and hit *Enter* to continue. On the next screen choose your country, region, or area, and hit *Enter* to continue.
6. Now the Ubuntu installer will ask if you want to have your keyboard layout detected. Select **No**, and hit *Enter* to continue. Then select the origin of your keyboard layout from the list that will appear next, and hit *Enter* again to continue. If the origin you selected has more than one keyboard layout, a screen will show up to let you choose the one that matches your keyboard. Hit *Enter* once you have chosen the correct layout to continue.
7. The Ubuntu installer will now proceed with the installation. Wait until the **Configure the network screen** shows up, where Ubuntu will ask you to type the hostname for your new system. Type a unique hostname for your system (I went for `UbuntuServer` after a blast of creativity), hit the *Tab* key to select the **Continue** button, and press *Enter* to continue:



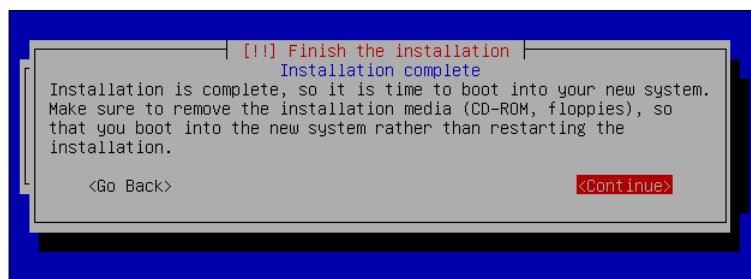
8. The **Configure the clock** screen will show up next. Select your time zone, and hit *Enter* to continue.
9. Now the **Partition disks** screen will appear. Leave the **Guided – use entire disk option** selected, and hit *Enter* to continue. If you only have one hard disk in your machine, hit *Enter* to continue. Otherwise, select the partition on which you want to install Ubuntu, and hit *Enter*.
10. A screen will show up next to confirm that you want to write the changes to disk. Select **Yes**, and hit *Enter* to continue.
11. Now you just have to wait until the Ubuntu installer partitions your machine's hard disk and installs Ubuntu. When it is finished, the **Set up users and passwords** screen will show up. Type your full name, and hit *Enter* to continue:



12. Now you need to select a username for your new account. You can leave the one that Ubuntu assigns automatically, based on your full name, or you can type any other username you wish. Just hit *Enter* when you're ready to continue.
13. The next step is to type a password for your new account. Choose a strong password, type it, and hit *Enter*. Ubuntu will ask you to type your password again for security purposes. Type it again, and hit *Enter* to continue.
14. You'll be taken to the **Configure the package manager** screen next. If you're using a proxy, you'll need to type the proxy information here. If you're connecting to the Internet from your home or you don't know what a proxy is, you can just leave this field blank and hit *Enter*.
15. The Ubuntu installer will try to configure **apt**, the package manager. Just wait until it finishes and the **Software selection** screen appears. Since we're going to communicate with the Ubuntu server through SSH, navigate to the OpenSSH server option with the arrow keys, hit *Space*, and then hit *Enter* to continue:



16. The Ubuntu installer will continue with the installation process. When finished, the **Installation complete** screen will appear. You can now remove the Ubuntu installation CD and hit *Enter* to reboot your Ubuntu server:



- 17.** Wait until the Ubuntu login screen shows up. Then type the username and password you chose before to log into your new Ubuntu server:

```
* Starting kernel log daemon... [OK]
* Starting OpenBSD Secure Shell server sshd [OK]
* Starting deferred execution scheduler atd [OK]
* Starting periodic command scheduler crond [OK]
* Running local boot scripts (/etc/rc.local) [OK]

Ubuntu 8.04.3 LTS UbuntuServer tty1

UbuntuServer login: aromero
Password:
Linux UbuntuServer 2.6.24-24-server #1 SMP Tue Dec 24 20:21:17 UTC 2009 i686

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

To access official Ubuntu documentation, please visit:
http://help.ubuntu.com/
To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

aromero@UbuntuServer:~$ _
```

- 18.** Once logged in, type `ifconfig`, and hit *Enter* to find out your Ubuntu server's IP address. Generally, the IP address will be assigned to the **eth0** or **eth1** interface, as shown below:

```
aromero@UbuntuServer:~$ ifconfig
eth0 Link encap:Ethernet HWaddr 00:1b:b9:97:52:f4
 inet addr:192.168.1.64 Bcast:192.168.1.255 Mask:
 inet6 addr: fe80::21b:b9ff:fe97:52f4/64 Scope:Lir
 UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:
 RX packets:52935 errors:0 dropped:0 overruns:0 fr
 TX packets:1498 errors:0 dropped:0 overruns:0 car
```

- 19.** Write down the IP address of your Ubuntu server, and leave it running for the next exercise.

## What just happened?

Now you can go and tell your friends you're a real Linux administrator! Well, at least for the VirtualBox part! Seriously, installing the Ubuntu 8.04 LTS server operating system is as easy as walking through the park. The hard part is learning how to be a good administrator, but that's another ticket, my friend! For now, let's just concentrate on the basic stuff needed to run your own VirtualBox headless server.

### **Pop quiz – setting up your own VirtualBox headless server**

1. To set up a VirtualBox headless server, you need to enable the LAMP server feature in Ubuntu.
  - a. True.
  - b. False.
  - c. None of the above.
2. A headless server needs to have a monitor and a keyboard connected at all times.
  - a. False.
  - b. True, because it won't run without them.
  - c. False, only the keyboard has to be connected.
3. You can't set up a VirtualBox headless server on a Windows PC.
  - a. True.
  - b. False, you can set it up on any host operating system supported by VirtualBox.
  - c. False, you can only set it up on a PC running Ubuntu Linux.

### **Accessing your headless server from a remote PC**

Now that you have your Ubuntu server running, the next step is to see if you can access it remotely through a SSH client. **SSH** stands for **Secure SHell**, and it's like using the terminal window of your Ubuntu server, but you're working from a remote PC and with a certain level of security. Since we're using a Windows XP desktop PC to connect remotely to the Ubuntu server, there's a popular open source program you can use as an SSH client: the PuTTY Telnet/SSH client.

### **Time for action – using PuTTY to access your Ubuntu server remotely**

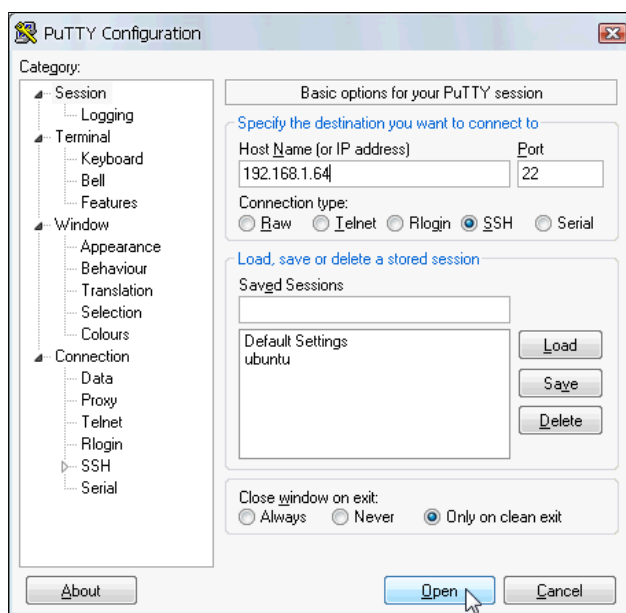
In this exercise, you'll learn how to access your shiny brand-new Ubuntu server from a remote Windows XP PC, thanks to the popular PuTTY Telnet/SSH client.

1. Open a web browser window on your desktop PC, go to <http://www.chiark.greenend.org.uk/~sgtatham/putty/download.html>, scroll down until you locate the **putty.exe** link under the **Binaries** section, and click on it.
2. Once the `putty.exe` file finishes downloading, you can double-click on it to open the PuTTY SSH client.



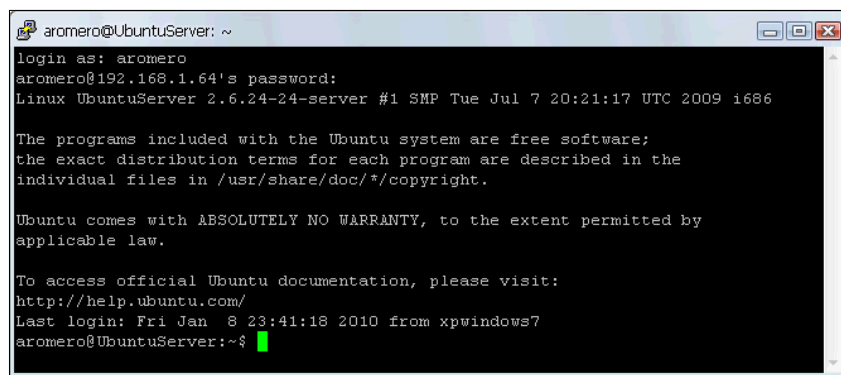
If a **Security Warning** dialog pops up when double-clicking on the `putty.exe` file, you can safely ignore it and click on the **Run** button to continue.

3. Type the IP address of your Ubuntu server in the **Host Name (or IP address)** field of the **PuTTY Configuration** screen, make sure the **SSH** option is selected, and click on **Open** to continue:



The first time you use PuTTY to connect to a remote machine, the **PuTTY Security Alert** dialog will show up complaining about the server's host key not being cached in the register. You can safely ignore this warning and click on **Yes** to continue because you're connecting to your own Ubuntu server from your own LAN, so there is no risk involved.

4. A terminal window will open up, and the **login as:** prompt will show up. Type the username and password you chose for logging into your Ubuntu server. If everything works as intended, you'll see the following welcome screen:

A screenshot of a terminal window titled 'aromero@UbuntuServer: ~'. The terminal shows the login process: 'login as: aromero', 'aromero@192.168.1.64's password:', and 'Linux UbuntuServer 2.6.24-24-server #1 SMP Tue Jul 7 20:21:17 UTC 2009 i686'. Below this is a welcome message about Ubuntu being free software and the warranty disclaimer. It also provides a link to the official Ubuntu documentation and shows the last login information: 'Last login: Fri Jan 8 23:41:18 2010 from xpwindows?'. The prompt 'aromero@UbuntuServer:~\$' is visible at the bottom with a green cursor.

```
aromero@UbuntuServer: ~
login as: aromero
aromero@192.168.1.64's password:
Linux UbuntuServer 2.6.24-24-server #1 SMP Tue Jul 7 20:21:17 UTC 2009 i686

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

To access official Ubuntu documentation, please visit:
http://help.ubuntu.com/
Last login: Fri Jan 8 23:41:18 2010 from xpwindows?
aromero@UbuntuServer:~$
```

5. Now you can connect to your Ubuntu server from your remote desktop PC! Leave the window open for the next exercise.

## ***What just happened?***

Now you can turn off your server's monitor so that we can call it a real headless server! The SSH client is the headless server administrator's best friend because you can use it from anywhere in a local area network to connect to your server. In the above exercise, we used the PuTTY SSH client because I showed you how to connect to your Ubuntu headless server from a Windows XP machine, but there are other clients you can use for the same purpose.

## **Have a go hero – using SSH from an Ubuntu PC to connect to your VB headless server**

It would be great if you could grab an Ubuntu desktop PC and use the `ssh` program from a terminal window to connect to your Ubuntu headless server.

## **Installing VirtualBox on your Ubuntu server through apt-get**

The next thing to do now is install VirtualBox on your Ubuntu server. This is where **apt-get** comes into play. And what is that, you may ask? Well, the apt-get package manager is the powerful command-line tool used in Ubuntu systems to install software the quick way. And I'm going to show you how to use it to install VirtualBox in the next exercise.

## Time for action – installing VirtualBox through apt-get on your Ubuntu server

And now let me show you one of the easiest ways to install VirtualBox on an Ubuntu headless server: through the apt-get package manager. You'll need to make some minor modifications to your Ubuntu server configuration so it can download and install VirtualBox from the official web site sources.

1. Go to the Ubuntu Server PuTTY terminal window you opened in your Windows desktop PC, and type `sudo nano /etc/apt/sources.list`, followed by *Enter*. If Ubuntu asks for your password, type it, and press *Enter* to continue.
2. The **GNU nano 2.0.7** screen will show up next. Scroll down to the end of the file, and add the following line: `deb http://download.virtualbox.org/virtualbox/debian hardy non-free`
3. Now hit *Ctrl+X*, type *Y*, and hit *Enter* to save the changes to the `/etc/apt/sources.list` file and exit the Nano editor:

```

aromero@UbuntuServer: ~
GNU nano 2.0.7 File: /etc/apt/sources.list Modified

Uncomment the following two lines to add software from Canonical's
'partner' repository. This software is not part of Ubuntu, but is
offered by Canonical and the respective vendors as a service to Ubuntu
users.
deb http://archive.canonical.com/ubuntu hardy partner
deb-src http://archive.canonical.com/ubuntu hardy partner

deb http://security.ubuntu.com/ubuntu hardy-security main restricted
deb-src http://security.ubuntu.com/ubuntu hardy-security main restricted
deb http://security.ubuntu.com/ubuntu hardy-security universe
deb-src http://security.ubuntu.com/ubuntu hardy-security universe
deb http://security.ubuntu.com/ubuntu hardy-security multiverse
deb-src http://security.ubuntu.com/ubuntu hardy-security multiverse

deb http://download.virtualbox.org/virtualbox/debian hardy non-free

File Name to Write: /etc/apt/sources.list
^G Get Help ^T To Files M-M Mac Format M-P Prepend
^C Cancel M-D DOS Format M-A Append M-B Backup File

```

4. You'll be taken back to the Ubuntu Server terminal window. Type `sudo wget http://download.virtualbox.org/virtualbox/debian/sun_vbox.asc`, and hit *Enter* to download the Sun public key you'll need to download VirtualBox.



5. Ubuntu will show the '**sun\_vbox.asc** saved [1747/1747]' message once the public key is downloaded to your Ubuntu server. Next, type `sudo apt-key add sun_vbox.asc`, and hit *Enter* to register Sun's public key. Ubuntu will respond with the **OK** message if everything goes out smoothly.
6. Now let's not forget to update the package list. Type `sudo apt-get update`, and hit *Enter*. Once Ubuntu finishes upgrading the list, it will show the **Reading package lists... Done** message, and you will be taken back to the `$` prompt.
7. The next step is to install the DKMS package. Type `sudo apt-get install dkms`, and hit *Enter*. Then type `y`, and hit *Enter* to proceed with the installation. Once Ubuntu finishes installing the DKMS package, you'll be taken back to the `$` prompt.



Remember that **DKMS** stands for **Dynamic Kernel Module Support**, and this tiny little piece of software takes care of maintaining the Linux updated so that your virtual machine can operate without any problems.

8. Next, type `sudo apt-get install linux-headers-$(uname -r)`, and hit *Enter* to install the header files for your kernel.
9. The last package you need before installing VirtualBox is `build-essential`. This package contains all VirtualBox needs to compile the kernel module. Type `sudo apt-get install build-essential` and hit *Enter*. Then type `y`, and hit *Enter* again to continue. Wait for the `$` prompt to show up again.
10. And now, to install VirtualBox on your Ubuntu server, type `sudo apt-get install virtualbox-3.1` followed by *Enter*. Ubuntu will show you a list of the packages that need to be installed:



- 12.** Now the VirtualBox installer will complain about not finding a suitable precompiled module for your Ubuntu server's current kernel. Hit *Enter* to continue:

```
aromero@UbuntuServer: ~
Package configuration

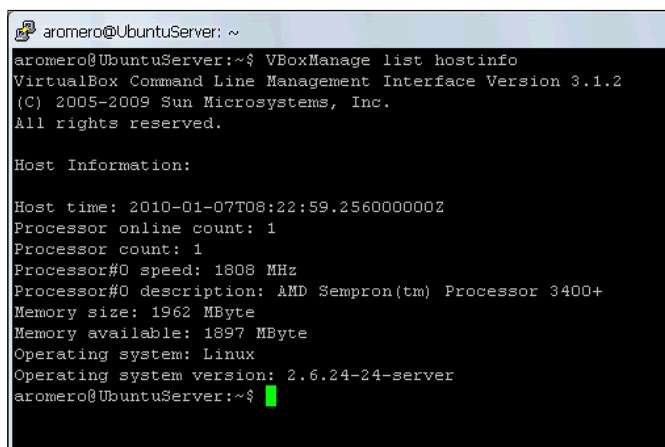
Configuring virtualbox-3.1 #####
á
á Unable to find a precompiled module for the current kernel!
á
á Without a suitable kernel module you will not be able to start any VMs.
á It is strongly recommended to compile a kernel module now. The kernel
á headers and the tools to build kernel modules (gcc, make, binutils, ...) á
á are required. However if you know that a suitable kernel module already á
á exists at another location, you might want to override the default by á
á setting KDIR=<full_path_to_vboxdrv_module> in /etc/default/virtualbox. á
á The compilation can also be done later by executing
á
á /etc/init.d/vboxdrv setup
á
á as root.
á
á
á [Ok]
á
#####
```

- 13.** Next the installer will ask if you want to have the `vbox` kernel module compiled. Select `Yes`, and hit `Enter` to continue.
- 14.** Wait until the `vbox` kernel module compilation process finishes. The following lines will appear to indicate the module was compiled successfully:

```
Messages emitted during module compilation will be logged to /var/log/vbox-instal1.log.
Success!
 * Starting VirtualBox kernel module
 * done.

Processing triggers for libc6 ...
ldconfig deferred processing now taking place
aromero@UbuntuServer:~$
```

- 15.** To test your VirtualBox installation, type `VBoxManage list hostinfo`, and hit *Enter*. VirtualBox will show the following output:

A terminal window titled 'aromero@UbuntuServer: ~' showing the output of the 'VBoxManage list hostinfo' command. The output displays host information including time, processor details, memory, and operating system version.

```
aromero@UbuntuServer:~$ VBoxManage list hostinfo
VirtualBox Command Line Management Interface Version 3.1.2
(C) 2005-2009 Sun Microsystems, Inc.
All rights reserved.

Host Information:

Host time: 2010-01-07T08:22:59.256000000Z
Processor online count: 1
Processor count: 1
Processor#0 speed: 1808 MHz
Processor#0 description: AMD Sempron(tm) Processor 3400+
Memory size: 1962 MByte
Memory available: 1897 MByte
Operating system: Linux
Operating system version: 2.6.24-24-server
aromero@UbuntuServer:~$
```

**16.** Now you can start creating virtual machines on your new headless server!

### ***What just happened?***

This was a pretty cool exercise where you learned to install VirtualBox, along with all the required software, on an Ubuntu server. I know there was a lot going on back there, but you made it through! Basically, you need to update the package sources and add the VirtualBox official download site so that apt-get can download and install VirtualBox. You also need to download and register a public key from Sun for security purposes. And because there is some compiling work involved, you need to install some basic packages used by Ubuntu to compile the modules required by VirtualBox (steps 7, 8, and 9). In step 15, you used the VBoxManage command-line interface to see some basic information from your Ubuntu headless server and to confirm that VirtualBox was installed successfully.

And now, let's get our hands dirty with your VirtualBox headless server...

## **Creating, managing, and running your first remote virtual machine on the Ubuntu headless server**

In this section, you're going to learn how to create, manage, and run your first remote virtual machine on the Ubuntu headless server you installed in the previous section. I decided to use the Ubuntu Desktop 9.10 operating system for your virtual machine because you already saw how to create a virtual machine with this operating system in Chapter 2. Now you'll see some slight differences involved in creating a virtual machine inside a headless server.

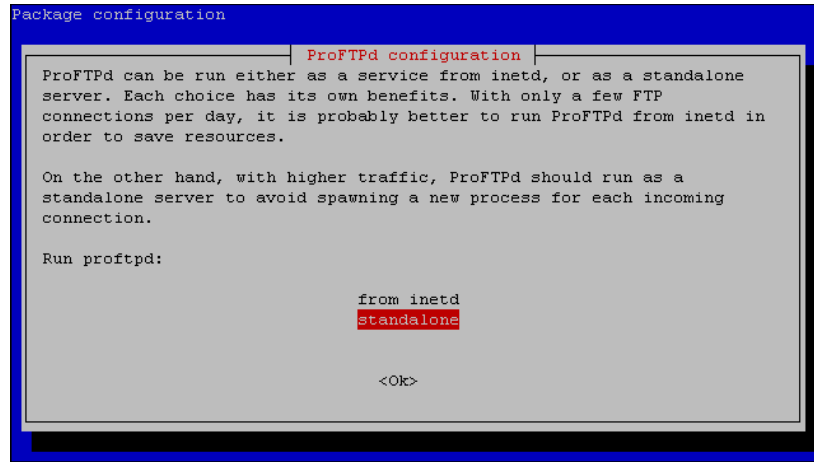
## Enabling FTP to upload guest images to your headless server

Now that VirtualBox is installed on your Ubuntu headless server, you need some methods to upload a guest ISO image containing the guest operating system you want to install on your virtual machine. In this case, I'll show you how to enable the FTP service in your Ubuntu headless server so you can upload ISO images to it from a remote desktop PC.

### Time for action – enabling proftpd on your Ubuntu headless server

In this exercise, I'll show you how to install `proftpd` in your Ubuntu headless server so that it can act as an FTP server too.

1. Go to the Ubuntu Server terminal window you opened in your Windows desktop PC, and type `sudo apt-get install proftpd`, followed by *Enter*. If Ubuntu asks for your password, type it, and press *Enter* to continue.
2. The `apt-get` package manager will ask if you want to install the `proftpd` package. Type `Y`, and then hit *Enter* to proceed.
3. The **ProFTPD configuration** screen will show up next. Select the **standalone** option, and hit *Enter* to continue:



4. If all goes well, the **Starting ftp server proftpd** message will appear on the screen, and you will be returned to the `$` prompt:

```
unpacking proftpd (from ../proftpd_1.3.1-6ubuntu1_100.deb) ...
Setting up proftpd (1.3.1-6ubuntu1) ...
Adding system user `proftpd' (UID 108) ...
Adding new user `proftpd' (UID 108) with group `nogroup' ...
Not creating home directory `/var/run/proftpd'.
Adding system user `ftp' (UID 109) ...
Adding new user `ftp' (UID 109) with group `nogroup' ...
Creating home directory `/home/ftp' ...
`/usr/share/proftpd/templates/welcome.msg' -> `/home/ftp/welcome.msg.proftpd-new'
* Starting ftp server proftpd [OK]
root@drupal6:~#
```

5. Now you can upload ISO images to your Ubuntu headless server!

### ***What just happened?***

As you can see, it's pretty easy to enable the FTP service on an Ubuntu server so it can act as an FTP server. Thanks to apt-get, the Synaptic package manager frontend, installing software from the terminal window is a piece of cake on an Ubuntu headless server! Now you're going to be able to upload ISO images of any operating system so that you can create remote virtual machines.

## **Uploading an ISO guest image to your Ubuntu server**

Ok, we have an FTP server ready to receive the ISO image file for the guest operating system we want to install in our first remote virtual machine. Now I'll show you how to use the **FileZilla** FTP client to upload the Ubuntu 9.10 ISO image from your Windows XP desktop PC to your Ubuntu headless server. In Chapter 2, you learned how to install the Ubuntu 9.10 Desktop operating system in a local virtual machine. Now you'll upload that ISO image to your Ubuntu headless server so you can create the remote virtual machine.

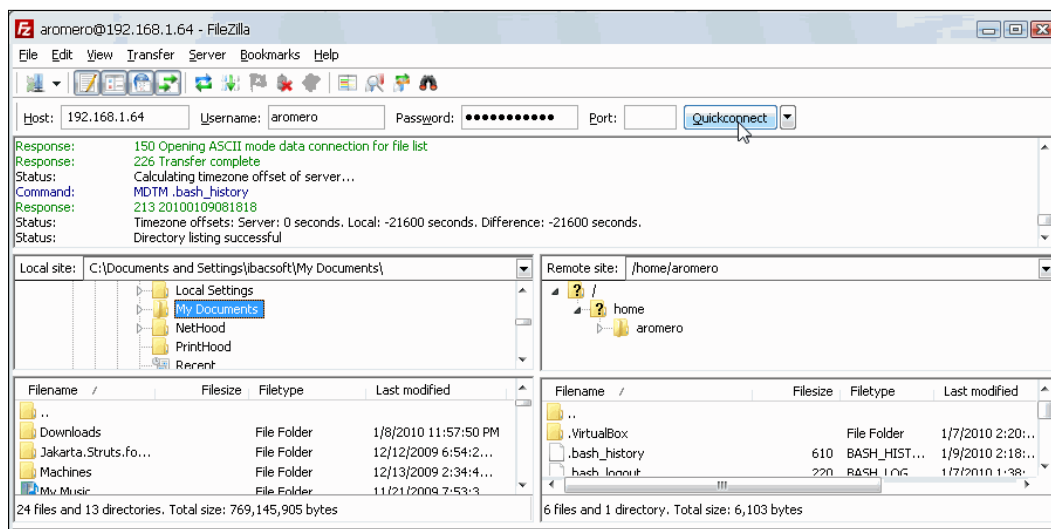
## Time for action – uploading a guest ISO image to your headless server

In this exercise, you'll learn how to use an FTP client to upload the Ubuntu 9.10 Desktop ISO image to your Ubuntu headless server so that you can later create a remote virtual machine.

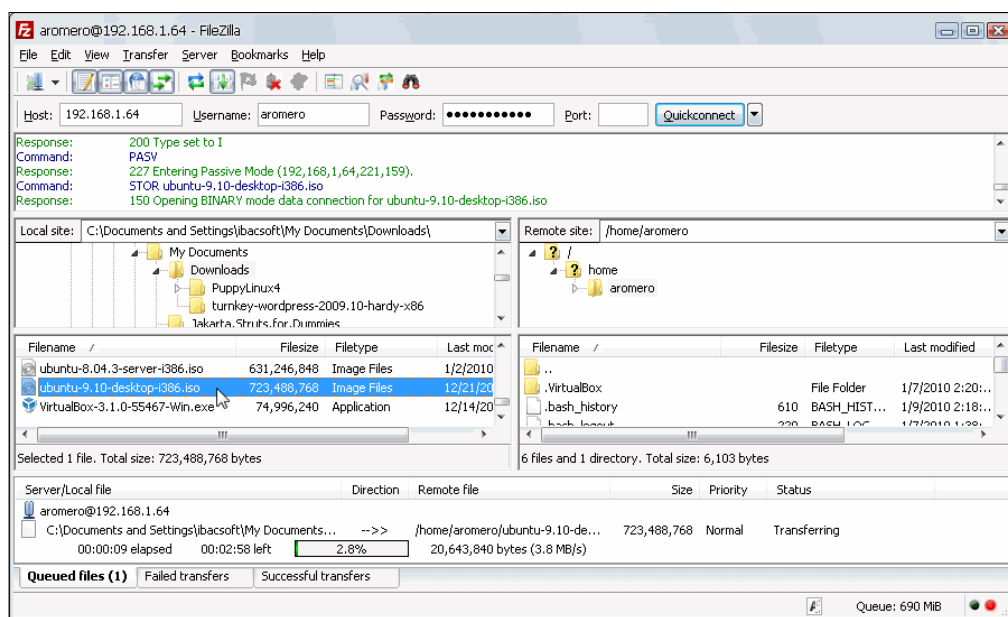
1. Open a web browser window on your Windows desktop PC, type `http://filezilla-project.org/download.php?type=client` to go to the FileZilla FTP client downloads web page, and then click on the download link for the Windows platform:



2. Download the file, and double-click on it to install the FileZilla client on your Windows desktop PC. Once the installation process finishes, open the FileZilla client, and type the IP address of your Ubuntu headless server on the **Host** field, along with your username and password. Then click on the **Quick** connect button:

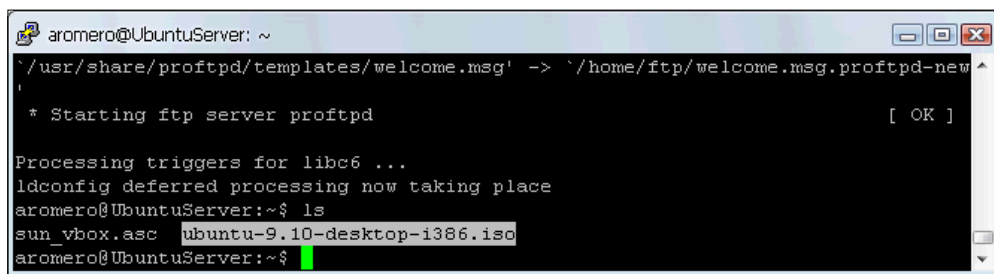


3. If the connection is successful, you'll see your account's home directory listing on the right hand side (the **Remote Site** panel) of the FileZilla window, as the above screenshot shows.
4. Now navigate through the left hand side (the **Local Site** panel) of the FileZilla window until you locate the Ubuntu 9.10 Desktop ISO image, and then double click on it to upload it to the Ubuntu server:





5. The bottom panel on the FileZilla window will show the file upload progress. Once finished, you can close the FileZilla FTP client. To verify the upload, go to your Ubuntu server's terminal window, and type `ls` followed by *Enter*. The Ubuntu Desktop ISO image filename will appear on the list:



```
aromero@UbuntuServer: ~
`/usr/share/proftpd/templates/welcome.msg' -> `/home/ftp/welcome.msg.proftpd-new ^
!
* Starting ftp server proftpd [OK]

Processing triggers for libc6 ...
ldconfig deferred processing now taking place
aromero@UbuntuServer:~$ ls
sun_vbox.asc ubuntu-9.10-desktop-i386.iso
aromero@UbuntuServer:~$
```

## What just happened?

Well, now you've uploaded the guest ISO image to your Ubuntu headless server, so you can install Ubuntu desktop in your first remote virtual machine. And all thanks to FileZilla, the open source FTP client available for Windows and Linux machines! Now let's continue with our remote virtual machine creation tutorial!

## Have a go hero – uploading ISO images to your headless server

Get all the ISO images you can from other Linux distros, Windows 2000/XP/Vista, and so on, and upload them to your Ubuntu headless server so you can later on create virtual machines from these ISO images.

## Have a go hero – using other FTP clients

Search the web for more open source FTP clients, and see if you can find any differences between FileZilla and the rest.

## Pop quiz – enabling FTP and uploading ISO images on your headless server

1. The FTP protocol lets you create virtual machines.
  - a. True.
  - b. False, it lets you upload files to your headless server.
  - c. False, it lets you create a shared folder.

2. What is FileZilla?
  - a. An FTP client you can use to upload ISO images of guest operating systems to your Ubuntu headless server.
  - b. A web server you can use to see the list of virtual machines in your headless server.
  - c. Another GUI for VirtualBox.
3. What is proftpd?
  - a. An FTP client.
  - b. Another GUI for VirtualBox.
  - c. An FTP server you can use to upload ISO images of guest operating systems to your Ubuntu headless server.

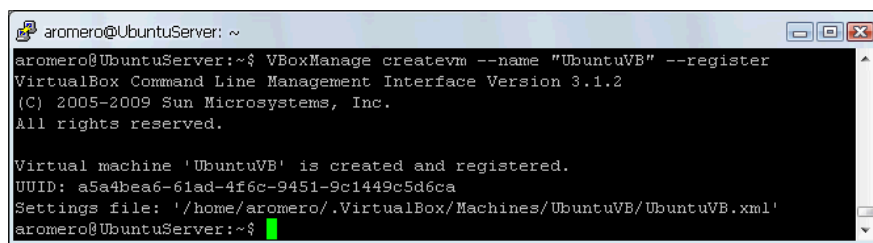
## Creating a virtual machine in your Ubuntu headless server

Ok, you have the Ubuntu 9.10 ISO image ready. What's next? You need to use the VBoxManage interface to create your remote virtual machine, and then you'll use the same interface to customize the virtual machine before installing the guest operating system on it. You already created a virtual machine in Chapter 2 with the Ubuntu 9.10 operating system. This time, I'll show you how to create it with the VBoxManage command-line interface instead of the VirtualBox GUI.

### Time for action – creating a virtual machine with VBoxManage

In this exercise, I'll show you how to create your first virtual machine in your Ubuntu headless server through the VBoxManage interface and how to customize it based on the guest operating system you plan to install.

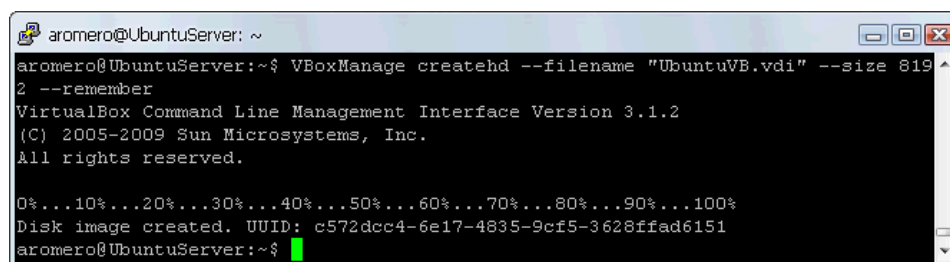
1. Go to the Ubuntu Server PuTTY terminal window you opened in your Windows desktop PC, and type `VBoxManage createvm --name "UbuntuVB" --register`, followed by *Enter*. If Ubuntu asks for your password, type it, and press *Enter* to continue. VirtualBox will respond with the following output to indicate that your virtual machine was created successfully:



```
aromero@UbuntuServer: ~
aromero@UbuntuServer:~$ VBoxManage createvm --name "UbuntuVB" --register
VirtualBox Command Line Management Interface Version 3.1.2
(C) 2005-2009 Sun Microsystems, Inc.
All rights reserved.

Virtual machine 'UbuntuVB' is created and registered.
UUID: a5a4bea6-61ad-4f6c-9451-9c1449c5d6ca
Settings file: '/home/aromero/.VirtualBox/Machines/UbuntuVB/UbuntuVB.xml'
aromero@UbuntuServer:~$
```

2. The next step is to adjust your virtual machine's settings to match the guest operating system you plan to install on it. Type `VBoxManage modifyvm "UbuntuVB" --ostype "Ubuntu" --memory 384 --boot1 dvd --nic1 nat`, and hit *Enter*. VirtualBox will respond with the first three lines of the above message to indicate a successful command.
3. Now let's create a new virtual hard drive for your virtual machine. Type `VBoxManage createhd --filename "UbuntuVB.vdi" --size 8192 --remember`, and press *Enter*. VirtualBox will show the following output to indicate that the virtual hard disk was created successfully:



```
aromero@UbuntuServer: ~
aromero@UbuntuServer:~$ VBoxManage createhd --filename "UbuntuVB.vdi" --size 8192 --remember
VirtualBox Command Line Management Interface Version 3.1.2
(C) 2005-2009 Sun Microsystems, Inc.
All rights reserved.
0%...10%...20%...30%...40%...50%...60%...70%...80%...90%...100%
Disk image created. UUID: c572dcc4-6e17-4835-9cf5-3628ffad6151
aromero@UbuntuServer:~$
```

4. After creating the hard drive, you need to attach a storage controller to your virtual machine so you can later attach the virtual hard drive to it. Type `VBoxManage storagectl "UbuntuVB" --name "IDE Controller" --add ide`, and then hit *Enter*.
5. Now you need to attach your virtual hard drive to the IDE storage controller you've just attached to your virtual machine. Type `VBoxManage storageattach "UbuntuVB" --storagectl "IDE Controller" --port 0 --device 0 --type hdd --medium "UbuntuVB.vdi"` and press *Enter*.
6. Next you need to attach the Ubuntu Desktop ISO image you uploaded before to the IDE storage controller in your virtual machine. Type `VBoxManage storageattach "UbuntuVB" --storagectl "IDE Controller" --port 0 --device 1 --type dvddrive --medium /home/aromero/ubuntu-9.10-desktop-i386.iso`, and hit *Enter*. (Remember to replace the `/home/aromero` part with your home directory, where you downloaded the Ubuntu Desktop ISO image.)
7. You've finished creating your virtual machine. Leave the terminal window open for the next exercise, where you'll see how to run it remotely.

## What just happened?

The `VBoxManage` interface is the most important piece of the virtual machine creation process. As you saw in the previous exercise, there are several steps involved when you are creating a virtual machine.

In step 1, you defined a name for your new virtual machine and registered it with your VirtualBox installation. Then, in step 2 you defined the guest operating system and the amount of RAM. You also defined the virtual DVD drive as the first boot device and you defined the NAT networking mode.

In step 3, you created a virtual hard drive for your VM, defining its name and size. If you recall from previous chapters, virtual hard drives can be dynamically expanding or fixed. In this case, the virtual hard drive is of the dynamically expanding type because it's the default option. If you want a fixed-size hard drive, you need to add the `--variant Fixed` option to the command in step 3. You also used the `--remember` option to keep the virtual hard drive registered after it was successfully written.

In steps 4 and 5, you attached a storage controller to your virtual machine so you could use the virtual hard drive you created in step 3. And in step 6, you attached the Ubuntu 9.10 ISO image to a secondary virtual device so your virtual machine can boot from it.

As you can see, the VBoxManage interface is a little bit more complicated than the VirtualBox main GUI, but once you get the hang of it, it's not so bad. Besides, on our Ubuntu headless server it's the only interface you can use to create virtual machines and modify their settings because the Ubuntu server edition doesn't install any graphical user interface by default. Now let's proceed to the really interesting part of this section...

### **Have a go hero – creating more remote virtual machines**

Practice with your new VirtualBox headless server all you can! Create a Windows XP or Windows 7 remote virtual machine, and see what differences you can spot between creating remote virtual machines and creating virtual machines with the standard VirtualBox GUI. Remember, you have to upload the ISO image of every guest operating system you want to install on your virtual machines. Or you can use the headless server's DVD/CD-ROM drive, if there's one available.

### **Have a go hero – creating remote virtual appliances**

Remember the virtual appliances you created in Chapter 7? Now it's time for you to create some remote virtual appliances. Try creating a Wordpress remote appliance from Turnkey Linux and a Drupal remote appliance from BitNami, and see if you can spot any differences between these remote appliances and the ones you created in Chapter 7.

## **Using a Remote Desktop client and running your remote VM**

Once you create a virtual machine, the next step is to start it and use a Remote Desktop client to run it from a remote PC. As we plan to run the virtual machine remotely from a Windows XP desktop PC, we need to use the Remote Desktop viewer included in Windows systems to connect to the remote virtual machine.

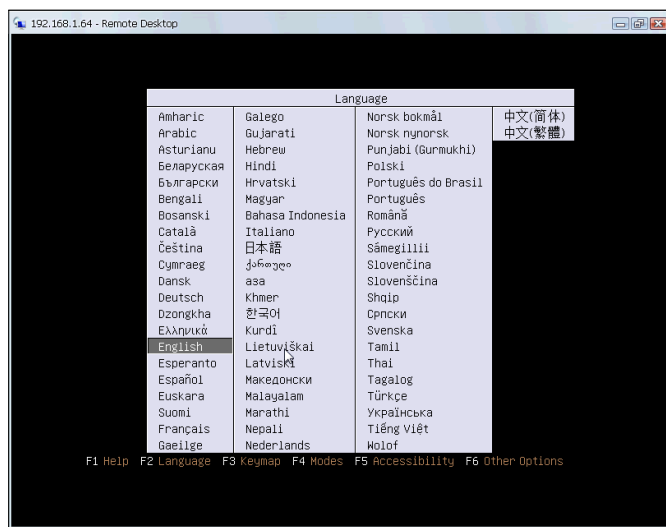
## Time for action – using an RDP viewer and starting your VM

In this exercise, I'll show you how to use the RDP viewer included in Windows, and then how to start your newly created virtual machine.

1. Before starting your remote virtual machine, it would be great if you opened the RDP viewer on your Windows desktop PC first. Select **Start | All Programs | Accessories | Remote Desktop Connection** to open the RDP viewer window, and click on the **Connect** button:



2. The **Do you trust this remote connection?** dialog will appear next. Now go to the Ubuntu Server terminal window you opened in your Windows desktop PC, and type `VBoxHeadless --startvm "UbuntuVB"` followed by **Enter**. VirtualBox will show the copyright notice along with the **Listening on port 3389** message to indicate the virtual machine started successfully. Now you can go to the **Remote Desktop Connection** window and click on the **Connect** button. The next screen will appear to indicate that your virtual machine is running:



3. Select **English** with the arrow keys, and press *Enter*. The Ubuntu Live CD menu screen will show up next. Select the **Install Ubuntu** option, and press *Enter* to begin the installation process.



There's one drawback when trying to install an operating system such as Ubuntu or Windows on a remote virtual machine: as you're using the RDP viewer, the mouse pointer will behave erratically, so you'll need to use the keyboard to navigate through the installation process. After that, you can install the Guest Additions, and then you'll be able to use the remote virtual machine seamlessly.

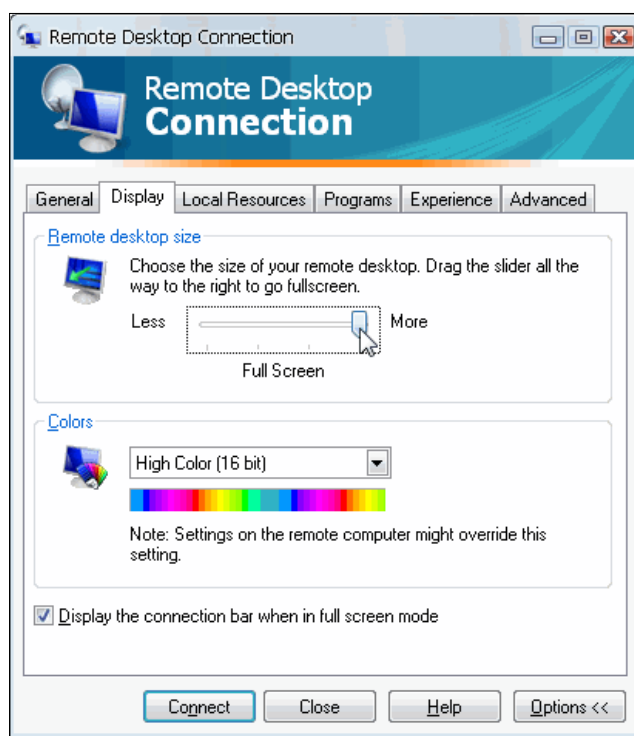
4. Once the **Welcome** window appears, select your language. Press *Enter* to continue the installation process. The **Where are you?** screen will appear next. Press *Tab* several times until you get to the **Region** list box, and then press *Enter* to open a list of available regions. Select your region, and then press *Enter* to close the list. Now press *Tab* once to go to the **Zone** list box, and select your zone the same way you selected your region. When finished, press *Tab* to navigate to the **Quit** button. Then use the right arrow key to go to the **Forward** button, and press *Space* to select it.
5. The **Keyboard layout** screen will appear next. Use the *Tab* key and the arrow keys to navigate through the layout options, and choose the appropriate layout for you. When ready, navigate to the **Forward** button, and press *Space* to continue.
6. The **Prepare disk space** screen will show up next. You can leave the default options and press *Enter* to continue.
7. On the next screen (**Who are you?**), you'll need to fill in your personal information. Type your full name, and then use the *Tab* key to navigate to the other fields so you can fill in your username and password. Leave the **Require my password to log in** option selected, navigate to the **Forward** button, and press *Space* to continue.
8. The **Ready to install** screen will appear next. Just press *Enter* to continue, and wait until the **Installation Complete** dialog shows up to indicate Ubuntu finished installing on your virtual machine. Press *Enter* to close the dialog, and the Ubuntu virtual machine will prepare for reboot. The **Please remove the disc and close the tray (if any) then press ENTER** message will appear next.
9. Open another PuTTY terminal window on your Windows desktop PC, and log into your Ubuntu headless server. Once logged in, type `VBoxManage storageattach "UbuntuVB" --storagectl "IDE Controller" --port 0 --device 1 --type dvddrive --medium emptydrive`, and press *Enter* to remove the Ubuntu ISO image. VirtualBox will show the copyright notice to indicate the command was completed successfully.

- 10.** Go back to the Ubuntu Remote Desktop screen, and press *Enter* to reboot your virtual machine. Wait until Ubuntu finishes booting up and the login screen shows up with your name on it. Press *Enter*, and then type your password to log into your Ubuntu remote virtual machine.
- 11.** Now go back to the PuTTY terminal window you opened in step 9, and type `VBoxManage storageattach "UbuntuVB" --storagectl "IDE Controller" --port 0 --device 1 --type dvddrive --medium /usr/share/virtualbox/VBoxGuestAdditions.iso` followed by *Enter* to mount the Guest Additions ISO image on your Ubuntu remote virtual machine. Again, VirtualBox will show the copyright notice to indicate that the command was completed successfully.
- 12.** Go to your Ubuntu Remote Desktop window. You'll notice the **VBOXADDITIONS\_3** dialog has just opened up. Click on OK to close the dialog, and then hit *Alt + F1* to open the **Accessories** menu. Press the down arrow key to navigate to the **Applications** submenu, then press the right arrow key to open that submenu. Use the down arrow key to navigate to the **Terminal** menu item.
- 13.** Hit *Enter* on the **Terminal** menu item to open a terminal window, and then type `sudo apt-get install dkms` followed by *Enter* to install the `dkms` package on your Ubuntu virtual machine. If Ubuntu asks for your password, type it, and press *Enter*. Wait until the **Do you want to continue?** message appears, type *Y*, and hit *Enter* to continue.
- 14.** Wait until the `dkms` package finishes installing, then type `sudo /media/cdrom/VBoxLinuxAdditions-x86.run`, and hit *Enter* to start installing the Guest Additions on your Ubuntu virtual machine. Wait until the Guest Additions Installer finishes building all the modules and installing all the VirtualBox components.
- 15.** Once the Guest Additions installation is complete, type `sudo shutdown -h now`, and hit *Enter* to shut down your Ubuntu virtual machine. The **Remote Desktop Disconnected** dialog will pop up to indicate your virtual machine was shut down and the connection was broken. Click on the **OK** button to close the **Remote Desktop** window. You'll be taken back to the **Remote Desktop Connection** dialog. Don't close this dialog because you'll need it in a moment.
- 16.** Go to the PuTTY terminal window where you started your virtual machine. The **VRDP server is inactive** message will show up to indicate the virtual machine was shut down, and the `$` prompt will be active again. Now type `VBoxHeadless --startvm "UbuntuVB"` followed by *Enter* to start your virtual machine again and test the Guest Additions installation. The following message will appear to indicate that your virtual machine started successfully:

```
aromero@UbuntuServer: ~
aromero@UbuntuServer:~$ VBoxHeadless --startvm "UbuntuVB"
VirtualBox Headless Interface 3.1.2
(C) 2008-2009 Sun Microsystems, Inc.
All rights reserved.

Listening on port 3389.
█
```

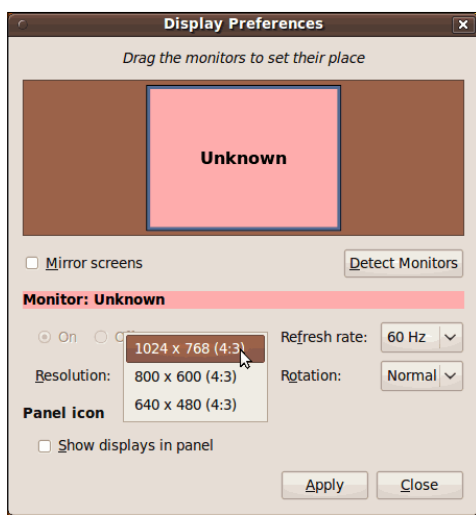
- 17.** Now go to the **Remote Desktop Connection** dialog, and click on the **Options>>** button to open the settings screen. Click on the **Display** tab, and slide the **Remote desktop size** control fully to the right, until the **Full Screen** legend appears beneath the slider bar:



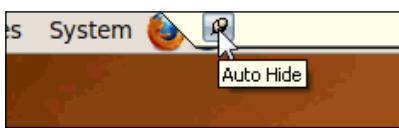
- 18.** Click on the **Connect** button to connect to your remote virtual machine. The **Do you trust this remote connection?** message will appear next. Enable the **Don't ask me again for remote connections to this computer** option, and click on **Connect** to continue.



19. The login window for your Ubuntu remote virtual machine will show up in full screen, and you will be able to move the mouse pointer as if you were working on your local machine. Log into your Ubuntu remote VM, and wait for the desktop to load. You'll notice it doesn't cover the whole screen because it has the default 800x600 resolution. To change the resolution to full screen, select **System | Preferences | Display** from the menu bar.
20. The **Display Preferences** dialog will appear next. Click on the **Resolution** list box, and select the biggest resolution available (in my case, it's 1024x768) to use the full screen mode:



21. Click on **Apply** to apply the changes. The **Does the display look OK?** dialog will pop up next, asking if you want to change your display resolution. Click on the **Keep This Configuration** button to apply the changes, and then click on **Close** to close the **Display Preferences** dialog. Now you'll be able to run your remote virtual machine in full screen mode!
22. So that you can alternate between your remote virtual machine and your local desktop, the Remote Desktop viewer will show a connection bar at the top of the screen. You can click on the **Auto Hide** button to hide/show the bar automatically so it doesn't interfere with the normal operation of your virtual machine:



23. Shut down your Ubuntu remote virtual machine for the next exercise.

## ***What just happened?***

Not so bad, huh? Now you have a fully operational remote virtual machine! Ok, I have to admit, it was a bit more complicated to install than using the traditional VirtualBox GUI, but remember that you're running it from your Ubuntu headless server!

In steps 1 and 2, you saw how to start your remote virtual machine and open the RDP viewer to see it. Then you proceeded to install the Ubuntu 9.10 Desktop operating system until you got to step 8. Then, in step 9, you used the VBoxManage command to unmount the Ubuntu ISO image from the virtual DVD drive and rebooted your virtual machine.

In step 11, you mounted the Guest Additions image on the virtual DVD drive and proceeded to install the Guest Additions on your Ubuntu virtual machine the same way as you did in Chapter 2. Then, in steps 15-16, you learned how to shut down your remote virtual machine and how to restart it to test the Guest Additions installation.

In steps 17 and 18, you saw how to adjust your Remote Desktop viewer's settings to see your remote virtual machine in full screen, and in steps 19 through 21, you learned how to adjust your Ubuntu virtual machine display to achieve the same objective.

Finally, in step 22, you learned how to use the connection bar shown by the Remote Desktop viewer so that you can alternate between your remote virtual machine and your Windows XP desktop.

Now let's see some additional material you need to configure on your remote virtual machines.

## **Have a go hero – installing Guest Additions on your other remote virtual machine**

Go and install the Guest Additions on the Windows XP or Windows 7 remote virtual machine you created in the *Have a go hero – creating more remote virtual machines* section.

## **Have a go hero – running two remote virtual machines at the same time**

How about creating and running two remote virtual machines at the same time? You can use the Ubuntu virtual machine you created in the previous *Time for action – using an RDP viewer and starting your VM* section and the Windows XP or Windows 7 remote VM from the previous *Have a go hero – installing Guest Additions on your other remote virtual machine* section.

When you start a virtual machine with the `VBoxHeadless` command, the Remote Desktop viewer connects to port 3389 on the virtual machine. If you want to start another virtual machine, you need to specify a different port. For example, if you're running your remote Ubuntu VM and you want to start a remote Windows XP VM, use the `VBoxHeadless WindowsXP -vrdpport 3390` command, and then connect through your Remote Desktop viewer specifying that port number along with your headless server's IP address – if your headless server has the 192.168.1.64 IP address, type `192.168.1.64:3390` in the Remote Desktop dialog.

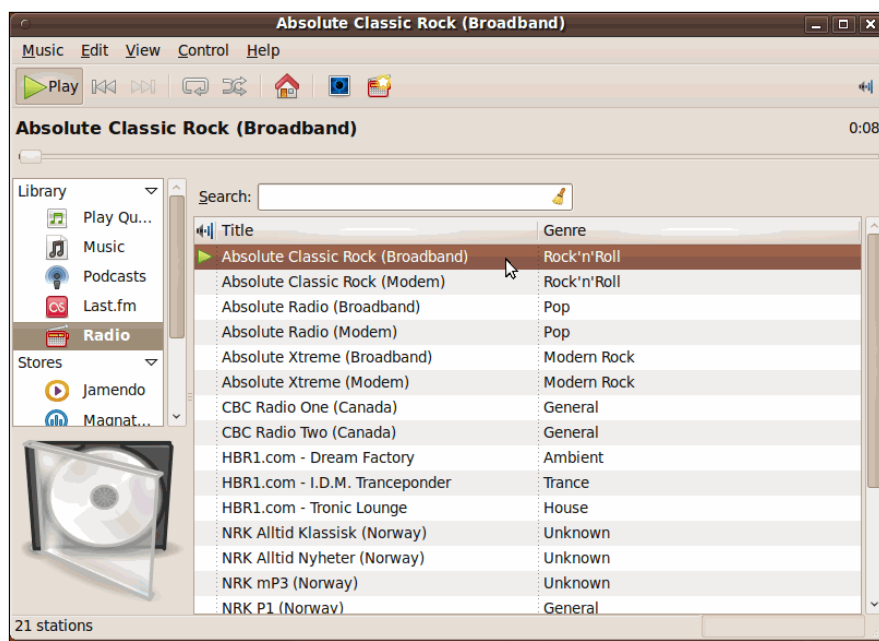
## Enabling sound on your remote virtual machines

When you use a VirtualBox headless server to host remote virtual machines, you need to use the `VBoxManage` interface because it's the only tool available to create, modify, and run virtual machines. One of the tiny little problems of this setup is that you need to define most of the virtual machine settings manually. Sound is one of them. So now, let's see how to enable sound on your new remote virtual machine.

### Time for action – enabling audio on your remote virtual machine

In this exercise, I'll show you how to enable audio on your Ubuntu remote virtual machine via the `VBoxManage` interface.

1. Go to the Ubuntu server PuTTY terminal window, type `VBoxManage modifyvm "UbuntuVB" --audio oss --audiocontroller ac97`, and press *Enter*. VirtualBox will respond with the copyright notice to indicate that the command was successful.
2. Now type `VBoxHeadless --startvm "UbuntuVB"` followed by *Enter* to start your Ubuntu virtual machine, and open the Remote Desktop viewer to connect to your remote VM. Log into Ubuntu, and you will hear the intro sound. Once the desktop shows up, select **Applications | Sound & Video | Rhythmbox Music Player** from the main menu to open up the music player included in Ubuntu.
3. Once the **Music Player** window appears, select the **Radio** option under the **Library** panel, and double-click on a radio station from the right panel to start hearing some music:



4. Cool, huh? Now you have sound enabled in your Ubuntu remote virtual machine! Shut down your Ubuntu remote virtual machine for the next exercise.

### ***What just happened?***

That wasn't too hard, was it? VBoxManage interface to the rescue! The first thing you need to do is define what audio subsystem to use. Since we're using an Ubuntu host, you can choose between the `oss`, `alsa`, or `pulse` subsystems. Personally, I've always used the `oss` subsystem, and I haven't had any complaints up until now. But you can experiment with the other two options if you like. You can even change them back if they don't work as you expected.

Now, the other option you need to define is the audio controller. In this case, you can choose between `ac97` or `sb16`. AC97 is the most common controller nowadays; that's why I chose it for this exercise.

### **Have a go hero – selecting the audio controller for your remote VM**

Try using the SoundBlaster 16 controller instead of AC97 to see if there are any performance differences in your Ubuntu remote virtual machine or if maybe sound won't work at all. Who knows?

## Have a go hero – choosing audio controllers in other hosts

It would be a good idea to investigate whether each host operating system supported by VirtualBox uses the same audio controllers or not. You'll need access to several PCs with VirtualBox installed for this purpose, so you can go to the **Audio** category in the **Settings** dialog of a virtual machine to find out what audio controllers are available on that host operating system.

## Using shared folders on your remote virtual machine

Now the time has come to show you how to use shared folders on a remote virtual machine. In this case, remember you're using a remote RDP viewer to access the virtual machine running in the Ubuntu headless server. So, technically the host is the Ubuntu headless server, and the guest is the remote virtual machine. That means you must create a folder on your Ubuntu headless server, and you'll share that folder with your virtual machine.

Once you have the shared folder up and running, I'll show you how to use the FileZilla FTP server to access the shared folder in your Ubuntu headless server from your Windows XP desktop.

## Time for action – creating and accessing a shared folder on your Ubuntu headless server

Ok, first you'll create a shared folder between your Ubuntu headless server and your virtual machine. Then you'll use the FileZilla FTP client to access that shared folder from your Windows XP desktop.

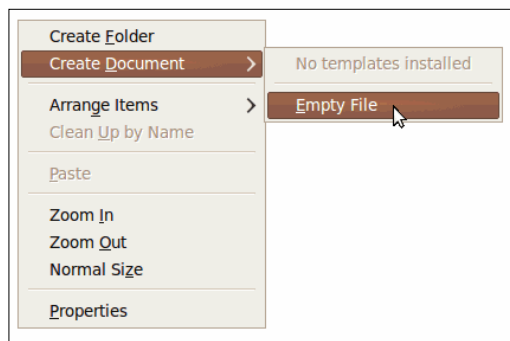
1. Go to the Ubuntu server PuTTY terminal window, and type the following lines to create the shared folder in your Ubuntu headless server (don't forget to press *Enter* after each line):

```
mkdir myshared
VBoxManage sharedfolder add "UbuntuVB" --name "myshared"
--hostpath "/home/aromero/myshared"
```

2. Now type `VBoxHeadless --startvm "UbuntuVB"` and hit *Enter* to start your remote virtual machine; then open the Remote Desktop viewer, connect to your Ubuntu remote virtual machine, and log into it.
3. Once you are inside Ubuntu's desktop, open a terminal window (**Applications | Accessories | Terminal**), and type the following lines to create the folder you want to share in your remote VM (don't forget to press *Enter* after each line, and replace `aromero` with your own username):

```
mkdir myremoteshare
sudo mount -t vboxsf -o uid=aromero myshared /home/aromero/
myremoteshare
```

4. Once the shared folder is created, select **Places | Home Folder** from the main menu to open the **Ubuntu File Browser** window, and double-click on the `myremotesshared` folder to open it.
5. Once inside the `myremotesshared` folder, right-click anywhere inside the File Browser's right panel, and select **Create Document | Empty File** from the pop-up menu to create a new empty file:



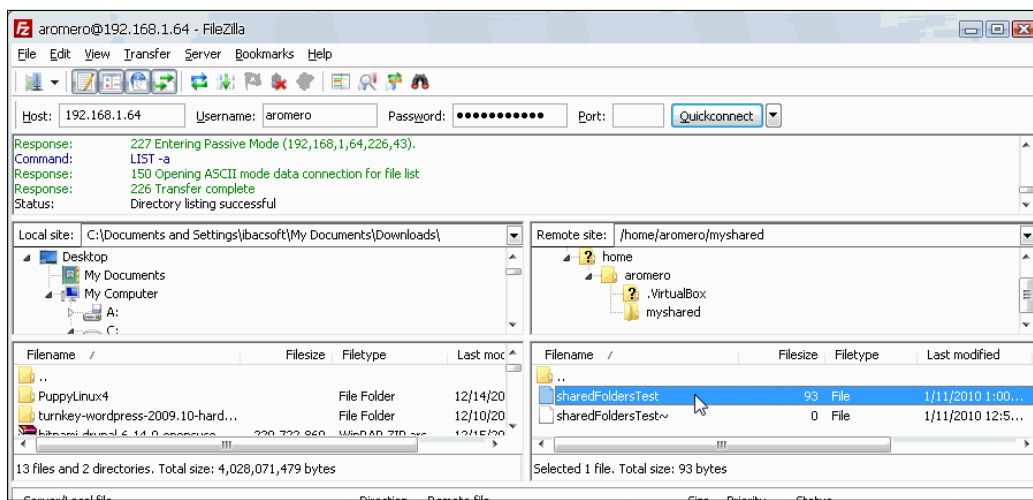
6. A new file icon will appear inside the File Browser's right panel. Type `sharedFoldersTest` to replace the new `file` text, and press `Enter` to change the filename; then double-click on the file to open it. Type `This is a test file to see if the shared folders feature works in my remote virtual machine inside the empty sharedFoldersTest file`; then hit `Ctrl + S` and `Ctrl + Q` to save the file and close the text editor.
7. Minimize your remote virtual machine screen, open a new PuTTY terminal window on your Windows desktop computer, and log into your Ubuntu headless server. Type `ls /home/aromero/myshared`, and hit `Enter`. The `sharedFoldersTest` you created on your remote virtual machine will show up in the list to indicate that your Ubuntu headless server has access to it through the `myshared` shared folder:

```

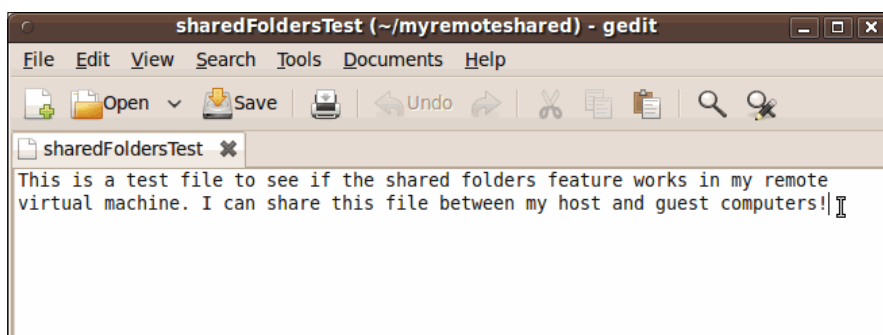
aromero@UbuntuServer: ~
aromero@UbuntuServer:~$ ls /home/aromero/myshared
sharedFoldersTest sharedFoldersTest~
aromero@UbuntuServer:~$

```

8. To access that file in your Windows desktop PC, open the FileZilla FTP client, and connect to your Ubuntu headless server. Then navigate to the `myshared` folder in the **Remote site** panel, and you'll see the `sharedFoldersTest` file inside:



9. Double-click on the filename to download it to your Windows desktop PC, and then open it with the Notepad text editor. Add the `I can share this file between my host and guest computers!` text to the file, save it, and upload it again to the Ubuntu headless server using FileZilla. Now go back to your remote virtual machine screen, and open the `sharedFoldersTest` file again. The file will show the line you added:



10. Your remote virtual machine is now ready to share folders!

## What just happened?

Very cool! Now you can share information between your virtual machines and the outside world! Oh, and you can share any kind of file, not just text files, okay?

## Have a go hero – sharing other types of files

Grab some free PDF files, JPEG, PNG images, or video files from the Internet using your Ubuntu remote virtual machine, and then access them from your Windows XP desktop via the FileZilla FTP client.

## Setting up your own remote virtual LAMP server

Running remote Ubuntu or Windows virtual machines is a good way to maximize resources if you have a very powerful computer to set up as a VirtualBox headless server, but that's not all you can do. No siree! In fact, I use a VirtualBox headless server to run my remote Windows 7 and Ubuntu virtual machines, and I also run a virtual web server, a virtual file server, and a virtual Java application server, all of them at the same time! And because I want to share my joy with you, in this section I'll teach you how to run a remote virtual web server from your very own VirtualBox headless server so you can start creating remote virtual machines to interact with remote virtual appliances like the Turnkey Linux LAMP appliance we'll work with.



In Chapter 7, *Using Virtual Appliances*, you saw how to download virtual appliances from the Turnkey Linux website.

In Chapter 6, *Networking with Virtual Machines*, I showed you how to use the bridged networking mode to access your Ubuntu virtual machine's web server.

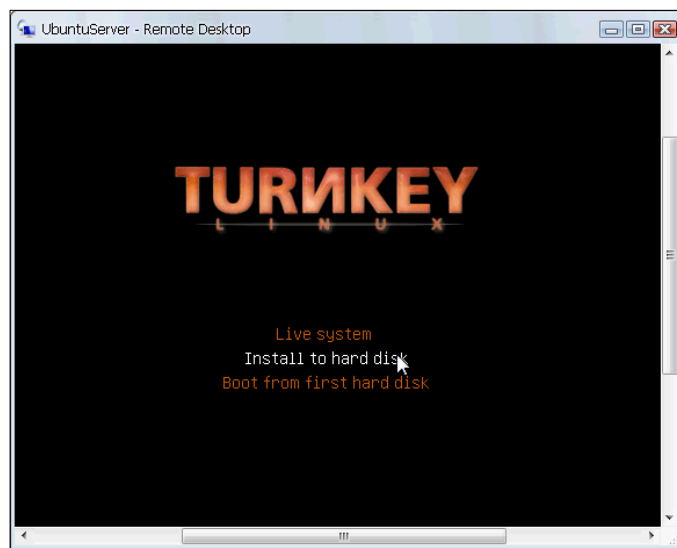
## Time for action – running your very own remote virtual LAMP server

In this exercise, I'll show you how to get the Turnkey Linux LAMP appliance and install it as a remote virtual machine in your Ubuntu headless VirtualBox server.

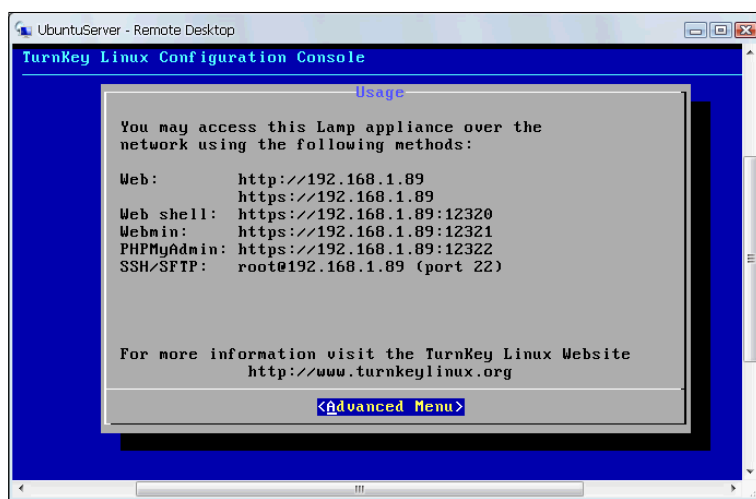
1. First you need to download the Turnkey Linux LAMP ISO image from <http://www.turnkeylinux.org/lamp>, and upload it to your headless VirtualBox server, as in the *Time for action – uploading a guest ISO image to your headless server* section of this chapter.



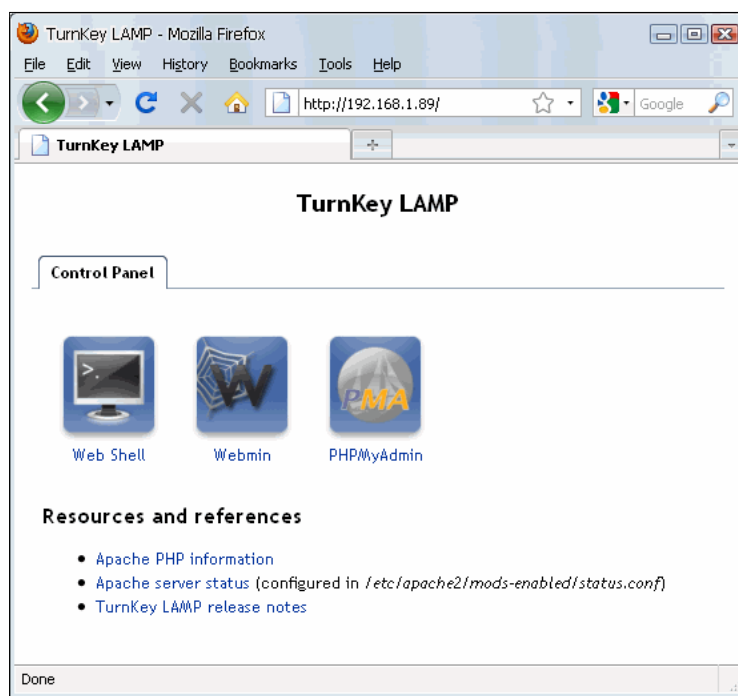
2. Once the Turnkey LAMP ISO image is uploaded, you need to create a new virtual machine named TurnkeyLAMPVB. Open a PuTTY terminal window to connect to your Ubuntu headless server, and type `VBoxManage createvm --name "TurnkeyLAMPVB" --register` followed by *Enter* to create your new virtual machine.
3. Now type `VBoxManage modifyvm "TurnkeyLAMPVB" --ostype "Ubuntu" --memory 384 --boot1 dvd --nic1 bridged --bridgeadapter1 eth0` followed by *Enter* to define the basic parameters for your new virtual machine, and use the bridged networking mode.
4. Next, type `VBoxManage createhd --filename "TurnkeyLAMPVB.vdi" --size 8192 --remember`, and press *Enter* to create the virtual hard disk for your new virtual LAMP server.
5. Type `VBoxManage storagectl "TurnkeyLAMPVB" --name "IDE Controller" --add ide`, and press *Enter* to attach a storage controller to your virtual machine.
6. Type `VBoxManage storageattach "TurnkeyLAMPVB" --storagectl "IDE Controller" --port 0 --device 0 --type hdd --medium "TurnkeyLAMPVB.vdi"` and press *Enter* to attach your virtual hard drive to the IDE storage controller you've just attached to your virtual machine.
7. Now type `VBoxManage storageattach "TurnkeyLAMPVB" --storagectl "IDE Controller" --port 0 --device 1 --type dvddrive --medium /home/aromero/turnkey-lamp-2009.10-hardy-x86.iso`, and hit *Enter* to attach the TurnkeyLAMP ISO image you uploaded previously to the IDE storage controller in your virtual machine. Don't forget to replace the `/home/aromero` part with your home directory, where you downloaded the Turnkey LAMP ISO image.
8. Now open the RDP viewer window, and click on the **Connect** button. The **Do you trust this remote connection?** dialog will appear next. Go to the PuTTY terminal window, and type `VBoxHeadless --startvm "TurnkeyLAMPVB"` followed by *Enter*. VirtualBox will show the copyright notice along with the **Listening on port 3389** message to indicate that the virtual machine started successfully.
9. Go to the **Remote Desktop Connection** window, and click on the **Connect** button. The next screen will appear in order to indicate that your virtual machine is running:



10. Select the **Install to hard disk** option, and press *Enter*. Follow the on-screen prompts to install your Turnkey LAMP virtual appliance, as you did with the Turnkey Wordpress virtual appliance in Chapter 7, *Using Virtual Appliances*.
11. Once the installation is complete, restart your virtual appliance, and wait until your virtual machine reboots. After the Turnkey Linux screen shows up, select the **Boot from first hard disk** option, and hit *Enter* to continue.
12. Now just wait until the **Turnkey Linux Configuration Console** screens shows up, and take note of your virtual appliance's IP address:



- 13.** Open a web browser window in your Windows remote PC, and type `http://youripaddress` to enter the Turnkey LAMP control panel—don't forget to replace `youripaddress` with the IP address of your Turnkey LAMP virtual appliance:



- 14.** Your Turnkey LAMP remote virtual server is ready for some action!

## ***What just happened?***

The Turnkey LAMP remote virtual server is ready to roll, and you can access it as if it were a real PC connected to your LAN. Now you can start developing the next PHP killer web application! This exercise was pretty simple, and it shows you how easy it is to set up a remote virtual server in just a few minutes, thanks to the VirtualBox headless server and the preconfigured virtual appliances available in the Turnkey Linux website.

In step 3, we used the `--nic1 bridged --bridgeadapter1 eth0` parameters from the `VBoxManage modifyvm` command to indicate that we'd be using the bridged networking mode in the first network interface (`eth0`) of your Turnkey LAMP virtual appliance. Aside from that, you could say the process was practically identical to the one we used when creating the UbuntuVB remote virtual machine.

## Have a go hero – exploring your new Turnkey LAMP remote virtual server

Try the Webshell, the Webmin, and the PHPMyAdmin interfaces, and see how easy it is to communicate with your new web server through the Turnkey LAMP control panel tools. If you're into PHP programming, go ahead and start creating your first virtual website!

## Have a go hero – using port-forwarding instead of the bridged mode

Now that you know how to download Turnkey Linux ISO images to create remote virtual appliances, grab the Turnkey Drupal ISO image, and create a new remote virtual machine. Name it TurnkeyDrupalVB, and, instead of the bridged networking mode, use the exercise in the *Using port-forwarding with the NAT mode* section from Chapter 6, *Networking with Virtual Machines*, to use port-forwarding. This way, you can communicate with your Drupal remote virtual server from the other PCs in your LAN.

## Have a go hero – using two remote virtual servers at the same time

Use the `--vrdport` parameter from the `VBoxHeadless` command to start your Turnkey LAMP server and your Turnkey Drupal server at the same time, and see if you can visit both websites from other PCs in your LAN.

And remember: if you get stuck, you can always email me at [questions@packtpub.com](mailto:questions@packtpub.com). I'll be glad to help!

## Pop quiz – remote virtual machines and alternative frontends

1. Your boss was completely amazed when you showed him what VirtualBox can do! Now he wants you to install a Windows XP virtual machine on three PCs for the administrative staff. They all use Ubuntu desktops but need to use MS Office applications such as Word and PowerPoint occasionally. The best thing to do would be:
  - a. Use the VirtualBox main GUI to run each virtual machine.
  - b. Show the administrative staff how to use VBoxManage to run their virtual machines.
  - c. Create a desktop shortcut on each PC so that the administrative staff can use the VBoxSDL interface to run their virtual machines on.
2. What is the command-line interface needed to run a remote virtual machine named "UbuntuVB" on a headless server?
  - a. `VBoxManage createvm --name "UbuntuVB"`
  - b. `VBoxSDL --startvm "UbuntuVB"`
  - c. `VBoxHeadless --startvm "UbuntuVB"`

3. How would you add more RAM to a virtual machine named "WindowsXP" if the original amount was 384 MB and you needed to upgrade it to 512MB.
  - a. `VBoxSDL --modifyvm "WindowsXP" --ram 512`
  - b. `VBoxManage modifyvm "WindowsXP" --memory 512`
  - c. `VBoxHeadless --modifyvm "UbuntuVB" --ram 384`
4. Could you use the VBoxManage command-line interface to start a virtual machine automatically each time you turned on your Windows XP desktop PC?
  - a. Yes, by creating a script that would execute automatically every time you turned on the Windows desktop PC.
  - b. No, that would be impossible.
  - c. Geez, I don't know!
5. How would you know how many virtual machines are hosted in your Ubuntu headless server? Which command would you use?
  - a. `VBoxManage list vms`
  - b. `VBoxManage list showvminfo`
  - c. `VBoxHeadless list vms`

## Summary

In this chapter, we saw all the basics about the alternative frontends available in VirtualBox and how you can use them to run your virtual machines based on your specific needs. We also saw a detailed tutorial on how to set up your own Ubuntu headless server and use it to run a remote virtual machine from a Windows XP desktop PC.

Specifically, we covered:

- ◆ VBoxManage, the alternative frontend you can use to control and run your virtual machines from your host's command-line interface.
- ◆ VBoxDSL, a simple graphical frontend you can use instead of the classic VirtualBox frontend to control your virtual machines.
- ◆ VBoxHeadless, the VRDP server interface built into VirtualBox and how to setup a 'headless' VirtualBox server running on top of the Ubuntu Linux Server operating system.
- ◆ How to create, manage, and run virtual machines in your headless server from a remote PC.

I hope you enjoyed this chapter, along with the rest of the book. Now you have all the basic knowledge to start tackling some more advanced material with your virtual machines and VirtualBox. And don't forget to keep experimenting with your Ubuntu headless server, too!



# A

## Using Snapshots

Now it's time for you to learn how to do some time-travelling with VirtualBox! I'm not kidding around! Seriously, I'm going to teach you about using snapshots with your VMs, so you can go back and forth between different points in time in your virtual machines.

In this appendix we will learn how to:

- ◆ Save your virtual machine state through snapshots
- ◆ Revert changes to your virtual machines
- ◆ Create alternate realities so you can test different software configurations in your virtual machines

Are you ready? Let's dive in...

### Reverting changes to your virtual machine

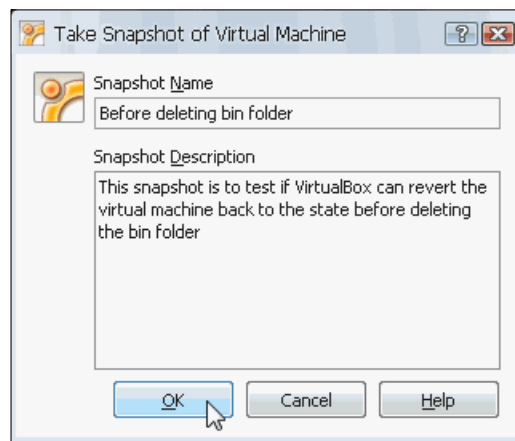
Today you'll finally have the opportunity to get even with your boss! The Snapshots feature in VirtualBox will let you scare the living daylights out of him! How, you may ask? Well, you can take a snapshot of your working virtual machine, then delete some important parts of your guest operating system and, when your boss realizes that he has lost all the work he's done for the last few days on his virtual machine, you'll come to the rescue by reverting the change to his virtual machine and restoring his sanity! And now let's see how to do that...



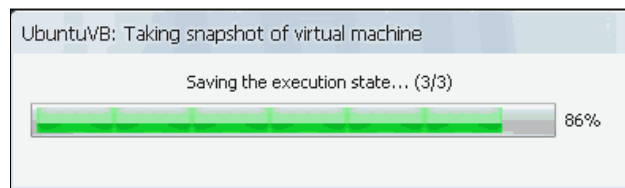
## Time for action – saving the state of your Ubuntu virtual machine by taking a snapshot

In this exercise, I'll show you how to use the Snapshots feature to save the state of a Linux Ubuntu virtual machine by taking a snapshot, deleting some important system files so it can't boot anymore, and then reverting those changes back to the state before deleting those system files so it can boot again, like nothing ever happened!

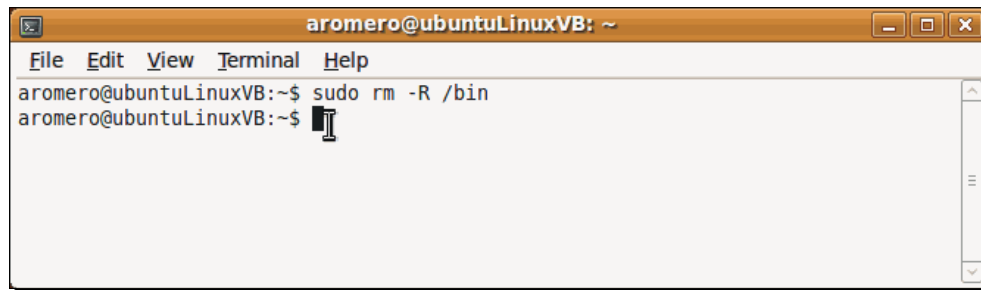
1. Start your UbuntuVB virtual machine and select **Machine | Take Snapshot** from the VirtualBox menu.
2. The **Take Snapshot of Virtual Machine** dialog will show up. Type **Before deleting bin folder** in the **Snapshot Name** field, and **This snapshot is to test if VirtualBox can revert the virtual machine back to the state before deleting the bin folder** in the **Snapshot Description** field, as shown below:



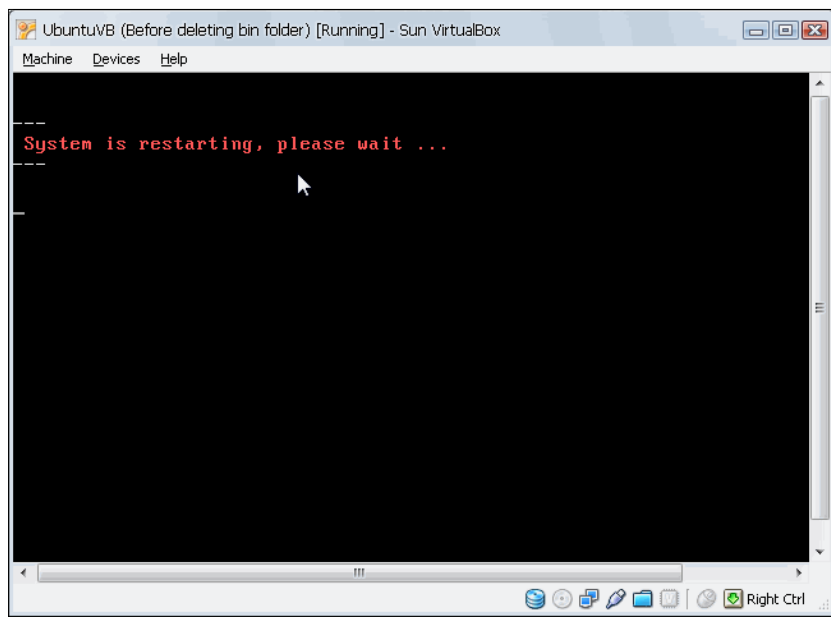
3. Click on **OK** to continue. VirtualBox will start to take the snapshot and the following dialog will show up during the process:



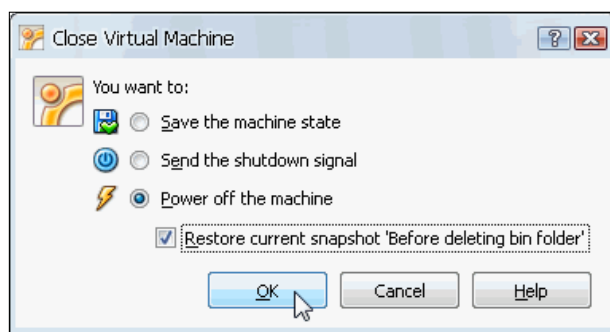
4. Once VirtualBox finishes taking a snapshot of your Ubuntu virtual machine, open a terminal window (select **Applications | Accessories | Terminal** from the Ubuntu menu bar), type `sudo rm -rf /bin` to remove the bin folder from your Ubuntu virtual machine and hit *Enter* (if Ubuntu asks for the **sudo** password, type it so you can remove the folder):



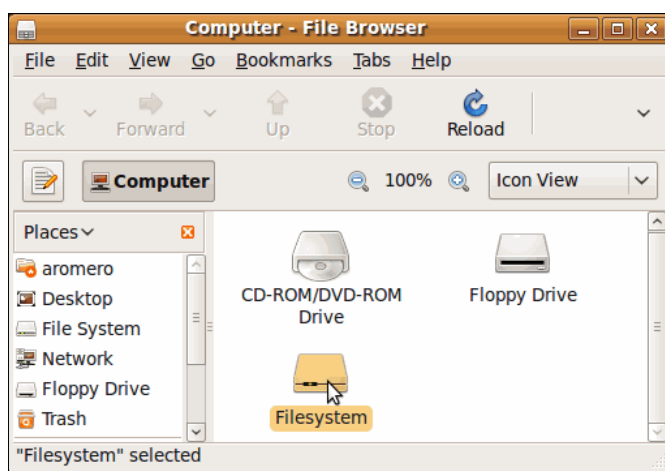
5. Type `exit`, and then press *Enter* to close the terminal window and restart your Ubuntu virtual machine.
6. The Ubuntu virtual machine will not be able to restart and a **System is restarting, please wait ...** message will show up, or it will hang indefinitely, depending on your Ubuntu guest version:



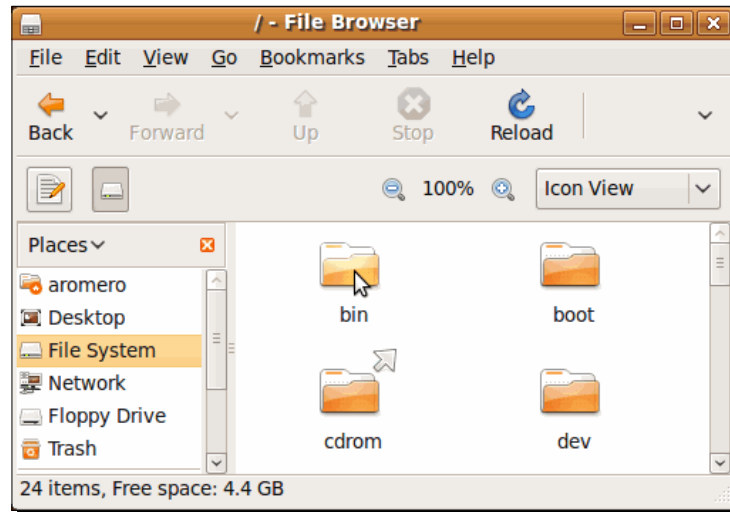
7. Select **Machine | Close** from your virtual machine's main menu to continue.
8. The **Close Virtual Machine** dialog will appear next. Select the **Power off the machine** option, enable the **Restore current snapshot 'Before deleting bin folder'** check box, and click on **OK** to shut down your virtual machine:



9. Wait until VirtualBox finishes restoring the current snapshot. You'll be taken to the VirtualBox main screen. Select your **UbuntuVB** virtual machine and then select the **Snapshots(1)** tab from the right panel. The **Before deleting bin folder** line will appear between parenthesis to indicate your VM will start to the state it had before deleting the `bin` folder. Click on **Start** to continue.
10. After a few seconds, VirtualBox will revert your virtual machine to the state before deleting the `bin` folder. Select **Places | Computer** from Ubuntu's main menu to continue.
11. The **Computer – File Browser** window will open up. Double-click on the **Filesystem** icon to browse in your Ubuntu virtual machine's filesystem:



**12.** Voila! As you can see in the following screenshot, the `bin` folder is intact:



**13.** Close your UbuntuVB virtual machine for the next exercise.

### ***What just happened?***

Whew! I bet you got nervous during the last steps of the exercise, didn't you? How did you like that little magic trick with the snapshot? You can take a snapshot of your virtual machine right after a clean operating system installation and then do whatever you want with your virtual machine because if you don't like what you see, just go back to your saved snapshot and presto! I'm pretty sure your boss will knock his socks off if you dare to perform this little magic trick on his virtual machine! Too bad you can't use snapshots to revert some big mistakes in real life...

### **Have a go hero – using several Snapshots in your virtual machines**

Practice taking several snapshots of your virtual machines, every time you install a new software application, for example. You can use snapshots as a backup solution for your virtual machine's data, too.

### Pop quiz – using the folder sharing feature

1. This morning your boss came crying to your office because some file he downloaded in his Windows XP virtual machine had a virus and now he can't start it up. You:
  - a. Go to your boss's office, select the snapshot you took last night from his Windows virtual machine and his problem is solved.
  - b. Tell him that all work is lost, that you'll need to delete his entire virtual machine and reinstall it again.
  - c. Tell him to go and jump out of a bridge.
2. Snapshots are useful when you need to:
  - a. Enhance mouse pointer integration in your virtual machine.
  - b. Install some experimental software and you don't want to lose important information in your virtual machine.
  - c. Play 3D games on your Windows virtual machine.

## Creating alternate realities in your virtual machines

Suppose you need to test certain software package, but aren't sure if you will want to leave it installed in your computer because it could cause conflict with other similar software packages. For example, let's say you use the Mozilla Firefox web browser in your daily chores, but want to try out the new Google Chrome web browser. You could create an alternate reality where you'd uninstall Mozilla Firefox and install Google Chrome and, if you don't like what you see, you can revert those changes to make Google Chrome disappear and Mozilla Firefox reappear as if nothing ever happened!

### Time for action – using branching snapshots in your VMs

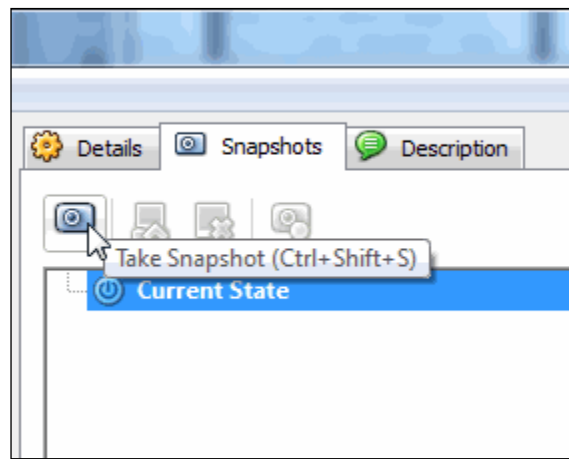
In this exercise, you'll learn how to use branching snapshots so you can create alternate realities in your virtual machines.

1. Go to your UbuntuVB virtual machine settings page and select the **Snapshots (1)** tab. Then, select the **Before deleting bin folder** snapshot and click on the **Delete snapshot** button or hit *Ctrl+Shift+D* to delete the snapshot.



When you use the **Delete snapshot** command, you're actually deleting the point in time when the snapshot was taken. So, if you accidentally forget to restore your virtual machine's state to the **Before deleting bin folder** snapshot before actually deleting it, the **Delete snapshot** command will erase that snapshot and you'll lose your `bin` folder forever! That's why you need to be very careful when deleting snapshots because you could lose important data or leave your virtual machine completely useless!

2. A **VirtualBox – Question** dialog will pop up, asking if you really want to delete the snapshot. Click on **Delete** to continue.
3. Now select the **Current State** item inside the **Snapshots** panel and click on the **Take Snapshot** button:

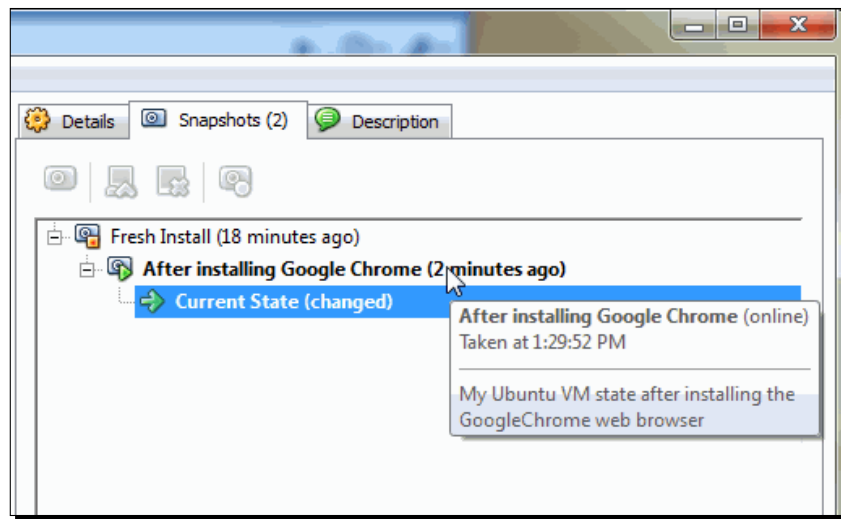


4. The **Take Snapshot of Virtual Machine** dialog will pop up next. Type **Fresh Install** in the **Snapshot Name** field, **Fresh installation of Ubuntu Karmic** in the **Snapshot Description** field, and click on **OK** to continue.
5. The **Fresh Install** snapshot will appear above the **Current State** item. Now start your **UbuntuVB** virtual machine and log in. Once the Ubuntu desktop shows up, open a web browser window, go to <http://www.google.com/chrome>, and follow the instructions to download and install the Google Chrome web browser.

6. Wait for the installation process to complete and then select **Applications | Internet | Google Chrome** and go to <http://www.virtualbox.org> to verify the Google Chrome web browser is working in your virtual machine:



7. Now select **Machine | Take Snapshot** from your virtual machine's menu. The **Take Snapshot of Virtual Machine** dialog will pop up next. Type **After installing Google Chrome** in the **Snapshot Name** field, **My Ubuntu VM state after installing the Google Chrome web browser** in the **Snapshot Description** field, and click **OK** to save the new snapshot.
8. If you go to the Snapshots tab of your UbuntuVB virtual machine, you'll see that the new snapshot is right below the Fresh Install snapshot you took before:



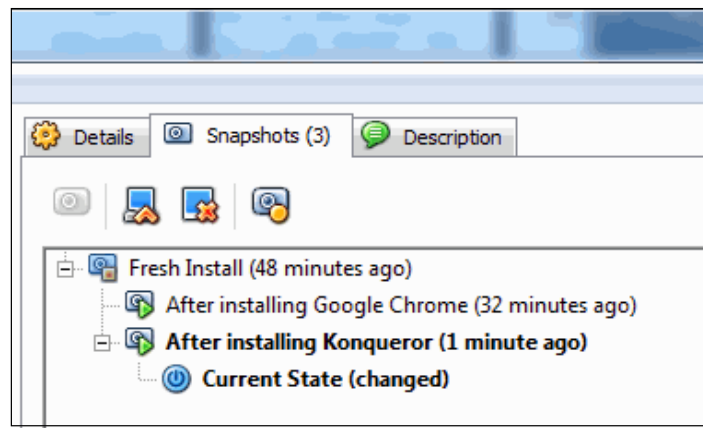
9. Go back to your UbuntuVB virtual machine's main screen and shut it down without closing the Google Chrome web browser.
10. Once your UbuntuVB VM is shut down, select the **Fresh Install** snapshot in the **Snapshots** panel and click on the **Restore Snapshot** button or hit *Ctrl+Shift+R* to revert your virtual machine's state back to that snapshot.
11. A **VirtualBox – Question** dialog will pop up to ask if you're sure you want to restore the **Fresh Install** snapshot. Click on **Restore** to continue.
12. Now start your **UbuntuVB** virtual machine again. After logging in you can check the **Applications | Internet** menu to verify the Google Chrome web browser isn't installed in your virtual machine, and the **(Fresh Install)** message will appear in your virtual machine's title bar to indicate your VM is in the state it was before you installed Google Chrome.
13. Now open a terminal window (**Applications | Accessories | Terminal**), type `sudo apt-get install konqueror`, press *Enter*, and follow the on-screen instructions to install the Konqueror web browser in your UbuntuVB virtual machine.



14. Once the installation process finishes, exit the terminal window, select **Applications | Internet | Konqueror** to open the Konqueror web browser, and go to `http://www.virtualbox.org` to verify it's working:



15. Select **Machine | Take Snapshot** from your virtual machine's main menu, or hit `Host + S` to take a snapshot of your current configuration. The **Take Snapshot of Virtual Machine** dialog will pop up next. Type `After installing Konqueror` in the **Snapshot Name** field, `My Ubuntu VM state after installing the Konqueror web browser` in the **Snapshot Description** field and click on **OK** to continue.
16. Wait until the snapshot is fully taken and then shut down your virtual machine without closing the Konqueror web browser window. In the **Snapshot** panel you will now see three snapshots:



- 17.** The **After Installing Konqueror** snapshot will be shown in bold to indicate your virtual machine's current state derives from it. Now select the **After installing Google Chrome** snapshot and click on the **Restore Snapshot** button to revert your virtual machine to that place in time where you installed Google Chrome.
- 18.** The **VirtualBox – Question** dialog will pop up to ask if you're sure you want to restore that snapshot. Click on **Restore** to continue.
- 19.** Start your **UbuntuVB** virtual machine and log in to confirm the Konqueror web browser is gone and the Google Chrome web browser is shown on your virtual machine's desktop. Now you can go back and forth in time with your virtual machines!

### ***What just happened?***

Cool! You'll be able to do all those cool time-travelling tricks from the Back to the Future film series! Well, maybe not all of them, but I bet you didn't think it could be possible to try out different software configurations and then revert all changes as if nothing ever happened, right?

This was a lengthy exercise, but it's worth all the hassle. In steps 7 and 8 you created a snapshot after installing the Google Chrome web browser. If you look at the screenshot from step 8, you'll notice the **After installing Google Chrome** snapshot is a derivation from the **Fresh Install** snapshot, and if you look at the screenshot from step 16 you'll notice that the **After installing Konqueror** snapshot also derives from the **Fresh Install** snapshot.

Both snapshots are completely independent from each other. This effectively creates two time paths for your virtual machine, and if you take more snapshots on either path, you can go back and forth specific points of time at your leisure!



You can think of a snapshot as a way to capture a specific point in time from your virtual machine.

In steps 17 and 18 you restored the **After installing Google Chrome** snapshot to go back to the point in time where you installed Google Chrome. From here you can install some other software package or make some changes to your virtual machine configuration, and then take another snapshot to preserve your new point in time, so later you can go back to the point where you installed Google Chrome, or to any other point in time preserved with a snapshot.



With the branching feature you're able to create alternate realities for your virtual machines.

If you take a snapshot when your virtual machine is running, its memory state will also be preserved. That explains why on step 17, when you restored the **After installing Google Chrome** snapshot, the Google Chrome window was still open in your Ubuntu desktop.

And have you wondered how many snapshots can you take on a single virtual machine? Well, there isn't any limit imposed by VirtualBox, so practically you can take unlimited snapshots, but you need to take into account that each snapshot requires certain disk space in your host, so the only limitation is the space available on your host's hard disks.

### Have a go hero – creating more alternate realities for your VMs

Practice with your other virtual machines, taking several snapshots and using the branching feature to create alternate realities. For example, you could install Microsoft Office in a Windows XP/Vista/7 virtual machine, and then create another alternate reality where you can install OpenOffice instead.

### Pop quiz – creating alternate realities for your VMs

1. You need to test two of the most popular IDEs for developing Java programs: NetBeans and Eclipse. The best way to do it would be to:
  - a. Get two machines with the same RAM, processor and operating system, and then install NetBeans on one machine and Eclipse on the other.
  - b. Use snapshots and VirtualBox to create a virtual machine with two alternate realities: one with NetBeans installed and the other with Eclipse installed.
  - c. Flip a coin to decide which IDE you're going to use.

2. After several days of testing your Eclipse and NetBeans installations, you decide NetBeans is the most appropriate IDE for you. What would you need to do to uninstall Eclipse from your virtual machine?
  - a. Delete the snapshot where you installed Eclipse, and keep only the NetBeans-related snapshots.
  - b. You'll need to erase everything, reinstall VirtualBox, and create a new virtual machine where you can install NetBeans again.
  - c. You need to reformat your host's hard disk, and you'll need to delete all your valuable data!

## Summary

I hope you liked the exercises in this appendix. Snapshots are a very useful feature when working with virtual machines, and now you will be able to use them in your own VMs!

Specifically, we covered how to:

- ◆ Save your virtual machine state through the snapshots feature
- ◆ Revert changes to your virtual machine via a snapshot
- ◆ Create alternate realities, where you can follow two or more completely independent time paths in your virtual machines



# B

## Pop Quiz Answers

### Chapter 1

#### Doing the thing

|   |       |
|---|-------|
| 1 | b     |
| 2 | b     |
| 3 | a & c |

### Chapter 2

#### Creating virtual machines

|   |   |
|---|---|
| 1 | b |
| 2 | a |
| 3 | b |
| 4 | a |

## Configuring basic settings on your VMs

|   |   |
|---|---|
| 1 | b |
| 2 | b |
| 3 | c |

## Using the auto capture and host key features

|   |   |
|---|---|
| 1 | b |
| 2 | c |
| 3 | b |

## Running your Ubuntu Linux VM

|   |   |
|---|---|
| 1 | b |
| 2 | c |
| 3 | b |

## Your first VM

|   |   |
|---|---|
| 1 | b |
| 2 | c |
| 3 | b |
| 4 | e |
| 5 | f |

## Chapter 3

### Creating virtual machines

---

|   |   |
|---|---|
| 1 | a |
| 2 | b |
| 3 | c |
| 4 | a |

---

### Using the auto capture and host key features

---

|   |   |
|---|---|
| 1 | c |
| 2 | a |
| 3 | b |

---

### Running your Windows 7 VM

---

|   |   |
|---|---|
| 1 | a |
| 2 | a |
| 3 | c |

---

### Your first VM

---

|   |   |
|---|---|
| 1 | a |
| 2 | c |
| 3 | b |
| 4 | a |

---



## Chapter 4

### Guest Additions

---

|   |   |
|---|---|
| 1 | c |
| 2 | b |
| 3 | b |

---

### Using the fullscreen feature

---

|   |   |
|---|---|
| 1 | c |
| 2 | b |
| 3 | c |

---

### Using the folder sharing feature

---

|   |   |
|---|---|
| 1 | b |
| 2 | b |

---

---

## Chapter 5

### Using virtual disks in VirtualBox

|   |   |
|---|---|
| 1 | c |
| 2 | a |
| 3 | a |

### Creating additional virtual disk images

|   |   |
|---|---|
| 1 | c |
| 2 | a |

### Virtual storage

|   |   |
|---|---|
| 1 | a |
| 2 | b |

### Using different storage controller types

|   |   |
|---|---|
| 1 | b |
| 2 | a |

### Storing data in VirtualBox

|   |   |
|---|---|
| 1 | c |
| 2 | b |

## Chapter 6

### Working with the default network adapter types

---

|   |   |
|---|---|
| 1 | a |
| 2 | c |
| 3 | b |

---

### Using the bridged networking mode

---

|   |   |
|---|---|
| 1 | b |
| 2 | c |
| 3 | c |

---

### Using the internal networking mode

---

|   |   |
|---|---|
| 1 | b |
| 2 | a |
| 3 | a |
| 4 | c |

---

### Virtual networking

---

|   |   |
|---|---|
| 1 | b |
| 2 | c |
| 3 | c |

---

---

## Chapter 7

### Importing virtual appliances

|   |   |
|---|---|
| 1 | c |
| 2 | c |
| 3 | c |

### Exporting virtual appliances

|   |   |
|---|---|
| 1 | b |
| 2 | b |
| 3 | b |

### Working with virtual appliances

|   |   |
|---|---|
| 1 | b |
| 2 | a |
| 3 | b |

### Virtual appliances

|   |   |
|---|---|
| 1 | b |
| 2 | a |

## Chapter 8

### Using the VBoxManage interface

---

|   |   |
|---|---|
| 1 | b |
| 2 | c |
| 3 | b |

---

### Setting up your own VirtualBox headless server

---

|   |   |
|---|---|
| 1 | b |
| 2 | a |
| 3 | b |

---

### Enabling FTP and uploading ISO images on your headless server

---

|   |   |
|---|---|
| 1 | b |
| 2 | a |
| 3 | c |

---

### Remote virtual machines and alternative front-ends

---

|   |   |
|---|---|
| 1 | c |
| 2 | c |
| 3 | b |
| 4 | a |
| 5 | a |

---

## **Appendix A**

### **Using the folder sharing feature**

---

|   |   |
|---|---|
| 1 | a |
| 2 | b |

---

### **Creating alternate realities for your VMs**

---

|   |   |
|---|---|
| 1 | b |
| 2 | a |

---



# Index

## Symbols

### 2D hardware acceleration

using, on Windows VM 137

### 3D hardware acceleration

allowing, in virtual machine 131

using, on Windows VM 137

### BSOD 220

--enable parameter 204

--ip parameter 204

--lowerip parameter 204

--netmask parameter 204

--netname parameter 204

--remember option 283

--upperip parameter 204

## A

### alternate realities

creating, in virtual machines 308-313

### apt-get 270

### Auto capture keyboard feature 60

## B

### base memory 81

### BitNami Drupal virtual appliance

accessing 236

DrupalVB - Settings screen 234

using 233-236

### Blue Screen of Death. *See* BSOD

### bridged networking mode

using 193, 194

VM's web server, accessing from  
another VM 191

VM's web server, accessing from host PC 187

## C

### cloning

about 155

disk images, simultaneously 159

Ubuntu Linux hard disk image 156, 158

using, to create multiple VM 155

### Compiz

using, in Ubuntu VM 132-137

### controlvm command

Puppy Network Wizard window 258

using, for saving VM's state 258

using, to pause VM's state 257

using, to resume VM's state 258

## D

### Damn Small Linux. *See* DSL

### default NAT mode

accessing 179-183

connecting to 174

default network adapter types 174

port-forwarding, using 184

testing, on Windows guests 184

### default network adapter types

softwares 174

viewing 175, 176, 177

working with 178

### DHCP 190

### disk controller

choosing 163

different combinations, using 166

IDE 163

IDE drives, using 166-169

SATA 163

SATA drives, using 166-169



- SCSI 163
- disk image files** 82, 140
- dkms** 272
- DKMS** 27
- downloading**
  - Ubuntu Linux Live CD 44, 45
- DSL**
  - creating 28-36
  - testing 36-39
- Dynamic Host Configuration Protocol.** *See* DHCP
- Dynamic Kernel Module Support.** *See* DKMS

## F

- FileZilla**
  - using 278
- First Run Wizard**
  - using, to boot Windows 7 installation disk 84, 85
  - using, with another operating system 86
- first virtual machine, creating**
  - basic Ubuntu Linux VM settings, configuring 51
  - requirements 44
  - Ubuntu Linux Live CD, downloading 44, 45
  - Ubuntu Linux, using 44
  - Ubuntu Linux VM, creating 45
  - Ubuntu Linux VM, installing 53
  - Ubuntu Linux VM, running 62
- folder sharing**
  - about 120
  - between host and guest 120-125
  - between Ubuntu Linux host and Window XP guest 127
  - between Window XP host and OpenSolaris guest 126
  - file formatting characteristics,
    - comparing 125, 126
  - permenant shared folder, creating 126
  - shared folder, creating as regular user 126
- full screen mode**
  - using 117-119

## G

- gateway** 184
- guest** 8
- Guest Additions**
  - about 108
  - installing, for Linux 112-115
  - installing, for OpenSolaris 115, 116
  - installing, for Windows 109, 110, 111
  - installing, on other remote VM 289
  - uses 108
- guest ISO image**
  - uploading 278, 280
- guest port** 185

## H

- hard disk images**
  - expanding 160, 161
  - fixed-size image, creating 161-163
- host** 8
- Host+F key** 118
- Host+Home key**
  - using 118
- Host Key**
  - about 60
  - redefining 60
  - Right Ctrl 60
- Host key + L** 131
- host-only networking mode**
  - testing, on Windows guests 209
  - UbuntuVB and UbuntuVBClone VM,
    - communicating between 206-208
- host port** 185

## I

- IDE**
  - disks, using on Ubuntu Linux VM 169
  - drives, adding to Ubuntu VB virtual machine 166-169
  - using 164
- installing**
  - VirtualBox, on Linux 15
  - VirtualBox, on Windows 9

## **internal networking mode**

- about 199
- communicating, with VMs only 199, 201, 203
- private VM network, creating 199, 201, 203
- testing, on Windows guests 205
- using 205

## **Internet Explorer**

- using, in Windows 7 97

## **ipconfig command 189**

## **K**

### **KUbuntu**

- about 75
- experimenting with 75
- Ubuntu, differentiating 75

## **L**

### **LAMP 214**

### **LAN 173**

### **Linux**

- Guest Additions, installing for 112-115

### **Local Area Network. *See* LAN**

## **M**

### **Microsoft Office**

- using, in Windows 7 98-101

### **Microsoft Office Live**

- using 101

### **mkdir command 126**

### **mount -t vboxsf command 259**

### **Mozilla Firefox**

- Ubuntu Start Page 64
- using, for web browsing 64, 65

### **multiple virtual machines**

- creating, cloning process used 155, 159

## **N**

### **NAT 174**

### **Network Address Translation. *See* NAT**

### **Not Attached mode**

- about 194
- VM, isolating 194, 196

## **O**

### **OpenOffice.org**

- using, in virtual machines 66-69

### **OpenSolaris**

- Guest Additions, installing for 115, 116

### **Open Source. *See* OSE**

### **Open Virtualization Format. *See* OVF**

### **OSE 27**

### **OVF 215**

## **P**

### **Personal Use and Evaluation License. *See* PUEL**

### **portability 226**

### **port-forwarding, default NAT mode**

- about 184
- enabling 184-186
- NAT networking mode, limitations 186
- using, with NAT networking mode 187

### **proftpd, Ubuntu headless server**

- installing 276, 277

### **protocol 185**

### **PUEL**

- about 27
- RDP server 27
- USB over RDP 27
- USB support 27

### **PuppyLinuxVB virtual machine window 257**

### **PuppyLinux VM, VirtualBox® Images**

- downloading 228-231
- guest additions, installing 232
- Puppy Video Wizard screen 231
- Virtual Media Manager window 230

### **PuTTY SSH client**

- using, to remote Ubuntu server access 268, 270

## **R**

### **RAID 163**

### **RAM 81**

### **RDP 27**

### **RDP viewer**

- using, to run VM 284-289

### **Redundant Array of Independent Disks. *See***

### **RAID**

**Remote Display Protocol.** *See* RDP

**remote virtual LAMP server**

- port-forwarding, using 299
- running 295, 296, 297, 298
- setting up 295
- Turnkey LAMP, exploring 299
- two servers, using simultaneously 299

**remote virtual machines**

- audio controller, choosing 291
- audio, enabling 290, 291
- creating, simultaneously 289, 290
- shared folders, using 292

## S

**SATA**

- disk controller, using on VM 164, 165
- disks, using on Ubuntu Linux VM 169
- drives, adding to Ubuntu VB virtual machine 166, 167, 168, 169

**savestate command 259**

**SCSI 163**

**Seamless Windows**

- about 128
- activating, with Ubuntu 128, 129
- activating, with Windows 129, 130

**Send the shutdown signal option 103**

**Snapshots**

- several snapshots, using 307
- Ubuntu virtual machine state, saving 304-306

**sudo command 126**

## T

**Turnkey Linux File Server appliance**

- FileServerVB - Settings window 241
- using 239-247
- Website Dialog window 246

**TurnKey Wordpress virtual appliance**

- accessing 218
- downloading 215
- file content 216
- files 219
- importing 217
- Importing Appliance dialog 217

## U

**Ubuntu 9.10 Karmic Koala**

- using 17

**Ubuntu Desktop installation, on VM**

- Auto capture keyboard feature 60
- details, filling up 57
- Install Ubuntu option 55
- Keyboard layout screen 56
- language, setting 56
- Prepare disk space screen 56
- Ready to Install screen 57
- starting with 54, 55
- steps 55-59

**Ubuntu headless server**

- guest images, uploading 276
- ISO guest image, uploading 277-280
- proftpd, installing 276, 277
- RDP viewer, using 284-289
- remote virtual machine, managing 275
- shared folder, accessing 294
- shared folder, creating 292, 293
- virtual machine, creating 281, 282

**Ubuntu Linux hard disk image**

- cloning 156-159
- converting, to other formats 159
- registering 159

**Ubuntu Linux Live CD**

- downloading 44, 45

**Ubuntu Linux VM**

- basic configuration 51, 52
- IDE, using 169
- information sharing, between VM and host PC 70
- instant messaging 70
- KUbuntu 74
- OpenOffice.org, using 66-69
- running 62, 63
- SATA disks, using 169
- shutting down 71, 72, 73
- Ubuntu One service, using 69
- virtual machine, creating 46-49
- web browsing, Mozilla Firefox used 64, 65

## **Ubuntu Server 8.04 LTS**

- downloading 263-267
- installing 263-267
- Software selection screen 266

**Unique Identity Number.** *See* **UUID**

**UUID** 155

## **V**

### **VBoxHeadless server**

- accessing, from remote PC 268, 270
- VirtualBox installation on Ubuntu server, apt-get used 271, 272, 274

### **VBoxHeadless servers**

- about 262
- setting up 262
- Ubuntu Server 8.04 LTS, setting up 262

### **VBoxManage**

- interface, using 259
- using 252
- using, to start a VM 253-256
- using, with UbuntuVB virtual machine 259

**VBoxManage clonehd command** 159, 226

### **VBoxManage clonevdi command**

- using 159

### **VBoxManage command**

- using 253

**VBoxManage createhd command** 140

**VBoxManage dhcpserver command** 203

**VBoxManage setextradata command** 185, 187

**VBoxManage startvm** 253

### **VBoxSDL**

- interface, experimenting with 261
- using, to run VM 260, 261

### **VDI hard drive**

- hard drive, creating on Windows VM 150
- secondary virtual drive, adding 141-148
- VDI hard drive using 141

### **VHD hard drive**

- using 151
- virtual drive, adding 151-153
- virtual drive image, using on multiple VM 154
- virtual drive image, using on same machine simultaneously 155

### **virtual appliance**

- about 214
- customized virtual appliance, testing 226

customized Wordpress, exporting 221-225

exporting 221

importing 214

LAMP Stack Appliance 214

TurnKey Wordpress, using 215

UbuntuVB virtual machine, exporting as 226

working with 228

### **virtual appliance, working with**

Turnkey Linux File Server appliance, using 239

using, from Bitnami 233

VirtualBox® Images, using 228

### **VirtualBox**

second VM creating, Windows 7 used 78

default network adapter types 174

disk controller types 163

downloading, on Windows 9

dynamically expanding storage 49

feature 251

first virtual machine, creating 44

fixed-size storage 49

Guest Additions 107

installing, on Linux 15

installing, on Windows 9

Linux distributions 114

mini toolbar setting, adjusting 119

networking 173

overview 7

PUEL 27

RAM, assigning 49

Seamless Windows 128

testing 28

virtual disks, using 140

### **VirtualBox® Images**

about 228

PuppyLinux VM, using 228

### **VirtualBox installation**

Linux 15

Windows 9

### **VirtualBox installation, on Linux**

accessing 24

Apply button 22

Apply the following changes? dialog 22

APT line field 17

Configuring virtualbox-3.X dialog 23

experimenting with 28

Import Key dialog 19

- Mark additional required changes?
    - dialog box 21
  - Mark for Installation option 21
  - Origin button 20
  - Register button 26
  - Reload button 20
  - requirements 15, 16
  - Software Sources dialog 16
  - steps 18, 23, 24
  - Sun public key, deleting 20
  - Synaptic Package Manager 16
  - VirtualBox installation, on Ubuntu server**
    - apt-get, using 271-274
  - VirtualBox installation, on Windows**
    - Custom Setup 12
    - Downloads link 10
    - main screen 14
    - platforms 15
    - starting with 9
    - steps 11, 12
    - Sun xVM VirtualBox Setup dialog 11
    - x86/amd64 link 10
  - VirtualBox Remote Desktop Protocol.** *See* VRDP
  - VirtualBox, testing**
    - DSL, creating 28-36
    - DSL, testing 36, 38, 39
    - virtual machines, creating 41
  - virtual disks, VirtualBox**
    - disk image files 140
    - HardDisks 150
    - saving 150
    - using 140
    - VDI format 140
    - VHD format 140
    - VHD hard drive, using 151
    - VMDK format 140
  - virtual file server**
    - exploring 248
  - virtual hard disk storage**
    - dynamically expanding storage 82
    - fixed-size storage 82
  - virtual hard drive**
    - about 82
    - virtual hard driveabout 48
  - virtualization 8**
  - virtual machine**
    - 3D hardware acceleration, allowing 131
    - about 8
    - connecting, without shutting down 197-199
    - disconnecting, without shutting down 197-199
    - isolating, Not Attached used 194, 196
    - running 8
  - virtual machine management, from alternative frontends**
    - VBoxHeadless servers 252, 262
    - VBoxManage 252, 257
    - VBoxSDL 252, 260
    - VirtualBox 252
  - virtual machine, Ubuntu headless server**
    - creating 281, 282
  - VM**
    - branching snapshots, using 308-313
  - VM's web server, accessing from another VM**
    - about 191
    - bridged networking mode, testing 193
    - steps 191-193
  - VM's web server, accessing from host PC**
    - about 187
    - NAT to bridged networking mode conversion 188-191
  - VRDP 252**
  - Virtual Machine** *See also* VM
- ## W
- windowed modes**
    - using 117-119
  - Windows**
    - Guest Additions, installing for 109-111
  - Windows 7**
    - installing, on VM 86-89
  - Windows 7 installation disk**
    - booting, First Run Wizard used 84, 85
  - Windows 7 installation, on VM**
    - Information dialog 86
    - Install Now button 87
    - Select your current location screen 88
    - steps 86-91
  - Windows7VB2 virtual machine**
    - using 83
  - Windows 7 virtual machine**
    - audio, enabling 92-94
    - audio, verifying in Windows 94
    - Base Memory Size setting 81

- creating 79, 80
- dynamically expanding storage, using 82
- installation media, removing from 94
- instant messaging 98
- Microsoft Office, using 98-101
- other web browsers, using 98
- RAM size, determining 81
- Settings button 82
- shutting down 102, 103
- virtual hard drive setting 82
- web browsing, Internet Explorer used 96, 97

## **X**

**Xvesa option 231**



**Thank you for buying  
VirtualBox 3.1: Beginner's Guide**

## **Packt Open Source Project Royalties**

When we sell a book written on an Open Source project, we pay a royalty directly to that project. Therefore by purchasing VirtualBox 3.1: Beginner's Guide, Packt will have given some of the money received to the VirtualBox project.

In the long term, we see ourselves and you—customers and readers of our books—as part of the Open Source ecosystem, providing sustainable revenue for the projects we publish on. Our aim at Packt is to establish publishing royalties as an essential part of the service and support a business model that sustains Open Source.

If you're working with an Open Source project that you would like us to publish on, and subsequently pay royalties to, please get in touch with us.

## **Writing for Packt**

We welcome all inquiries from people who are interested in authoring. Book proposals should be sent to [author@packtpub.com](mailto:author@packtpub.com). If your book idea is still at an early stage and you would like to discuss it first before writing a formal book proposal, contact us; one of our commissioning editors will get in touch with you.

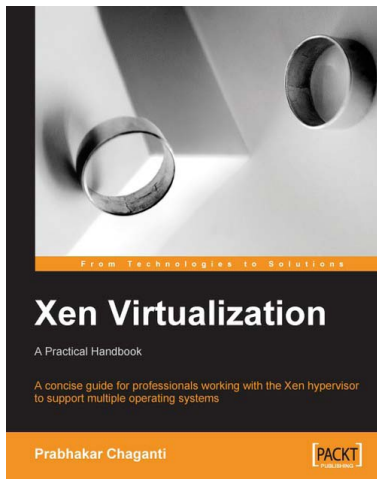
We're not just looking for published authors; if you have strong technical skills but no writing experience, our experienced editors can help you develop a writing career, or simply get some additional reward for your expertise.

## **About Packt Publishing**

Packt, pronounced 'packed', published its first book "Mastering phpMyAdmin for Effective MySQL Management" in April 2004 and subsequently continued to specialize in publishing highly focused books on specific technologies and solutions.

Our books and publications share the experiences of your fellow IT professionals in adapting and customizing today's systems, applications, and frameworks. Our solution-based books give you the knowledge and power to customize the software and technologies you're using to get the job done. Packt books are more specific and less general than the IT books you have seen in the past. Our unique business model allows us to bring you more focused information, giving you more of what you need to know, and less of what you don't.

Packt is a modern, yet unique publishing company, which focuses on producing quality, cutting-edge books for communities of developers, administrators, and newbies alike. For more information, please visit our website: [www.PacktPub.com](http://www.PacktPub.com).



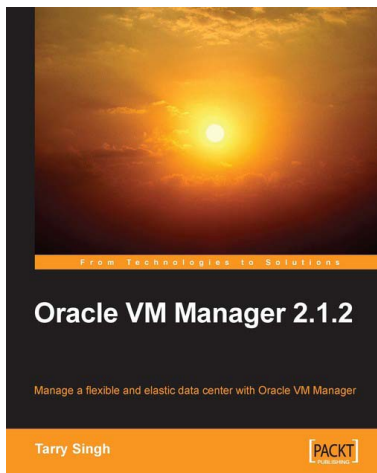
## Xen Virtualization

ISBN: 978-1-847192-48-6

Paperback: 148 pages

A fast and practical guide to supporting multiple operating systems with the Xen hypervisor

1. Installing and configuring Xen
2. Managing and administering Xen servers and virtual machines
3. Setting up networking, storage, and encryption
4. Backup and migration



## Oracle VM Manager 2.1.2

ISBN: 978-1-847197-12-2

Paperback: 244 pages

Creative Ways to Use Moodle for Constructing Online Learning Solutions

1. Learn quickly to install Oracle VM Manager and Oracle VM Servers
2. Learn to manage your Virtual Data Center using Oracle VM Manager
3. Import VMs from the Web, template, repositories, and other VM formats such as VMware
4. Learn powerful Xen Hypervisor utilities such as xm, xentop, and virsh

Please check [www.PacktPub.com](http://www.PacktPub.com) for information on our titles