Professional Expertise Distilled

Getting Started with Oracle VM VirtualBox

Build your own virtual environment from scratch using VirtualBox

Pradyumna Dash

[PACKT] PUBLISHING enterprise professional expertise distilled

Getting Started with Oracle VM VirtualBox

Build your own virtual environment from scratch using VirtualBox

Pradyumna Dash

[PACKT] PUBLISHING enterprise
professional expertise distilled

BIRMINGHAM - MUMBAI

Getting Started with Oracle VM VirtualBox

First published: December 2013

Production Reference: 1051213

Published by Packt Publishing Ltd.
Livery Place
35 Livery Street
Birmingham B3 2PB, UK.

ISBN 978-1-78217-782-1

`www.packtpub.com`

Cover Image by Jarek Blaminsky (`milak6@wp.pl`)

Credits

Author
Pradyumna Dash

Reviewers
Petr Juhaňák
Pranav Salunke
Philip Steiner
Martin Toshev

Acquisition Editors
Taron Pereira
Llewellyn Rozario

Commissioning Editor
Govindan K

Technical Editors
Tanvi Bhatt
Nikhil Potdukhe
Siddhi Rane
Tarunveer Shetty

Project Coordinator
Sherin Padayatty

Copy Editors
Alisha Aranha
Roshni Bannerjee
Karuna Narayanan
Alfida Paiva
Kirti Pai
Lavina Pereira

Proofreader
Lawrence A. Herman

Indexer
Hemangini Bari

Graphics
Yuvraj Mannari

Production Coordinator
Nilesh Bambardekar

Cover Work
Nilesh Bambardekar

About the Author

Pradyumna Dash has been in IT for more than 10 years. He has a Bachelor's degree in IT and Management and holds a number of certifications from vendors such as RedHat, Oracle, SNIA, EXIN, OpenGroup, IBM, Rackspace, AWS, and many more. Currently, he works as a solution architect for a consulting organization in a team specializing in free and open source solutions, virtualization, cloud computing, infrastructure consulting, infrastructure architecture and design, and developing solutions for large data centers. He has been working with Linux since 1998 and is an open source fanatic, and has extensive knowledge on open source virtualization solutions and cloud technologies.

Linux and knowledge sharing are two of his many passions.

He has also published multiple articles for Asia's biggest Linux magazine and is an active member in multiple Linux user groups and communities across the globe.

I would like to dedicate this book to my wife Nilanjana, my family, and my friends for their constant encouragement and, support, and for having the patience with me as I undertook another large project such as writing this book, which decreases the amount of time I spend with them. Speaking about encouragement, I must mention my teammate and manager, who constantly pushed and motivated me to do something beyond my day-to-day work.

About the Reviewers

Pranav Salunke works on OpenStack, especially with the OpenStack training project, and loves to work on open source projects. Currently employed by Aptira as a Cloud Engineer, he is working on various cloud computing and related technologies.

He is a top-notch Shell coder (one of the 46 percent most active Shell users) who loves pushing code. He is a full-time hacker who has contributed to various projects and repositories in eight languages. In particular, he is a pretty serious Shell expert with a surprisingly broad knowledge of Python as well. He has contributed to repositories mainly written in Shell, Python, Java, JavaScript, and Perl. For the latest information on his hacker profile, follow him at `http://osrc.dfm.io/dguitarbite`.

I would like to acknowledge Delilah for her support and positive attitude towards life.

Martin Toshev is a software engineer by profession and hobby. He has been working on a number of large enterprise projects for different companies for the last five years using mostly the Java technology stack. His has experience in the areas of social networking and J2EE middleware and has been an active user of different types of hypervisors, among which is VirtualBox.

Table of Contents

Preface

Oracle VM VirtualBox is an open source, cross-platform virtualization software package for x86 and AMD64- or Intel 64-based computers from Oracle Corporation. It was developed by innotek GmbH and then taken over by Sun Microsystems in the year 2008. Now developed by Oracle, this product was formerly known as Sun VirtualBox. This book is specially designed to quickly help you get up to speed with VirtualBox and give you confidence and an understanding of virtualization. It covers a wide range of topics that will help you install, configure, and manage VirtualBox. It will also help in installing Oracle Enterprise Linux as a guest VM with a lot of advanced features of VirtualBox.

This book gives you clear step-by-step instructions to install and use VirtualBox on your Fedora desktop. These are also applicable for Red Hat, CentOS, and Oracle Enterprise Linux. It will also instruct you on how to run your guest VM on top of VirtualBox. This book is full of concepts and practical examples, and lists advanced features of VirtualBox that enable you to use it in a very simple way.

What this book covers

Chapter 1, *Introduction to Virtualization*, introduces you to the basics of virtualization. It discusses VirtualBox features, the high-level architecture of VirtualBox, and reasons to use virtualization.

Chapter 2, *Installation of Oracle VM VirtualBox on Linux*, teaches you how to install and configure VirtualBox on Linux desktops. It also provides detailed steps to start, update, and remove VirtualBox from Linux OSs.

Chapter 3, *VM Creation on VirtualBox*, teaches you how to create, start, stop, remove, and clone guest virtual machines, and also discusses some advanced and cool features such as snapshot, Clone, and VM groups.

Chapter 4, *Installing OEL*, teaches you the steps to install Oracle Enterprise Linux 6.0 as a guest OS and also talks about how to share a folder in both Windows and Linux. It also talks about memory management techniques in VirtualBox.

Chapter 5, *Virtual Networking*, provides detailed information on different networking options available in VirtualBox and how they work.

Chapter 6, *Virtual storage*, provides detailed concepts on different virtual storage options available in VirtualBox and how they work.

What you need for this book

To use this book, you will need to have access to a desktop or laptop installed with Fedora / CentOS / RedHat Enterprise Linux / Oracle Enterprise Linux. You also need to download VirtualBox 4.2 and need access to Enterprise Linux 6.0 ISO, as the methods presented in this book detail steps for installing Oracle Enterprise Linux 6.0 as the guest VM.

Who this book is for

The book is aimed at desktop users, system administrators, technical architects, and virtualization enthusiasts who want to use open source based, enterprise-ready, powerful virtualization applications or want to move away from commercial products they currently use. Knowledge of the Linux environment is expected; however, prior knowledge of or experience with VirtualBox or virtualization is not required but will be beneficial.

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:

"`VBoxManage` command can also be used."

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

```
VBoxManage modifyvm "VM name" --natpf1 "Testssh,tcp,1234 ,22"
```

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: "If everything goes fine then the **Oracle VM VirtualBox Manager** screen appears."

[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 topic that you have expertise in and you are interested in either writing or contributing to a book, 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.

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/submit-errata`, selecting your book, clicking on the **errata submission form** 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
Introduction to Virtualization

This chapter will introduce you to the basics of virtualization. It discusses VirtualBox features, the high-level architecture of VirtualBox, and why you would want to use virtualization.

The adoption of virtualization technology has increased considerably in the past few years. However, it's not a new concept and has been around for some time now. This section provides the details of virtualization technology and concepts around it. This chapter also highlights why it's the talk of the town and why organizations are embracing virtualization.

This technology provides a way to separate physical hardware (computer) and software (OS and applications) by emulating hardware using software. Also, it enables a computing environment to run multiple independent systems at the same time.

Today, data centers use virtualization to make an abstraction of physical hardware. Virtualization technology helps to create large pools of logical resources consisting of CPUs, memory, disks, storage, and so on, and offers those resources to users or customers in the form of scalable and consolidated virtual machines.

Hypervisor

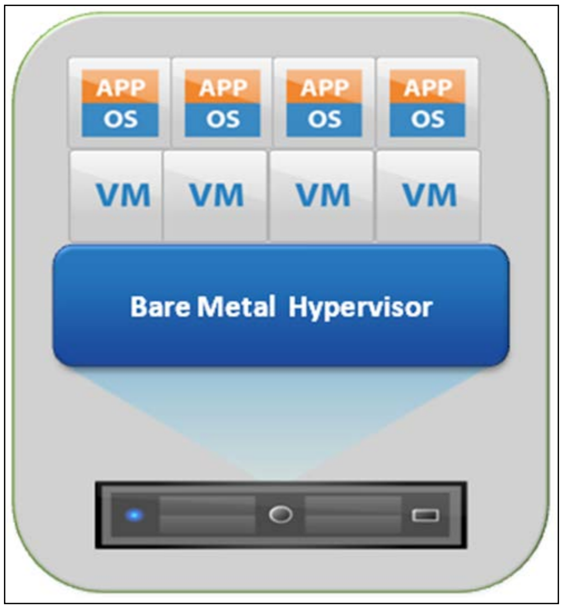
A hypervisor is also known as a virtual machine monitor (VMM). It is a piece of software that enables a user to simultaneously run one or more operating systems, which are known as virtual machines (VM).

Each VM is called a guest system or guest OS, and the physical server on which such a guest OS runs is known as a host system or host OS.

Hypervisors are divided into Type 1 and Type 2.

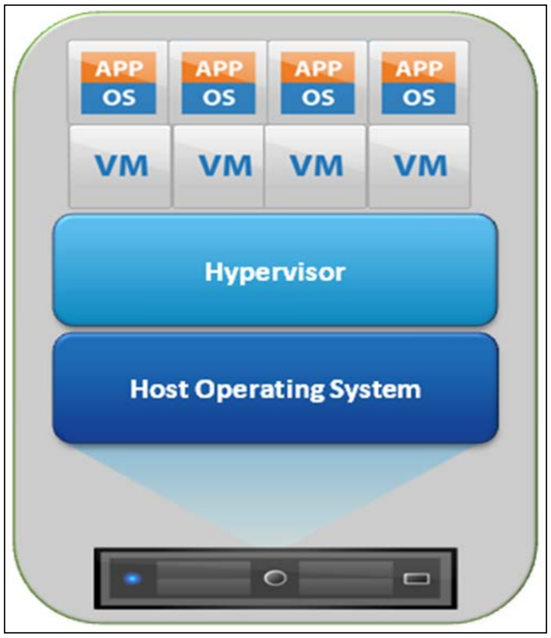
- **Type 1**: This is also known as a bare metal or native hypervisor. This type of hypervisor runs directly on hardware. Examples of bare metal hypervisors are the Oracle VM server, Microsoft Hyper-V, VMware ESX/ESXi, and RHEV.

  As shown in the following screenshot, a Type 1 hypervisor runs on the host's system without a host OS:



- **Type 2**: This is also known as a hosted hypervisor. This type of hypervisor runs on a host OS. In other words, a host OS is installed on top of the host system and the hypervisor runs on top of it. Examples of the hosted hypervisor are VirtualBox, Microsoft Virtual PC, VMware Server and Workstation, and parallels.

  As shown in the following screenshot, the Type 2 hypervisor needs a host OS installed on the host system and a hypervisor runs on top of it:



Reasons to use Virtualization

There are a lot of good and compelling reasons for organizations to opt for virtualization. A few of them are as follows:

- **Server consolidation**: Virtualization significantly reduces capital expenditures (CAPEX), hardware, power, cooling costs, and so on. If virtualization is not used, you will need a lot of servers to serve different workloads, which in turn consumes a lot of electricity, hardware cost, cooling, space, and so on. So, instead of running multiple workloads/ applications on a individual physical server, virtualization helps to consolidate all the workloads to multiple virtual machines running on fewer hardware/servers.
- **Testing and development**: Virtualization allows you to easily build a test environment and ensure it's operating on an isolated network rapidly. Severe crashes that required hours of reinstallation will take a few seconds by simply copying a virtual image to build a new server.

- **Reduce datacenter footprint**: Server consolidation with virtualization will reduce the overall footprint of the entire data center. Virtualization guarantees fewer servers, a smaller number of racks, networking components, and cabling needs, all of which translates into less data center floor space required.
- **Faster server provisioning**: Virtualization helps faster server provisioning and deployment. This can be done with the help of cloning, and templates or existing virtual machines can be used to get the server up and running instantaneously.
- **Reduce power and cooling expenditure**: Consolidating workloads or physical servers to virtual machines ensures fewer physical servers, which reduces power and cooling costs in the data center.

Apart from the preceding features—such as maximum uptime, easy workload migration, self-healing in nature, scalability, and self-service management—disaster recovery features motivate organizations to adopt virtualization.

Oracle VM VirtualBox

VirtualBox is a Type 2 hypervisor. It is open source, cross-platform, and high performance in nature. It was formerly developed by Sun Microsystems as Sun VirtualBox and now it's an Oracle product freely available as open-source software under the terms of the GNU General Public License (GNU GPL) Version 2 licensing. As this is cross-platform, it can run on any modern desktop operating system whether it is Linux, Windows, Mac, or Solaris.

Apart from the fact that VirtualBox is efficient, robust, cross-platform, and high performance in nature, it is also free, which makes this product compelling. Also, it is the only open-source hypervisor of its kind.

It has many components, such as a hypervisor for the host platform, an API and SDK for managing guest virtual machines, and a command-line tool and GUI that can manage VM guests, and some built-in features such as Remote Desktop Protocol (RDP) that can access guest VMs remotely and graphical tools that can manage VM guests as well.

Virtualization components

The following components are needed to install and configure VirtualBox and run a full-blown guest OS on top of it.

- **Host operating system** (**host OS**): As we know, the Type 2 hypervisor needs an OS to be running because it is installed and configured on the top of the host OS. In simple terms, the Type 2 hypervisors run alongside an operating system similar to normal software that you use on day-to-day basis. So, there is one physical computer installed with one host OS and one or more guests running on top of the VirtualBox. The beauty of virtualization is that it is never known to the end user that a single physical hardware is logically divided into multiple logical machines/guest OS. This is because it will always give the impression that the user is assigned a separated/ dedicated physical machine VirtualBox that can be installed and configured on Microsoft Windows, Linux, Solaris, and Mac OS.
- **Guest operating system** (**guest OS**): This is the operating system that is installed and runs inside the virtual machine.
- **Virtual machine** (**VM**): A guest OS is installed and configured in a VM, so a VM includes all the hardware settings (RAM, CPU, hard disk) required for the guest OS and application requirements, which will run on the VM as well.
- **Guest additions**: These are special or add-on software packages that are shipped with VirtualBox that need to be installed on the guest OS. The guest OS in turn increases the VM performance and adds certain extra features.

VirtualBox features

We have already discussed VirtualBox features in the earlier section, but let's dive deep into a few more features:

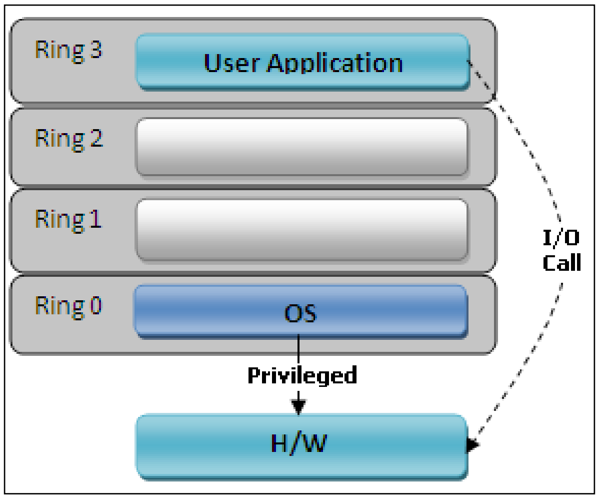
- **Free**: VirtualBox is free and open source.
- **Portability**: VirtualBox can run on both a 32-bit and 64-bit OS based on the Intel x86-64-based processors. VirtualBox is a Type 2 hypervisor. It is functionally identical on all of the host platforms. Also, the same file and image formats can be used across different host operating systems, which means a VM created on one host can easily run on another and, by using Open Virtualization Format (OVF), the guest VMs can be imported and exported when required.

- **No hardware virtualization required**: VirtualBox does not require a hardware virtualization feature. So VirtualBox can even run on older hardware where features such as Intel VT-X or AMD-V are not present.
- **VM groups**: This feature enables the user to organize virtual machines individually as well as collectively. Operations such as start, close, pause, reset, save state, shutdown, power off, and so on can be applied to VM groups like individual VMs.
- **Guest additions**: These are the add-on software packages that are installed on the guest VMs that are certified to run on VirtualBox and help improve the performance and provide additional integration and communication with the host system. There is also a very compelling feature known as a *shared folder* that helps in accessing files from the host OS system within a guest VM.
- **Multigeneration branched snapshots**: VirtualBox supports the save snapshots feature of guest VM state information. So ideally, you can go back and revert the virtual machine to any snapshot and start an alternative VM configuration from there, depending on your requirement. It also allows creating and deleting snapshots while the VM is active and running.
- **Remote machine display**: The VirtualBox Remote Desktop Extension (VRDE) is a feature that helps to access any guest VM that is running remotely. It supports RDP built into Microsoft Windows, but doesn't rely on the inbuilt RDP server for Microsoft Windows; rather it is plugged directly into the virtualization layer.
- **Great hardware support**: VirtualBox supports guest SMP, USB devices, full ACPI support, multiscreen resolution, built-in iSCSI support, and PXE network boot.

How VirtualBox functions

You can run multiple OSs on top of VirtualBox, as guest OSs under one host operating system. Each guest OS can be independently started, stopped, and paused. You can configure each VM independently depending on its role or purpose. For example, you can have a VM running on Linux installed with an Oracle database that will play a role in the database server, whereas another VM can run on Windows hosting an Apache Tomcat installation and running as an application server. You can run the guest VM either on software or hardware assisted virtualization depending on the host hardware supportability. But virtualizing an OS on an x86 processor without any hardware virtualization support is a very tough and complicated task.

Processors provide a mechanism that allows different programs to run at different privilege levels, which ensures that applications should not able to have privileged access to processors, hardware, and so on. This mechanism is known as rings or protection rings. The following diagram shows this architecture:



It starts with **Ring 0**, the lowest and the most privileged level. It interacts directly with the physical hardware, such as CPU and memory. Generally, kernel services are the most privileged services for all OSs; they run on Ring 0 and application/user programs run on Ring 3.

For each individual virtual guest OS, VirtualBox creates a separate process on top of the host OS. All of the guest user code runs natively on Ring 3. Guest kernels run on Ring 1, which helps in protecting the host against failures in the guest.
The VMM scans the Ring 1 code to maintain proper execution and operation of the guest kernel. It does so by either replacing the faulty code path with direct hypervisor calls or executing them in a safe emulator with the help of a code scanner that is unique to each guest. VM emulators comes into the picture only when the VMM is not able to find what the Ring 1 guest code is doing. VirtualBox uses QEMU as an emulator.

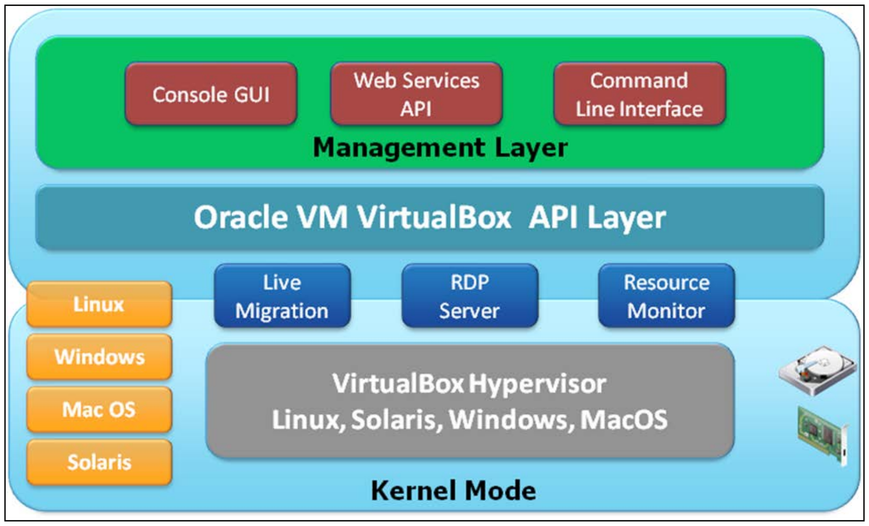
Processor virtualization also has several methods, such as **full virtualization** and **paravirtualization**.

Oracle VM VirtualBox architecture

VirtualBox is modular and flexible by design. The hypervisor is implemented as a Ring 0 kernel service. The kernel contains a device driver called vboxsrv. This device driver takes care of activities or tasks such as allocating physical memory for the guest virtual machine, loading hypervisor modules for functions such as saving and restoring the guest process context when a host interruption occurs, turning control over to the guest OS to begin execution, and deciding when the VT-x or AMD-V events need to be handled.

The guest itself manages its OS scheduling during its execution. The guest runs as a single process and only when scheduled by a host on the host system. Apart from this, there are additional device drivers present when the guest allows the OS to access resources such as disks, network controllers, and other devices.

The following diagram depicts high-level VirtualBox architecture:



Apart from the kernel modules, there are other processes running on the host that support the running guest. When a guest VM is started from the VirtualBox GUI, the VBoxSVC process starts automatically in the background.

VBoxSVC

VBoxSVC is the first client process that starts automatically and keeps running in the background; it is responsible for tasks such as maintaining the guest VM state and bookkeeping. With the help of COM/XPCOM, it provides communication between different VirtualBox components.

VirtualBox

VirtualBox is a GUI process based on a cross-platform QT library. When a guest VM starts, this process is started automatically. It can start with a startvm option or without any option. When this process starts with no --startvm option, it plays the role of the VirtualBox manager, displaying the VMs and their settings and communicating the settings and the state changes to the VBoxSVC process.
When it starts with the startvm option, it loads the actual hypervisor with the help of the VMM library and then runs the VM and provides the input and output for the guest VM.

VirtualBox also runs many internal processes and separate components.
The following are a few important components:

- **IPRT**: This is a portable library, this component is invoked whenever VirtualBox accesses host operating features with the help of this library for cross-platform portability.
- **VMM**: This is an integral part of the hypervisor.
- **REM** (**recompiled execution monitor**): This provides software emulation of CPU instructions.
- **EM** (**execution manager**): This manages and controls the execution of guest VM code.
- **HWACCM** (**hardware acceleration manager**): This provides support for VT-x and AMD-V.
- **PGM** (**page manager**): The guest VM paging activity is taken care of by this component.
- **TM** (**time manager**): This component handles timers and all the aspects of time inside guests.

- **CFGM** (**configuration manager**): This component provides all the settings related to the guest VM and all emulated devices stored in a tree structure.
- **SSM** (**saved state manager**): This component saves and loads the VM state.
- **VirtualBox**: This component is responsible for emulating many devices to provide the hardware environment that various guests need.
- **Main**: This is the only public API that VirtualBox provides and is responsible for tying all the preceding components together. This API is used by all the client processes because client processes cannot access the hypervisor directly.

Summary

In this chapter, we have covered the basics of virtualization in general and VirtualBox in particular. We have covered the features of VirtualBox, internals, architecture, and also the high-level technical aspects of VirtualBox.

In the next chapter, we will learn the process of installing and configuring VirtualBox on Linux.

2
Installation of Oracle VM VirtualBox on Linux

This chapter provides the information and steps to install and configure VirtualBox on Linux, and also detailed steps to start, update, and remove VirtualBox from the Linux OS.

Basic requirements

The following are the basic requirements for VirtualBox:

- **Processor**: Any recent AMD/Intel processor is fine to run VirtualBox.
- **Memory**: This is dependent on the size of the RAM that is required for the host OS, plus the amount of RAM needed by the guest OS. In my test environment, I am using 4 GB RAM and going to run Oracle Enterprise Linux 6.2 as the guest OS.
- **Hard disk space**: VirtualBox doesn't need much space, but you need to plan out how much the host OS needs and how much you need for the guest OS.
- **Host OS**: You need to make sure that the host OS is certified to run VirtualBox.

Host OS

At the time of writing this book, VirtualBox runs on the following host operating systems:

Windows

The following Microsoft Windows operating systems are compatible to run as host OS for VirtualBox:

- 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)
- Windows 8 (32-bit and 64-bit)
- Windows Server 2012 (64-bit)

Mac OS X

The following Mac OS X operating systems are compatible to run as host OS for VirtualBox:

- 10.6 (Snow Leopard, 32-bit and 64-bit)
- 10.7 (Lion, 32-bit and 64-bit)
- 10.8 (Mountain Lion, 64-bit)

Linux

The following Linux operating systems are compatible to run as host OS for VirtualBox:

- Debian GNU/Linux 5.0 (lenny) and 6.0 (squeeze)
- Oracle Enterprise Linux 4 and 5, Oracle Linux 6
- Red Hat Enterprise Linux 4, 5, and 6
- Fedora Core 4 to 17
- Gentoo Linux
- openSUSE 11.0, 11.1, 11.2, 11.3, 11.4, 12.1, and 12.2
- Mandriva 2010 and 2011

Solaris

Both 32-bit and 64-bit versions of Solaris are supported with some limitations. Please refer to `www.virtualbox.org` for more information. The following host OS are supported:

- Solaris 11 including Solaris 11 Express
- Solaris 10 (update 8 and higher)

Guest OS

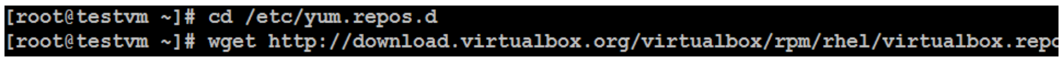
You need to make sure that the guest OS is certified to run on VirtualBox. VirtualBox supports the following guest operating systems:

- Windows 3.x
- Windows NT 4.0
- Windows 2000
- Windows XP
- Windows Server 2003
- Windows Vista
- Windows 7
- DOS
- Linux (2.4, 2.6, 3.0, 3.1, 3.2, 3.3, 3.4, 3.5, 3.6, and 3.7)
- Solaris
- OpenSolaris
- OpenBSD

Installation

I have downloaded and used the VirtualBox repository to install VirtualBox. Please ensure the desktop or laptop you are installing VirtualBox on is connected to the Internet. However, this is not mandatory. You can install VirtualBox using the RPM command, which generally adds the complexity of finding and installing the dependency packages, or you can have a central yum server where you can add VirtualBox repository.

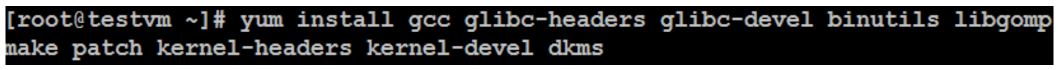
In my test lab, my laptop is connected to the Internet and I have used the `wget` command to download and add `virtualbox.repository`.



```
[root@testvm ~]# cd /etc/yum.repos.d
[root@testvm ~]# wget http://download.virtualbox.org/virtualbox/rpm/rhel/virtualbox.rep
```

Installing dependency packages

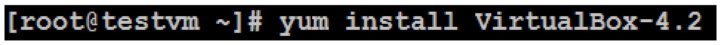
Please ensure installing the packages seen in the following screenshot to make VirtualBox work:



```
[root@testvm ~]# yum install gcc glibc-headers glibc-devel binutils libgomp
make patch kernel-headers kernel-devel dkms
```

Installing VirtualBox 4.2.16

Once the preceding dependency is installed, we are ready to install VirtualBox 4.2.16. Use the command seen in the following screenshot to install VirtualBox:



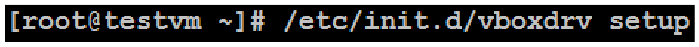
```
[root@testvm ~]# yum install VirtualBox-4.2
```

Rebuilding kernel modules

The `vboxdrv` module is a special kernel module that helps to allocate the physical memory and to gain control of the processor for the guest system execution. Without this kernel module, you can still use the VirtualBox manager to configure the virtual machines, but they will not start.

When you install VirtualBox by default, this module gets installed on the system. But to maintain future kernel updates, I suggest that you install Dynamic Kernel Module Support (DKMS). In most of the cases, this module installation is straightforward. You can use yum, apt-get, and so on depending on the Linux variant you are using, but ensure that the GNU compiler (GCC), GNU make (make), and packages containing header files for your kernel are installed prior to installing DKMS. Also, ensure that all system updates are installed.

Once the kernel of your Linux host is updated and DKMS is not installed, the kernel module needs to be reinstalled by executing the following command as root:



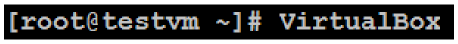
```
[root@testvm ~]# /etc/init.d/vboxdrv setup
```

The preceding command not only rebuilds the kernel modules but also automatically creates the `vboxusers` group and the VirtualBox user.

If you use Microsoft Windows on your desktop, then download the `.exe` file, install it, and start it from the desktop shortcut or from the program.

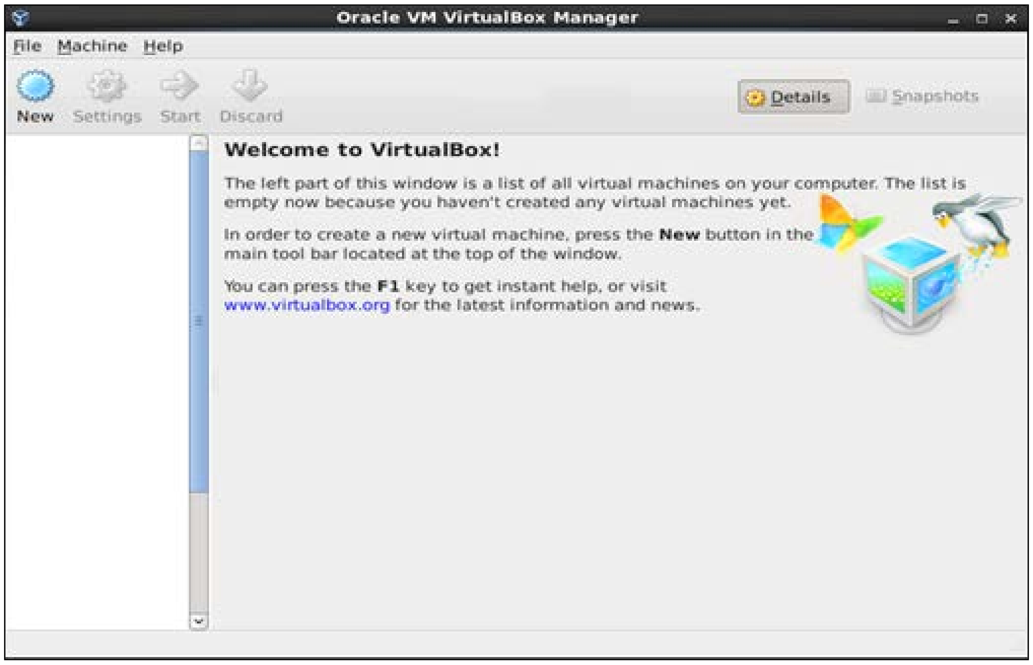
Start VirtualBox

Use the command seen in the following screenshot in Linux to run VirtualBox:



```
[root@testvm ~]# VirtualBox
```

If everything is fine, then the **Oracle VM VirtualBox Manager** screen appears as seen in the following screenshot:



Welcome to VirtualBox!

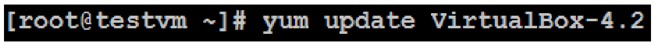
The left part of this window is a list of all virtual machines on your computer. The list is empty now because you haven't created any virtual machines yet.

In order to create a new virtual machine, press the **New** button in the main tool bar located at the top of the window.

You can press the **F1** key to get instant help, or visit www.virtualbox.org for the latest information and news.

Update VirtualBox

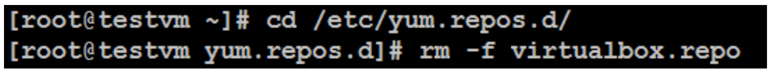
You can update VirtualBox with the help of the command seen in the following screenshot:



```
[root@testvm ~]# yum update VirtualBox-4.2
```

Remove VirtualBox

To remove VirtualBox, use the command seen in the following screenshot:



```
[root@testvm ~]# cd /etc/yum.repos.d/
[root@testvm yum.repos.d]# rm -f virtualbox.repo
```

Summary

In this chapter we have covered the installation, update, and removal of Oracle VM VirtualBox in the Linux environment.

In the next chapter we will learn the process to create, start, stop, remove, and clone a guest VM as well as the usage of some great features such as snapshot, Clone, VM Groups.

3
VM Creation on VirtualBox

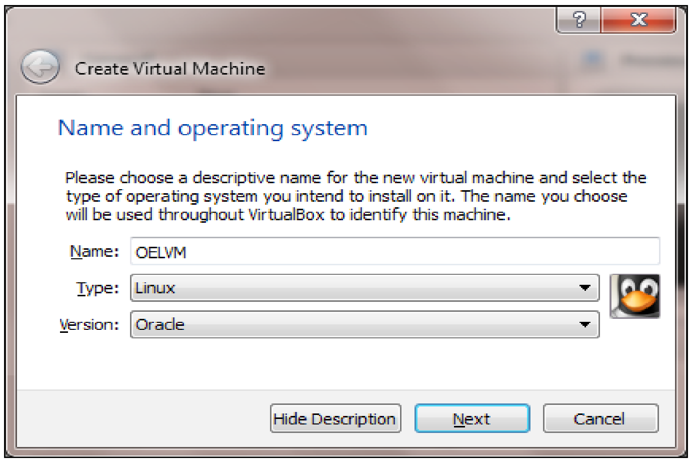
This chapter provides the information and steps for creating a guest virtual machine as well as starting, stopping, removal, and cloning it, and also some great features such as snapshot, clone, and VM groups.

Creating the guest virtual machine

On the VirtualBox Manager, click on the **New** button at the top and a wizard that will help to create the virtual machine will pop up.

The wizard will ask for some information based on the information provided by the user and the guest VM will then be created.

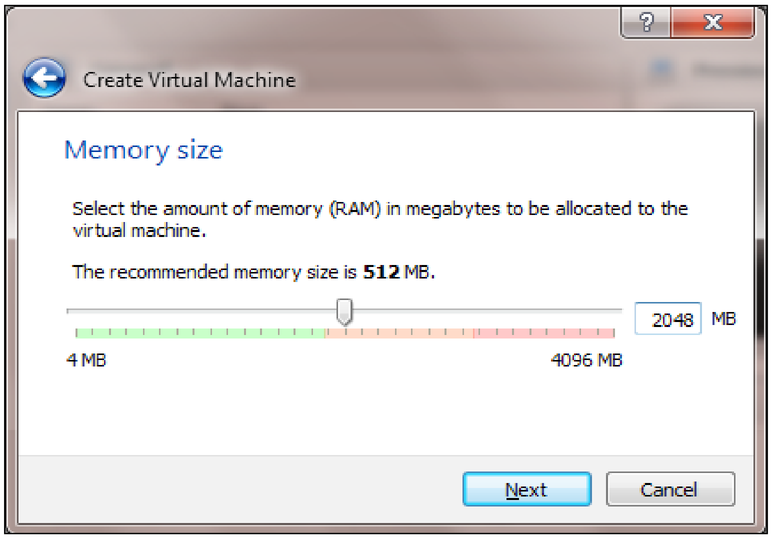
As shown in the following screenshot, provide the name of the VM; I have used the name `OELVM`. In the **Type** field, **Linux** should be selected. If you want some other OS to be the guest VM, then choose it from the list. In the **Version** field, choose **Oracle** and click on **Next**.



Please choose a descriptive name for the new virtual machine and select the type of operating system you intend to install on it. The name you choose will be used throughout VirtualBox to identify this machine.

Name: OELVM
Type: Linux
Version: Oracle

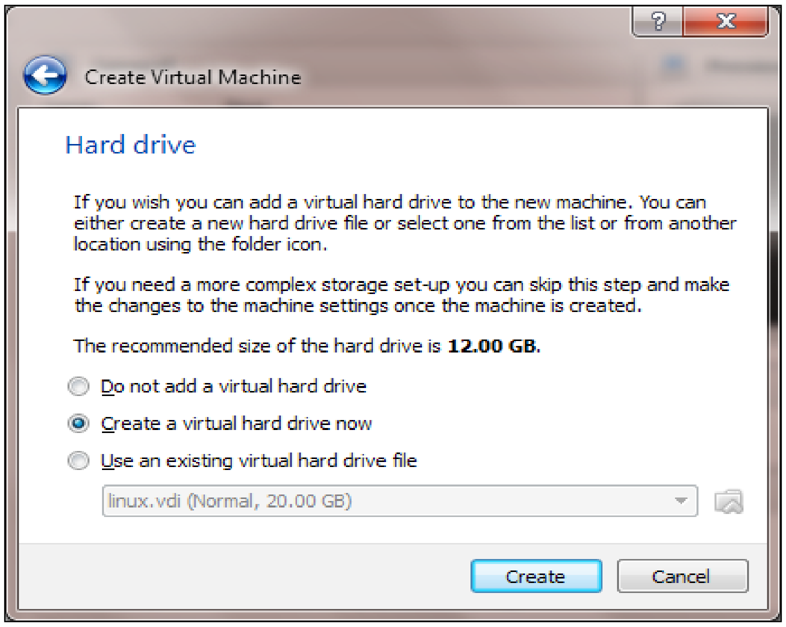
In the following screenshot, select the amount of memory. I have kept **2048 MB** for **Oracle Enterprise Linux**. However, you may resize it after the installation. Click on the **Next** button.



Select the amount of memory (RAM) in megabytes to be allocated to the virtual machine.

The recommended memory size is **512** MB.

The following screen will appear. Select the **Create a virtual hard drive now** radio button. Click on the **Create** button. It will create a new virtual hard drive.



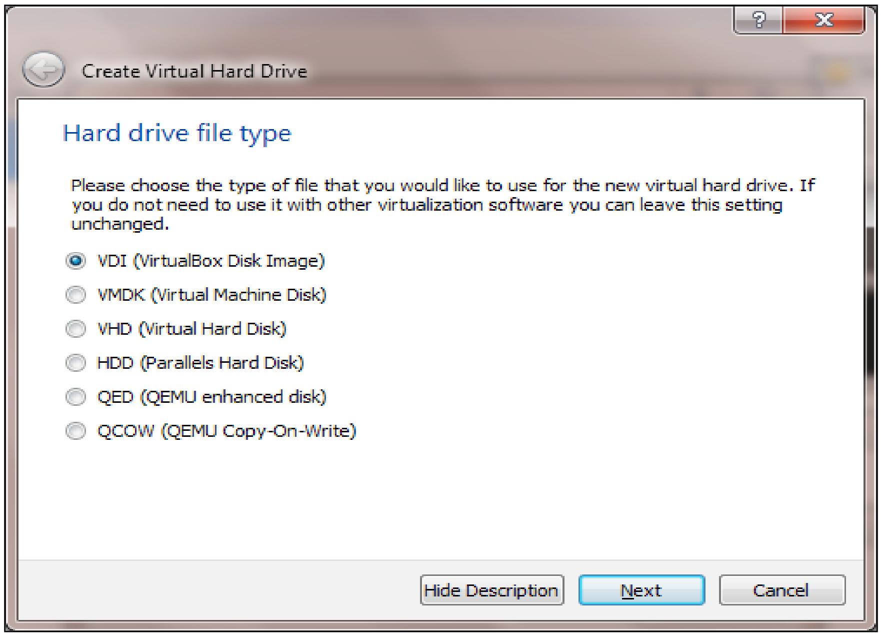
If you wish you can add a virtual hard drive to the new machine. You can either create a new hard drive file or select one from the list or from another location using the folder icon.

If you need a more complex storage set-up you can skip this step and make the changes to the machine settings once the machine is created.

The recommended size of the hard drive is **12.00 GB.**

- ○ Do not add a virtual hard drive
- ● Create a virtual hard drive now
- ○ Use an existing virtual hard drive file

Select **VDI** as shown in the following screenshot and click on **Next**:



Hard drive file type

Please choose the type of file that you would like to use for the new virtual hard drive. If you do not need to use it with other virtualization software you can leave this setting unchanged.

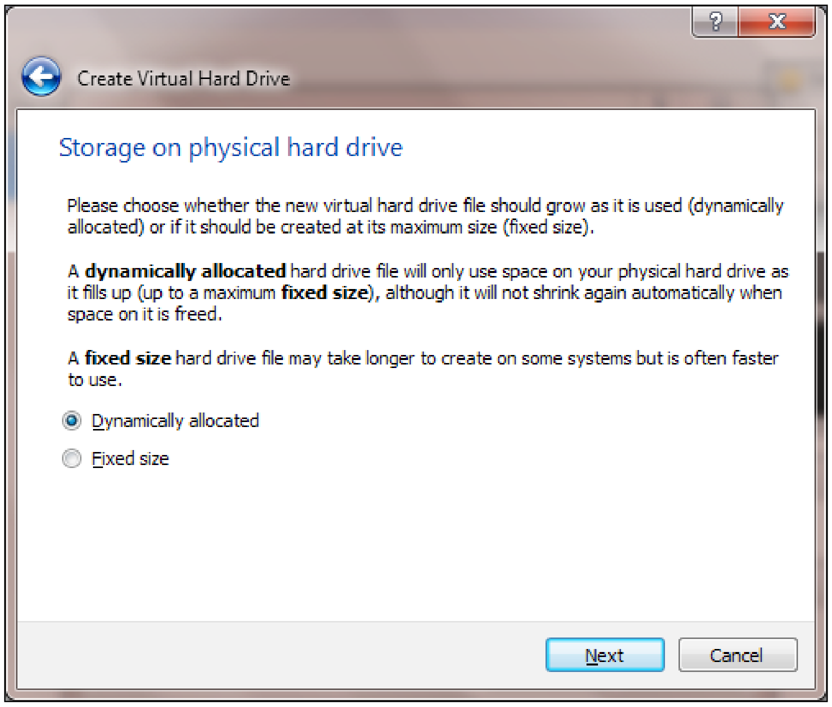
- [✓ VDI (VirtualBox Disk Image)]
- VMDK (Virtual Machine Disk)
- VHD (Virtual Hard Disk)
- HDD (Parallels Hard Disk)
- QED (QEMU enhanced disk)
- QCOW (QEMU Copy-On-Write)

You can choose either of the options; I have chosen **Dynamically allocated**. The difference between **Dynamically allocated** and **Fixed size** is self-explanatory.

A **dynamically allocated disk** is initially small on the host hard disk, and it grows later depending on the data that is written to the virtual disk by the guest OS.

A **fixed size disk** will immediately occupy the entire size that is allocated, even if only a fraction of the virtual hard disk space is actually in use.

As shown in the following screenshot, click on **Next**:



Storage on physical hard drive

Please choose whether the new virtual hard drive file should grow as it is used (dynamically allocated) or if it should be created at its maximum size (fixed size).

A **dynamically allocated** hard drive file will only use space on your physical hard drive as it fills up (up to a maximum **fixed size**), although it will not shrink again automatically when space on it is freed.

A **fixed size** hard drive file may take longer to create on some systems but is often faster to use.

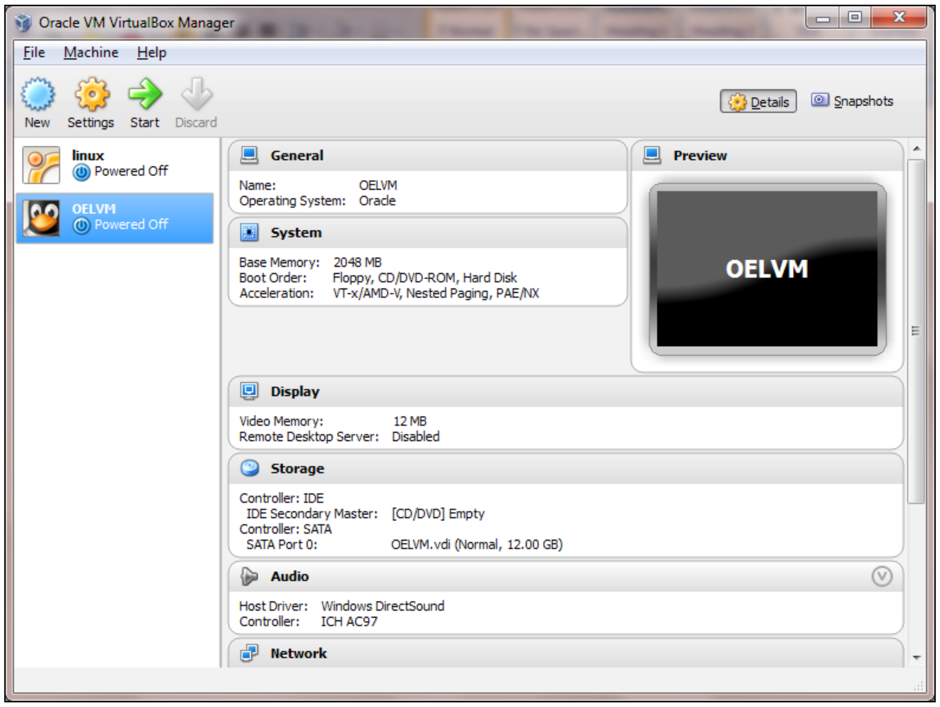
[✓ Dynamically allocated]

Fixed size

Select the size of the disk for your VM; I have selected 12 GB considering that it's a test machine. If you need more space, please resize it accordingly. Click on the **Create** button.

It will take a bit of time to create the VirtualBox disk image (VDI). Then your virtual hard disk will be created.

If everything goes well, you can see the guest VM that you have created. In my case, it is **OELVM**.



Name: OELVM
Operating System: Oracle

Base Memory: 2048 MB
Boot Order: Floppy, CD/DVD-ROM, Hard Disk
Acceleration: VT-x/AMD-V, Nested Paging, PAE/NX

Video Memory: 12 MB
Remote Desktop Server: Disabled

Controller: IDE
IDE Secondary Master: [CD/DVD] Empty
Controller: SATA
SATA Port 0: OELVM.vdi (Normal, 12.00 GB)

Host Driver: Windows DirectSound
Controller: ICH AC97

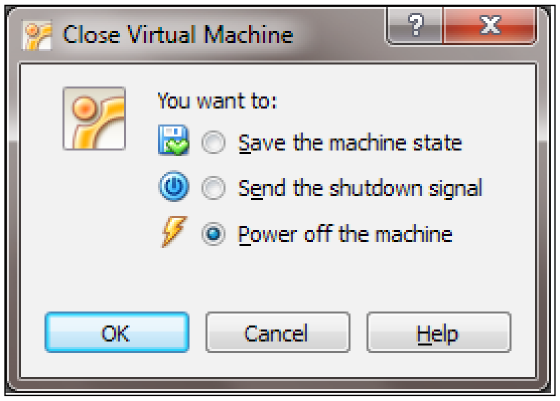
Starting the virtual machine

The guest VM can be started using various methods; you can choose any one of the following options:

- Double-click on the name of the guest VM in the VirtualBox Manager window
- Select the name of the guest VM and click on the **Start** button at the top of the window
- The `VBoxManage` command can also be used

We will talk about the guest VM's installation in the next chapter.

When you close the guest VM window, the following screen with three options appears:



You want to:

Save the machine state

Send the shutdown signal

[✓ Power off the machine]

The difference between these three options are very important.

- **Save the machine state**: When you select this option, VirtualBox "freezes" the VM by saving its state completely to the hard disk. So, if you opt for this option and start your VM later, you will see that the VM starts from where you have left it; that means all the programs will be still open in the state that you have left it in and the guest VM resumes operation.
- **Send the shutdown signal**: This option sends an ACPI shutdown signal to the guest VM. It is similar to pressing the power button on a physical computer.
- **Power off the machine**: If you select this option, VirtualBox stops running the VM without saving its state. This is similar to pulling the power cable of a physical host. This option is not suggested because there might be a chance of data inconsistency and data loss as well.

Based on your requirement, select any one of the three options. I suggest you avoid the **Power off the machine** option unless you want to quickly restore the current snapshot of the VM.

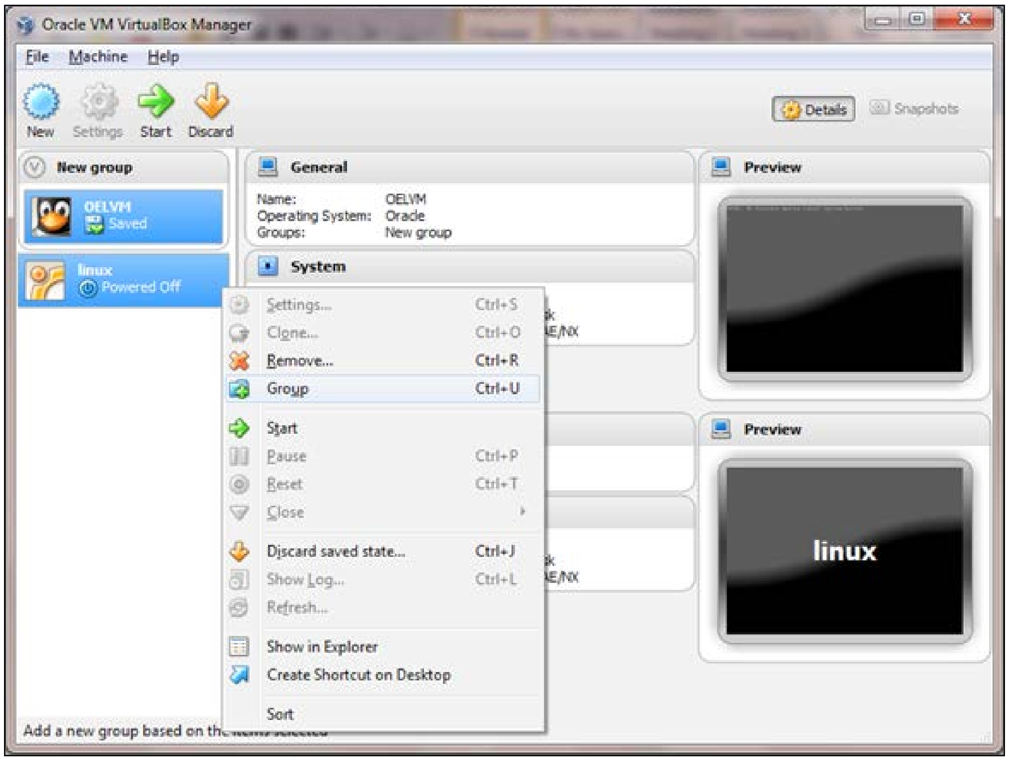
Virtual machine groups

VM groups enable the user to create ad-hoc groups of VMs that help to manage and perform some activities on them collectively or individually. For example, if you want to host a three-tier application based on the Apache, Tomcat, and Oracle database for your test environment hosted on individual virtual machines, you can create a group called "TestEnv" and put these three VMs in this group. By doing this you can perform activities such as start, stop, pause, reset, and close. This functionality helps a lot in terms of grouping and performing management-related activities simultaneously.

There are a number of features available while using groups.

VM groups can be created with any of the following methods:

- Create a group using the GUI option
- Drag one VM on top of another VM
- Select multiple VMs and select **Group**



Name: OELVM
Operating System: Oracle
Groups: New group

Add a new group based on the items selected

You can also use the command `VBoxManage` to create VM groups.

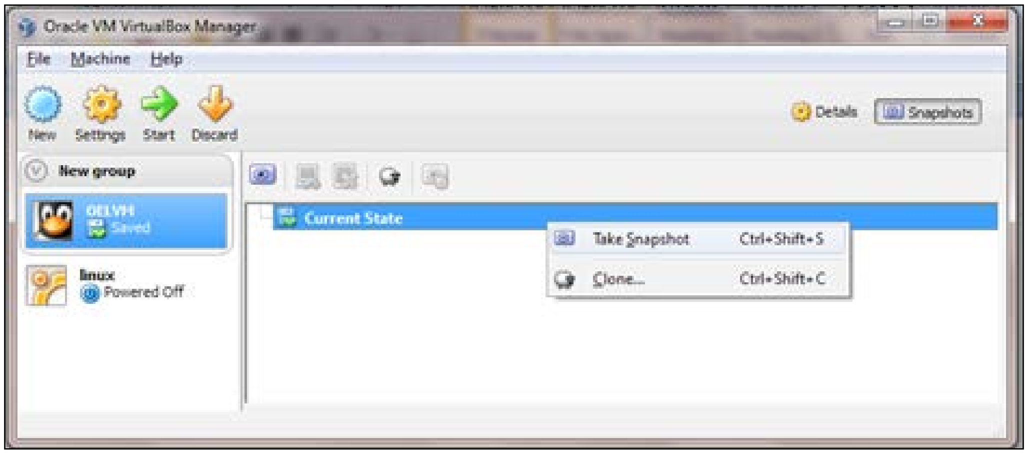
Virtual machine snapshots

Snapshots help to save the exact state of a guest VM for later use. At any point in time, you can easily revert to the previous required state even though you may have changed the guest VM. It's similar to the saved state but many states can be preserved.

A snapshot not only preserves the complete state of all the virtual disks that are attached to the guest machine but also stores a complete copy of the VM settings, including the hardware configuration in an XML text file. Thus, when you restore a snapshot, the VM settings are restored as well as all the configuration files are stored in the XML format, so it takes very little space to store all the information.

When a snapshot is taken while the VM is running, the memory state of the guest VM also gets saved in the snapshot and the snapshots stored in the local hard disk. As a result, when the execution resumes, the guest VM starts exactly from the same state as when the snapshot was taken.

You can take a snapshot of a VM by selecting a machine in the VirtualBox Manager and then clicking on the **Snapshots** button on the top-right corner.

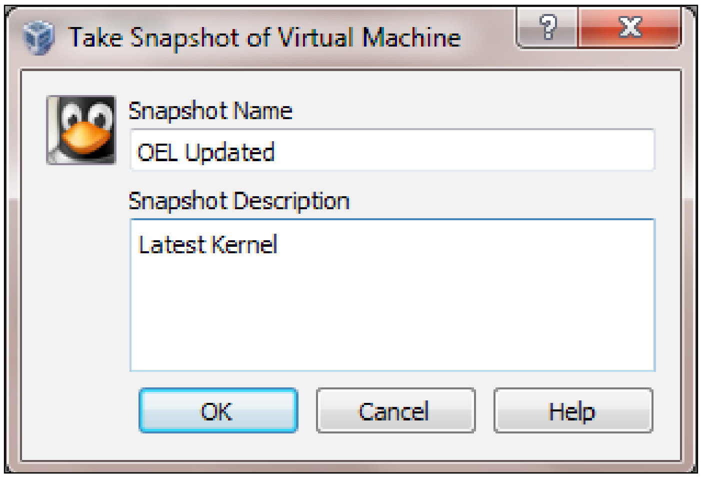


The list of snapshots looks empty except for the **Current State** item. **Current State** represents the "Now" point in the lifetime of the VM.

Taking, restoring, and deleting snapshots

You can take the snapshot of a guest VM that is either running, saved, or in the power-off state.

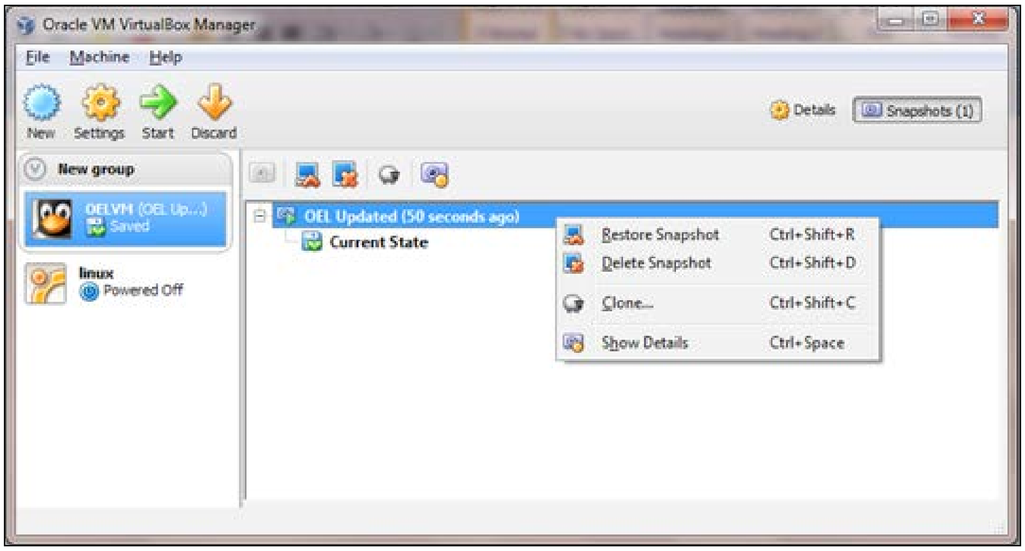
To take the snapshot, you either click on the **Snapshots** tab on the top-right corner of the VirtualBox Manager or right-click on the guest VM and select **Take Snapshot**. The following screen appears, where you will be asked to provide the name of the snapshot and the description. It's good practice to give a proper name and description then click on **OK**. Now, your snapshot is created with the snapshot name that you have provided. In this example, I have provided `OEL Updated`.



Snapshot Name
OEL Updated
Snapshot Description
Latest Kernel

To check the details of the snapshots, click on the VM. You can see the list of snapshots in the right-hand panel.

If you right-click on the snapshot, you can see options as shown in the following screenshot:



The **Restore Snapshot** option helps you to go forward or backward in time. By selecting this option you will lose the current state of the VM, but you will be able to restore the VM to the exact state when you have taken the snapshot.

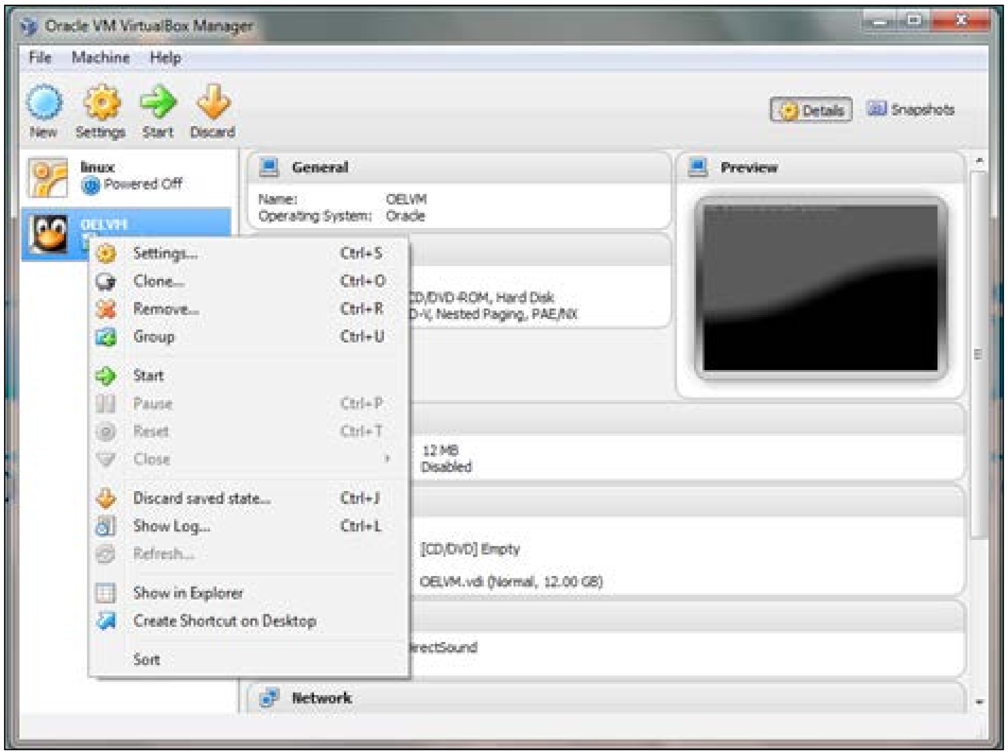
But if you don't want to lose your current state, take a new snapshot of the current VM before restoring the VM.

The **Delete Snapshot** option helps to delete the snapshots of the guest VM, which in turn frees up the disk space. You can also delete the snapshots while the VM is in a running state.

Virtual machine clone

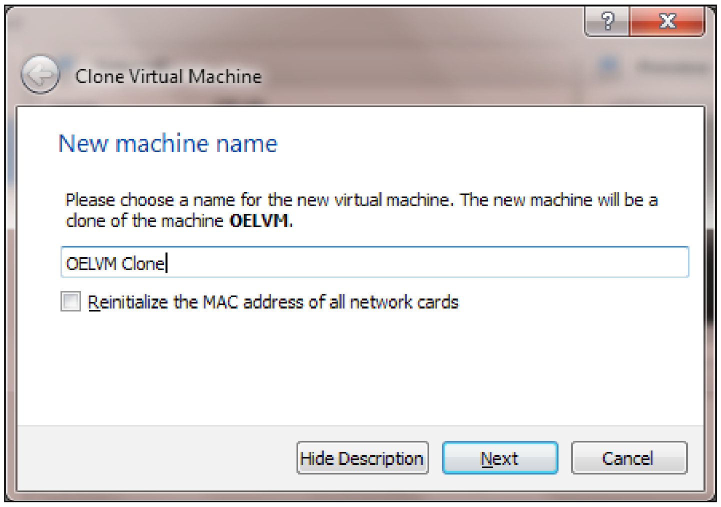
If you want to test some functionality in the currently running VM without interruption, or you want to make a backup, you can think of cloning. This feature helps to create the golden image or base image as well.

To make a clone, right-click on the guest VM and click on **Clone…**.



Name: OELVM
Operating System: Oracle
CD/DVD-ROM, Hard Disk
D-V, Nested Paging, PAE/NX
12 MB
Disabled
[CD/DVD] Empty
OELVM.vdi (Normal, 12.00 GB)
DirectSound

Once you click on **Clone...**, a screen appears, as shown in the following screenshot:



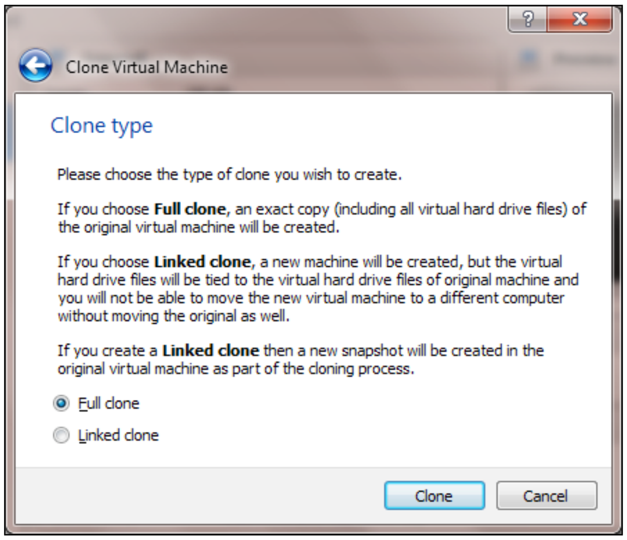
New machine name

Please choose a name for the new virtual machine. The new machine will be a clone of the machine **OELVM**.

OELVM Clone

☐ Reinitialize the MAC address of all network cards

Provide the name of the clone. If you select **Reinitialize the MAC address of all network cards**, every network card attached to the clone VM (Destination VM) gets a new MAC address assigned. If you don't select it, the clone will have the same MAC address as the source VM. This will produce a MAC address collision in the network layer if you decide to use the source VM and cloned VM in the same network. However, its not an issue if you want your source VM and cloned VM in different networks. So, depending on the requirement either choose it or leave it as it is and click on **Next**.



Clone type

Please choose the type of clone you wish to create.

If you choose **Full clone**, an exact copy (including all virtual hard drive files) of the original virtual machine will be created.

If you choose **Linked clone**, a new machine will be created, but the virtual hard drive files will be tied to the virtual hard drive files of original machine and you will not be able to move the new virtual machine to a different computer without moving the original as well.

If you create a **Linked clone** then a new snapshot will be created in the original virtual machine as part of the cloning process.

● Full clone

○ Linked clone

Select the clone type. The difference is explained in the window itself. I have selected **Full clone**. Once you select the clone type, you need to decide what you need to clone, either the current state or all. If you select all, not only is the current state of the guest VM cloned but all the other snapshots are also cloned.

Bear in mind that the time taken to complete a clone operation depends on the size of the disk images attached to the guest VM.

Apart from creating the VM from scratch or cloning we can import VMs in Open Virtualization Format (OVF).

Summary

In this chapter, we learned how to create a guest VM as well as start, stop, remove, and clone it. We also learned about some great features such as snapshot, clone, and VM groups.

In the next chapter, we will learn how to configure virtual machines and different settings such as installing Oracle Enterprise Linux 6.2 as a guest VM, and concepts about the memory management in guest VMs.

4 Installing OEL

This chapter provides a step-by-step guide on setting up an Oracle Enterprise Linux 6.0 guest virtual machine in VirtualBox. It also gives an overview of sharing folders in Windows and Linux, and the memory management techniques used in VirtualBox.

Installing Oracle Enterprise Linux 6.0

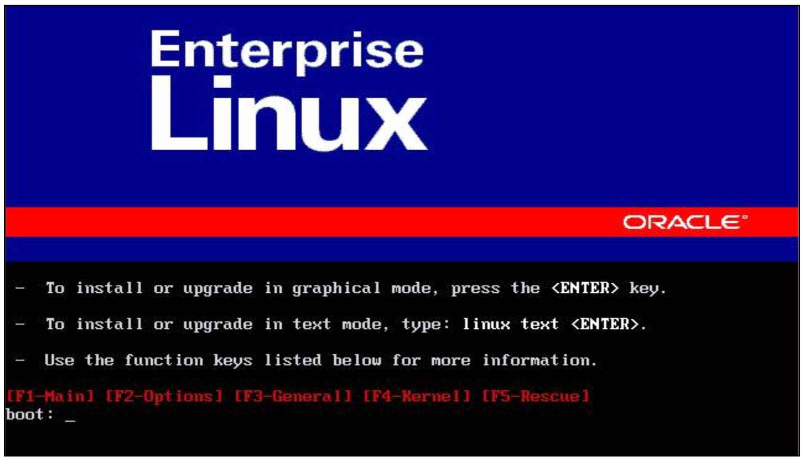
A wizard pops up the first time you start the VM. This wizard helps you to either select the installation medium (CD or DVD) where the operating system image is mounted, or directly select an ISO image.

The ISO image of Oracle Enterprise Linux 6.0 is used for this guide. You can download the ISO at `https://edelivery.oracle.com/linux`.

Prior to downloading you need to register on an eDelivery site. Once the user is created and confirmed, you can use the same credentials to log in to this site to download the ISO.

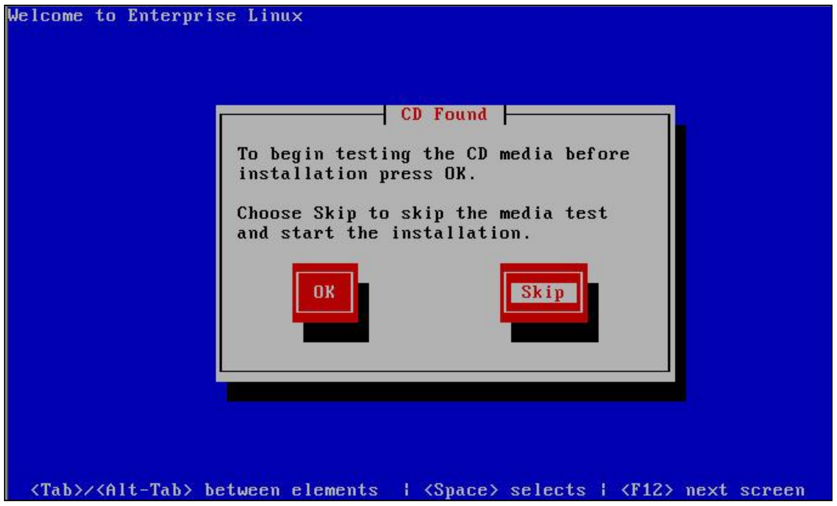
The steps for installation are as follows:

1. Once you select the ISO image, the following screen appears. On the boot screen, press the *Enter* key.



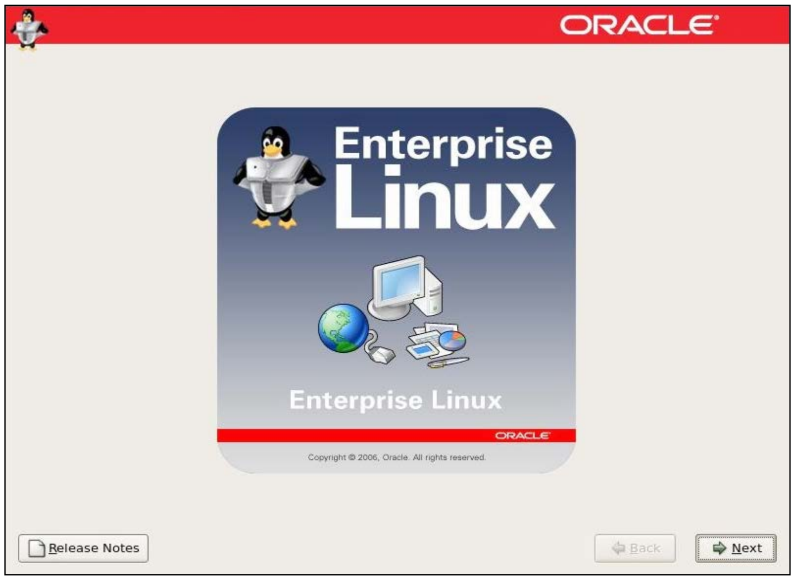
```
-  To install or upgrade in graphical mode, press the <ENTER> key.

-  To install or upgrade in text mode, type: linux text <ENTER>.

-  Use the function keys listed below for more information.

[F1-Main] [F2-Options] [F3-General] [F4-Kernel] [F5-Rescue]
boot: _
```

2. In the following screenshot, if you select **ok**, the media will be tested and it will take a bit of time to complete the testing. If you are sure that your media is fine, select **Skip** and press *Enter*.

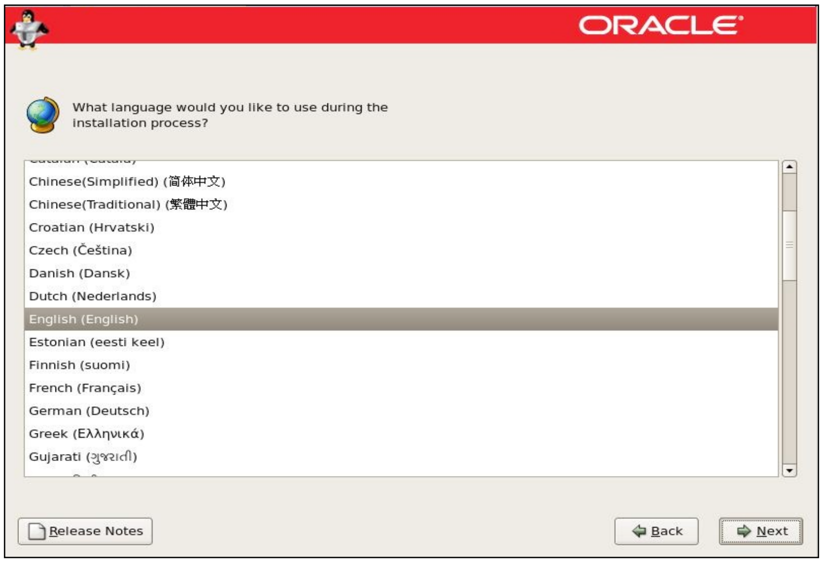


```
Welcome to Enterprise Linux

                 CD Found
   To begin testing the CD media before
   installation press OK.

   Choose Skip to skip the media test
   and start the installation.

          OK                  Skip

 <Tab>/<Alt-Tab> between elements  | <Space> selects | <F12> next screen
```

3. As shown in the following screenshot, the installation starts with a welcome screen; select **Next** to continue.



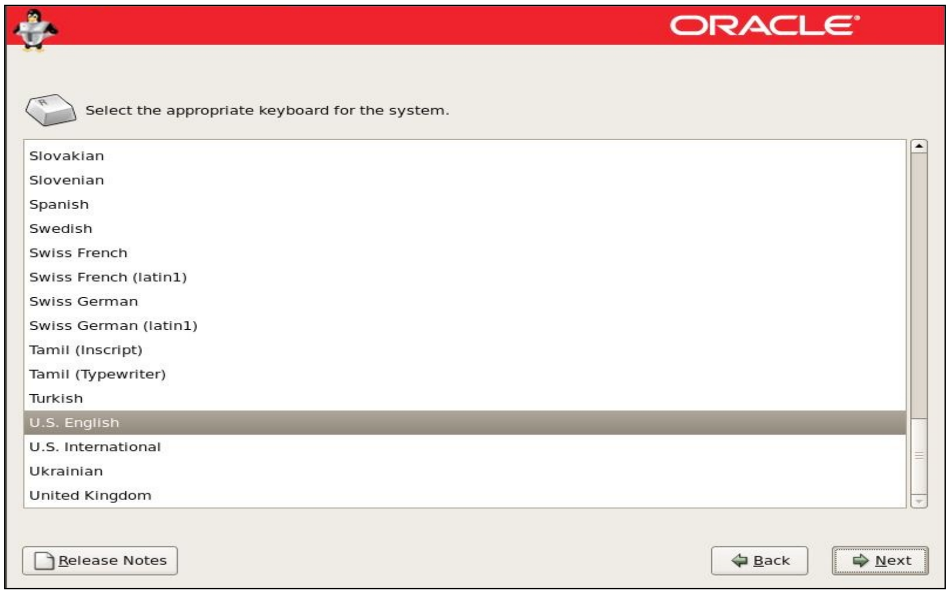
4. As shown in the following screenshot, select the appropriate language, in this case **English**, and press **Next**.



What language would you like to use during the installation process?

Chinese(Simplified) (简体中文)
Chinese(Traditional) (繁體中文)
Croatian (Hrvatski)
Czech (Čeština)
Danish (Dansk)
Dutch (Nederlands)
English (English)
Estonian (eesti keel)
Finnish (suomi)
French (Français)
German (Deutsch)
Greek (Ελληνικά)
Gujarati (ગુજરાતી)

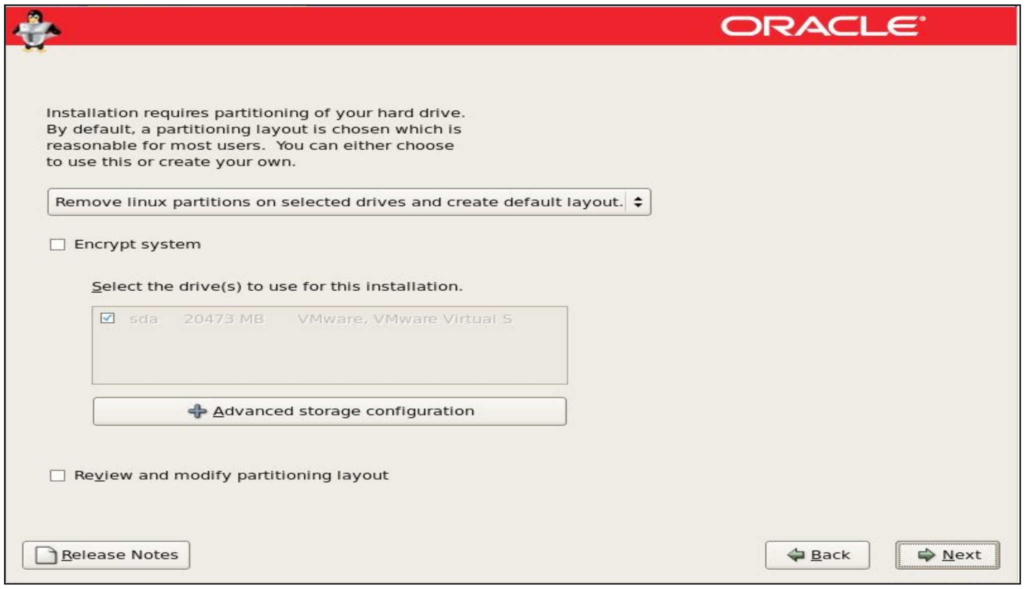
5. As shown in the following screenshot select the keyboard layout appropriate to your location in this case, **U.S. English**, then click on **Next.**



Select the appropriate keyboard for the system.

Slovakian
Slovenian
Spanish
Swedish
Swiss French
Swiss French (latin1)
Swiss German
Swiss German (latin1)
Tamil (Inscript)
Tamil (Typewriter)
Turkish
U.S. English
U.S. International
Ukrainian
United Kingdom

6. Considering the virtual machine is the first one and is new, as shown in the following screenshot, accept the default options to **remove the Linux partitions** and then click on **Next**.



Installation requires partitioning of your hard drive.
By default, a partitioning layout is chosen which is reasonable for most users. You can either choose to use this or create your own.

Remove linux partitions on selected drives and create default layout.

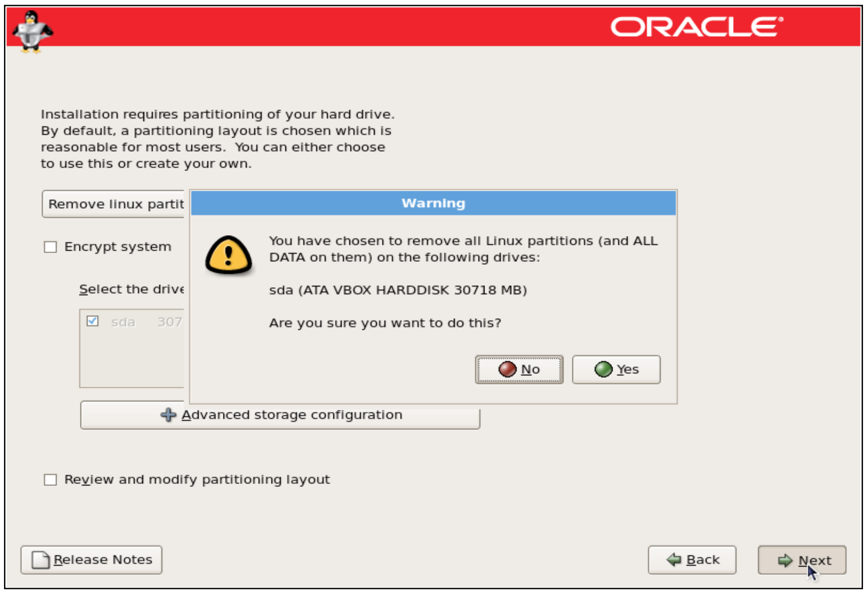
☐ Encrypt system

Select the drive(s) to use for this installation.

☑ sda 20473 MB VMware, VMware Virtual S

☐ Review and modify partitioning layout

7. As shown in the following screenshot, Oracle Enterprise Linux will warn you about losing all data on the virtual disk. Click on **Yes** in the pop up that appears.



Installation requires partitioning of your hard drive.
By default, a partitioning layout is chosen which is
reasonable for most users. You can either choose
to use this or create your own.

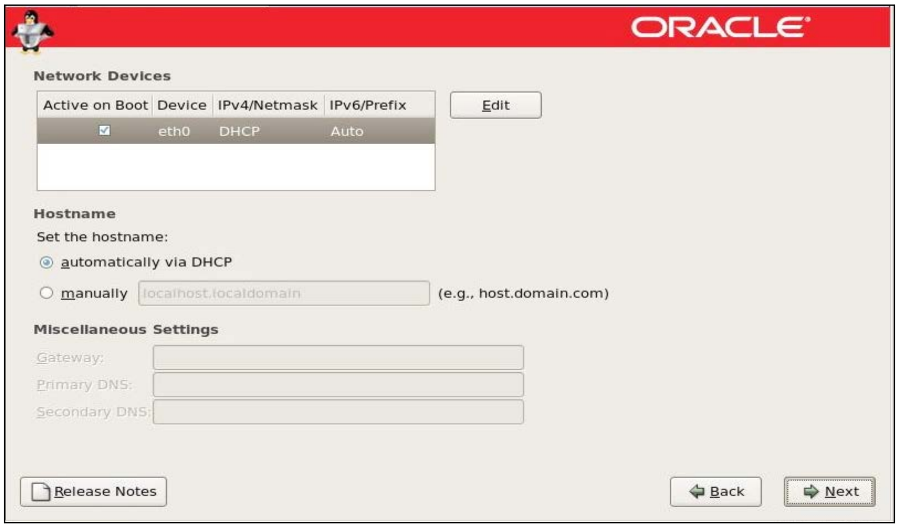
Warning

You have chosen to remove all Linux partitions (and ALL DATA on them) on the following drives:

sda (ATA VBOX HARDDISK 30718 MB)

Are you sure you want to do this?

8. As shown in the following screenshot, enter the hostname manually and give your machine a name. If you want to specify an IP manually, select **Edit** and provide the IP-related details. For guidance, the DHCP configuration is enabled.



Network Devices

Active on Boot	Device	IPv4/Netmask	IPv6/Prefix
✓	eth0	DHCP	Auto

Hostname

Set the hostname:

● automatically via DHCP

○ manually localhost.localdomain (e.g., host.domain.com)

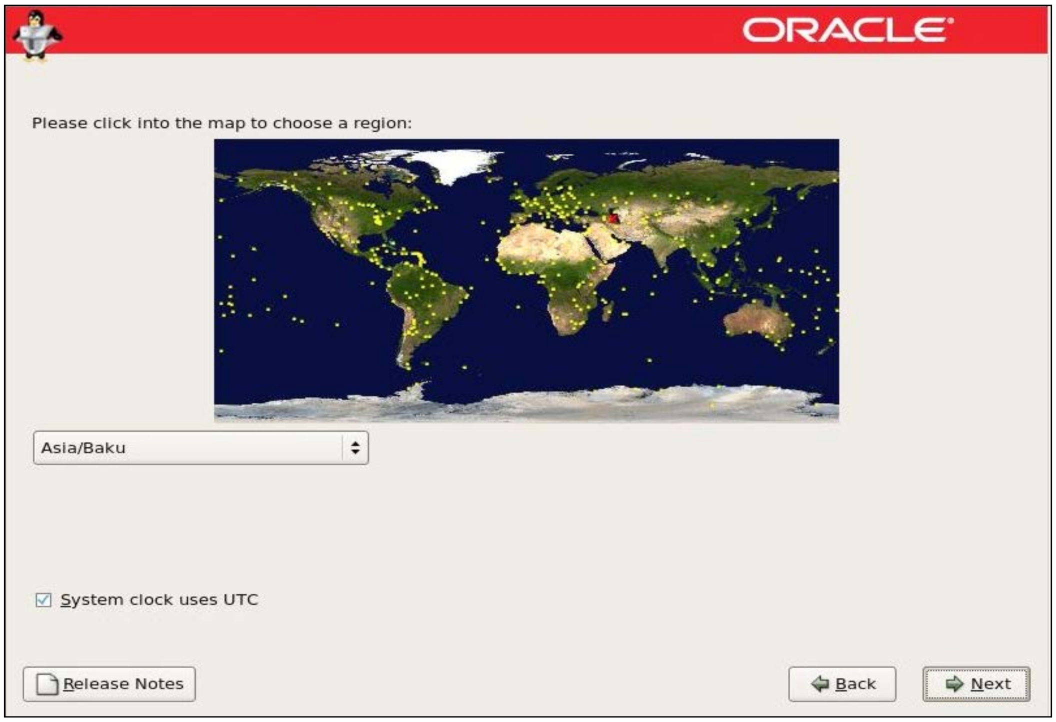
Miscellaneous Settings

Gateway:

Primary DNS:

Secondary DNS:

9. As shown in the following screenshot, select your geographical region and time zone. Leave the system clock set to UTC.

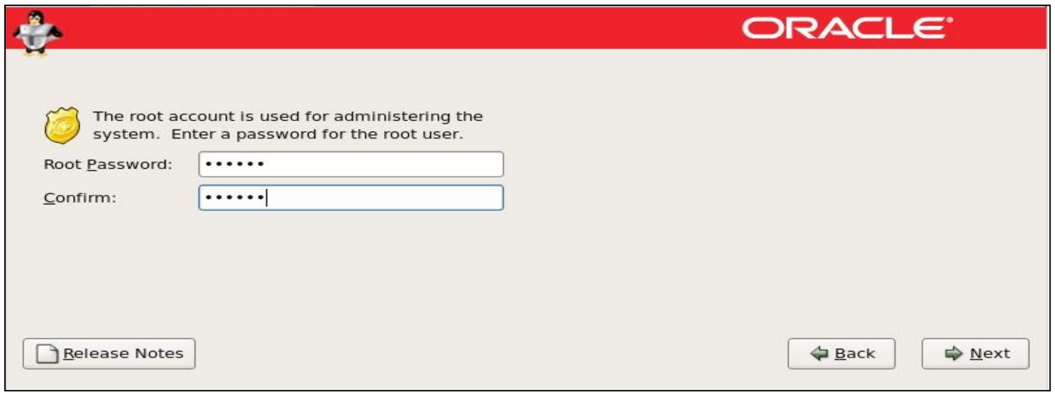


Please click into the map to choose a region:

Asia/Baku

☑ System clock uses UTC

10. As shown in the following screenshot, provide a password for the root user. Choose a secure password.

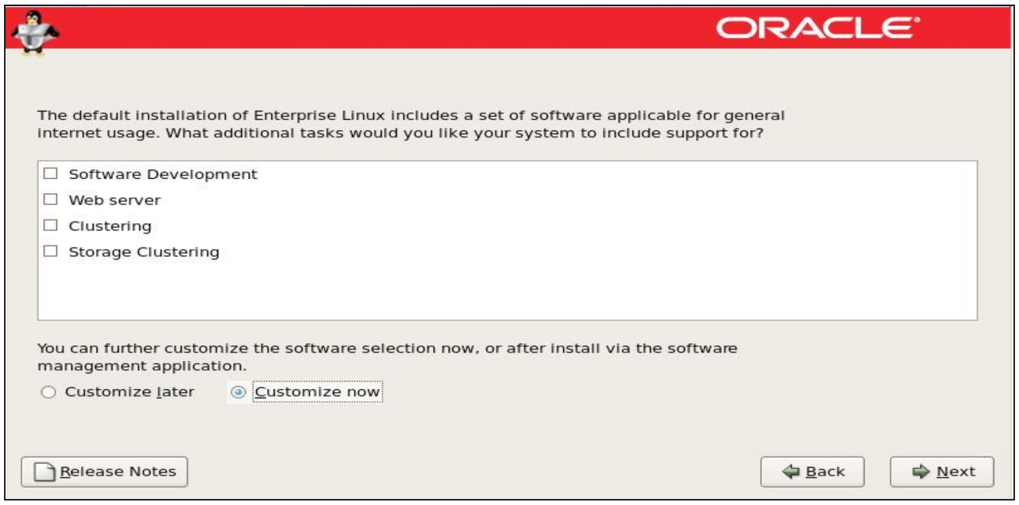


The root account is used for administering the system. Enter a password for the root user.

Root Password: ••••••

Confirm: ••••••

11. As shown in the following screenshot, depending on your need, select **Software Development** and then click on **Next**. You can install the software post installation either manually or by YUM (Yellowdog Updater, Modified (YUM) is a program which manages installation, updates and removal for Red Hat package manager (RPM) systems. YUM allows the user to update groups of machines without having to update each RPM separately).



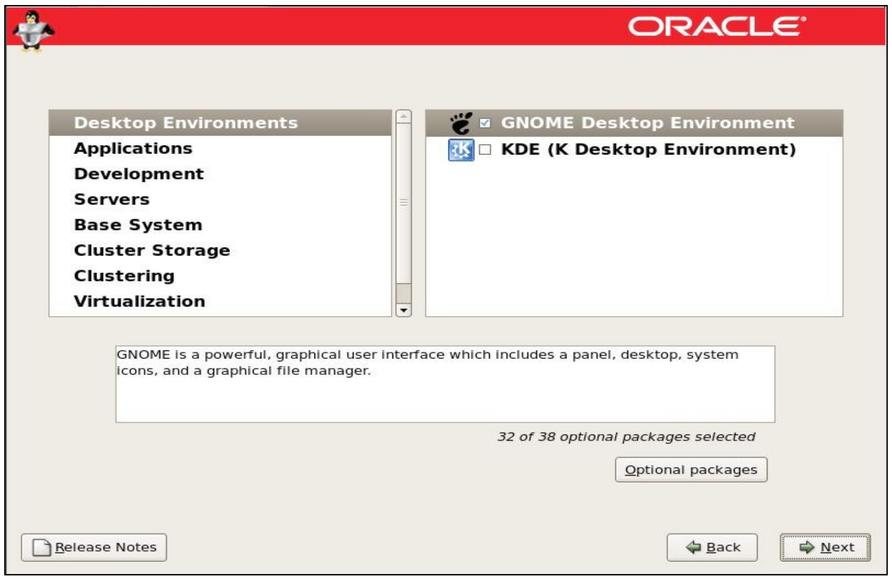
The default installation of Enterprise Linux includes a set of software applicable for general internet usage. What additional tasks would you like your system to include support for?

- ☐ Software Development
- ☐ Web server
- ☐ Clustering
- ☐ Storage Clustering

You can further customize the software selection now, or after install via the software management application.

Customize later [✓ Customize now]

12. As shown in the following screenshot, click on **Customize now** if you want to select the required packages during installation.



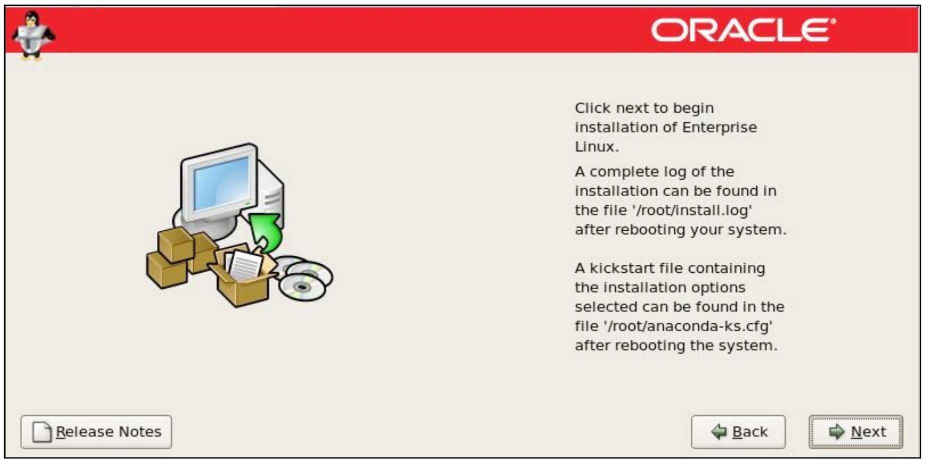
Desktop Environments
Applications
Development
Servers
Base System
Cluster Storage
Clustering
Virtualization

- ☑ GNOME Desktop Environment
- ☐ KDE (K Desktop Environment)

GNOME is a powerful, graphical user interface which includes a panel, desktop, system icons, and a graphical file manager.

32 of 38 optional packages selected

13. As shown in the following screenshot, click on **Next**.

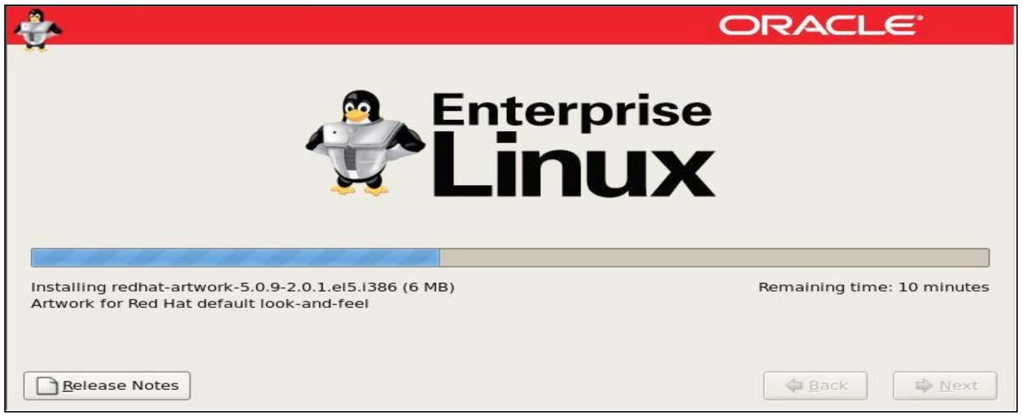


Click next to begin installation of Enterprise Linux.

A complete log of the installation can be found in the file '/root/install.log' after rebooting your system.

A kickstart file containing the installation options selected can be found in the file '/root/anaconda-ks.cfg' after rebooting the system.

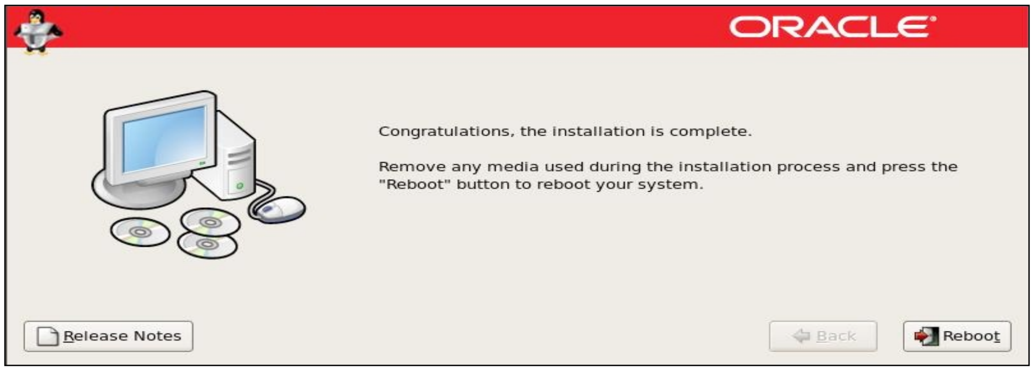
14. As shown in the following screenshot, the installation will take some time.



Installing redhat-artwork-5.0.9-2.0.1.el5.i386 (6 MB)
Artwork for Red Hat default look-and-feel

Remaining time: 10 minutes

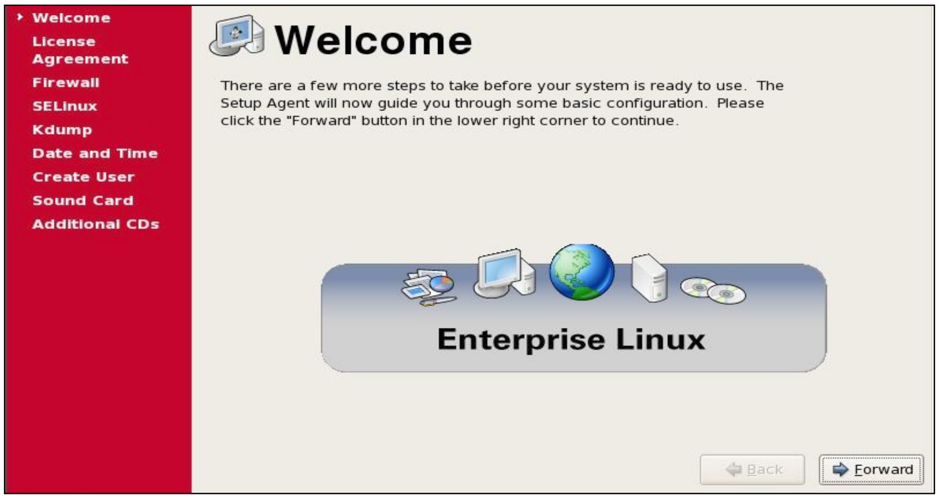
15. Once the installation is complete, click on **Reboot** and wait for the system to restart.



Congratulations, the installation is complete.

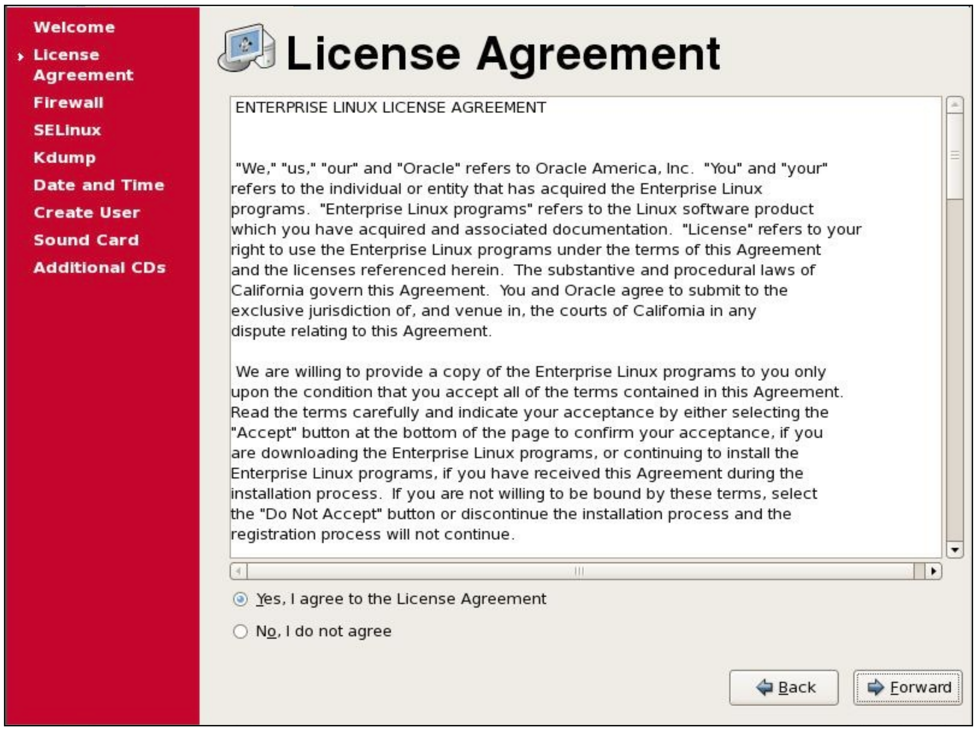
Remove any media used during the installation process and press the "Reboot" button to reboot your system.

16. As shown in the following screenshot, click on **Forward**.



There are a few more steps to take before your system is ready to use. The Setup Agent will now guide you through some basic configuration. Please click the "Forward" button in the lower right corner to continue.

17. As shown in the following screenshot, agree to the license agreement and click on **Forward.**



License Agreement

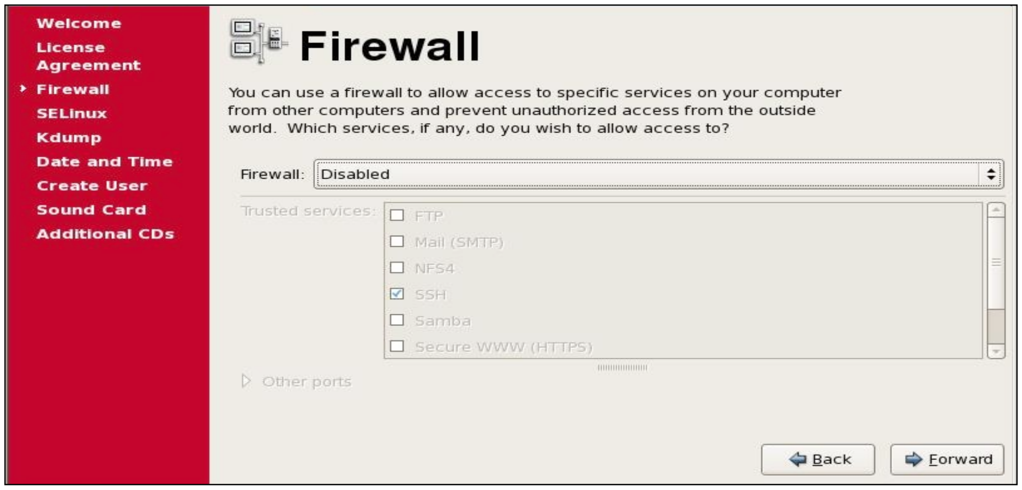
ENTERPRISE LINUX LICENSE AGREEMENT

"We," "us," "our" and "Oracle" refers to Oracle America, Inc. "You" and "your" refers to the individual or entity that has acquired the Enterprise Linux programs. "Enterprise Linux programs" refers to the Linux software product which you have acquired and associated documentation. "License" refers to your right to use the Enterprise Linux programs under the terms of this Agreement and the licenses referenced herein. The substantive and procedural laws of California govern this Agreement. You and Oracle agree to submit to the exclusive jurisdiction of, and venue in, the courts of California in any dispute relating to this Agreement.

We are willing to provide a copy of the Enterprise Linux programs to you only upon the condition that you accept all of the terms contained in this Agreement. Read the terms carefully and indicate your acceptance by either selecting the "Accept" button at the bottom of the page to confirm your acceptance, if you are downloading the Enterprise Linux programs, or continuing to install the Enterprise Linux programs, if you have received this Agreement during the installation process. If you are not willing to be bound by these terms, select the "Do Not Accept" button or discontinue the installation process and the registration process will not continue.

[✓ Yes, I agree to the License Agreement]
No, I do not agree

18. We leave the firewall **Disabled,** assuming that the installation is used for testing purposes, as shown in the following screenshot:



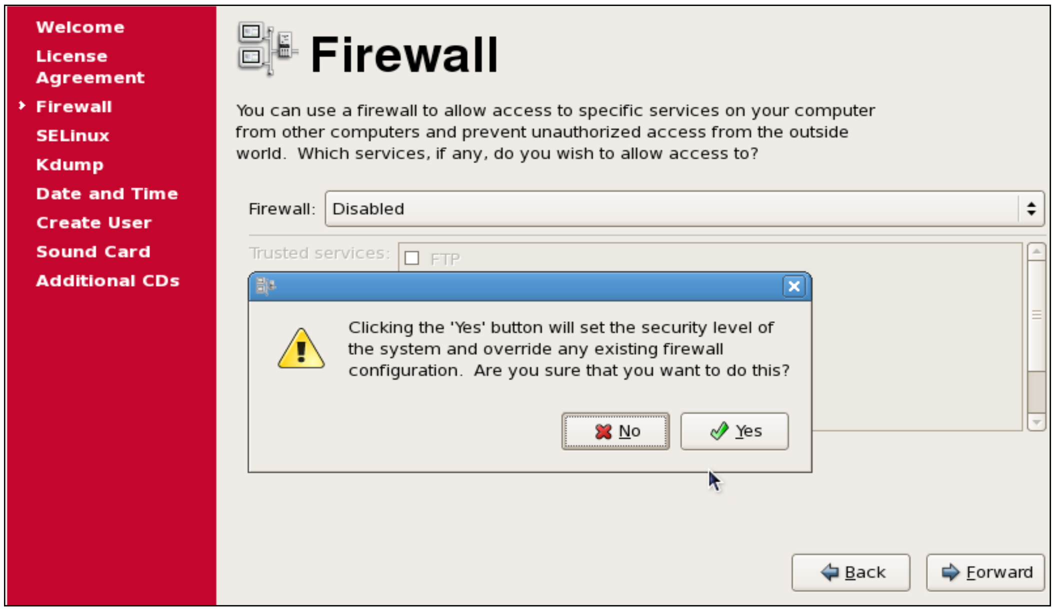
Firewall

You can use a firewall to allow access to specific services on your computer from other computers and prevent unauthorized access from the outside world. Which services, if any, do you wish to allow access to?

Firewall: Disabled

Trusted services:
- ☐ FTP
- ☐ Mail (SMTP)
- ☐ NFS4
- ☑ SSH
- ☐ Samba
- ☐ Secure WWW (HTTPS)

19. Select **Yes,** as shown in the following screenshot:

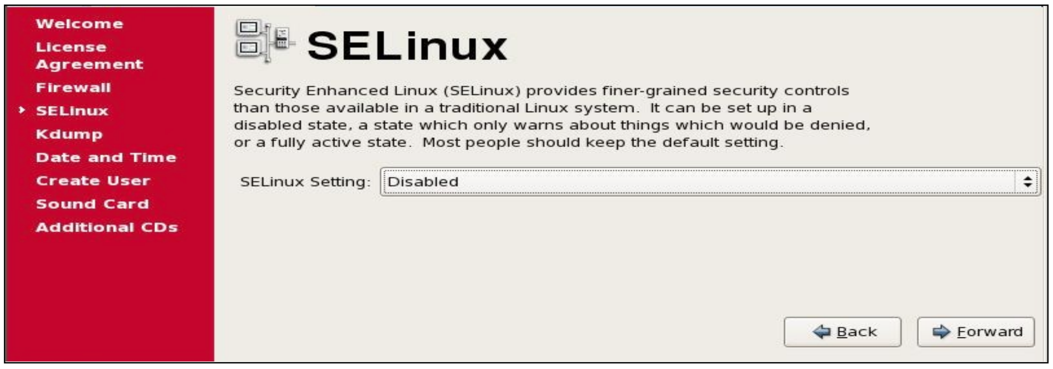


You can use a firewall to allow access to specific services on your computer from other computers and prevent unauthorized access from the outside world. Which services, if any, do you wish to allow access to?

Firewall: Disabled

Clicking the 'Yes' button will set the security level of the system and override any existing firewall configuration. Are you sure that you want to do this?

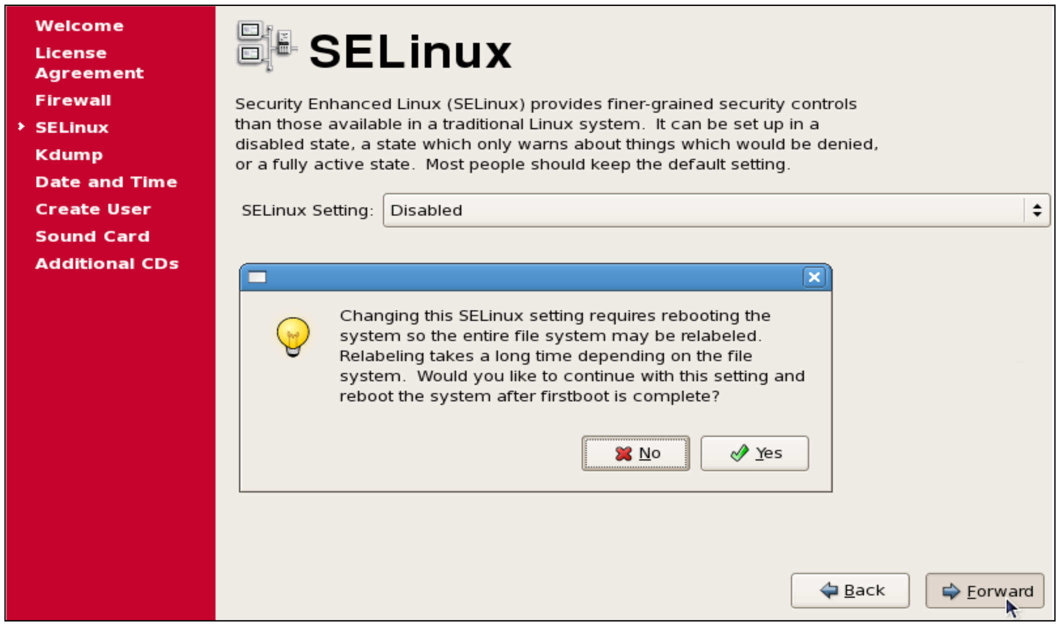
20. We leave SELinux **Disabled,** assuming that the installation is used for testing purposes, as shown in the following screenshot:



Security Enhanced Linux (SELinux) provides finer-grained security controls than those available in a traditional Linux system. It can be set up in a disabled state, a state which only warns about things which would be denied, or a fully active state. Most people should keep the default setting.

SELinux Setting: Disabled

21. As shown in the following screenshot, select **Yes**.



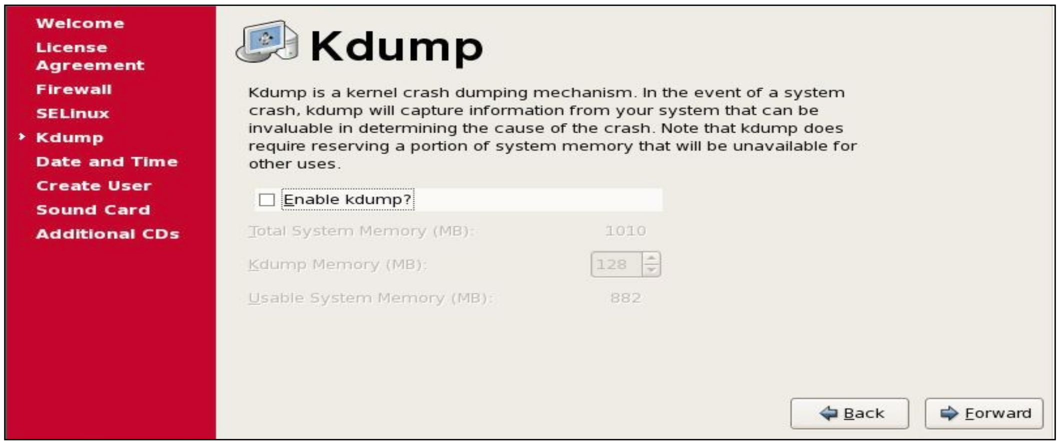
SELinux

Security Enhanced Linux (SELinux) provides finer-grained security controls than those available in a traditional Linux system. It can be set up in a disabled state, a state which only warns about things which would be denied, or a fully active state. Most people should keep the default setting.

SELinux Setting: Disabled

Changing this SELinux setting requires rebooting the system so the entire file system may be relabeled. Relabeling takes a long time depending on the file system. Would you like to continue with this setting and reboot the system after firstboot is complete?

22. As shown in the following screenshot, leave Kdump **disabled** and click on **Forward**.



Kdump

Kdump is a kernel crash dumping mechanism. In the event of a system crash, kdump will capture information from your system that can be invaluable in determining the cause of the crash. Note that kdump does require reserving a portion of system memory that will be unavailable for other uses.

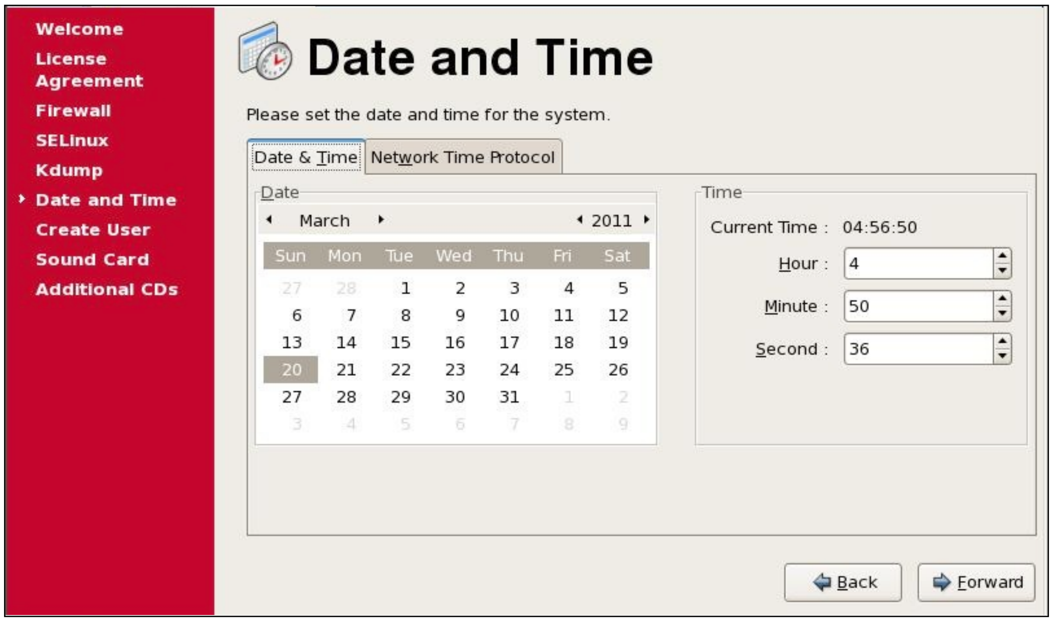
☐ Enable kdump?

Total System Memory (MB): 1010

Kdump Memory (MB): 128

Usable System Memory (MB): 882

23. As this is a VM, the date and time should be automatically picked up from the host machine, as shown in the following screenshot:

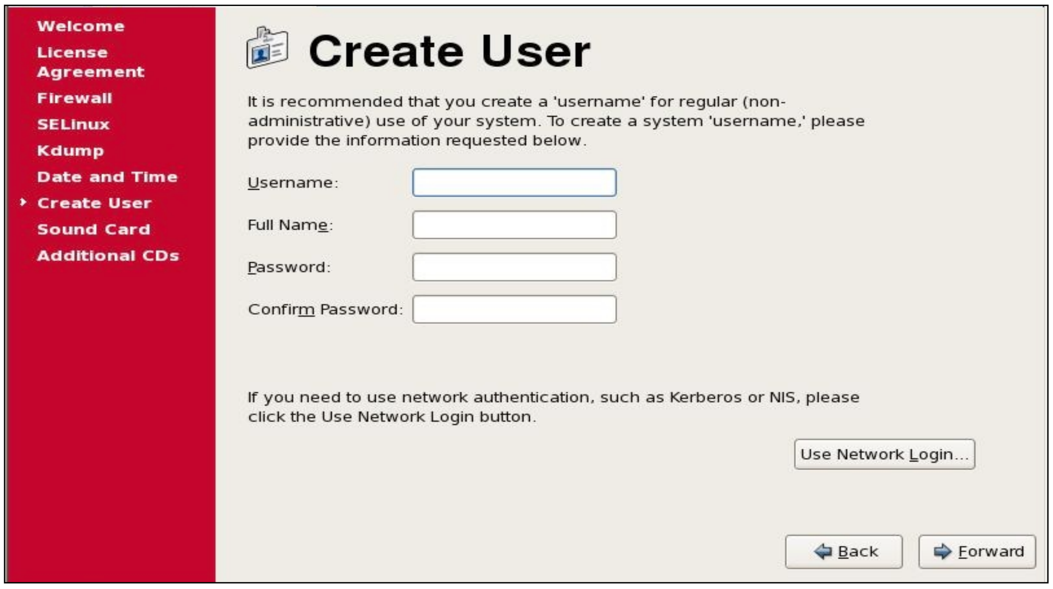


Please set the date and time for the system.

Date: March 2011, 20

Time
Current Time : 04:56:50
Hour : 4
Minute : 50
Second : 36

24. As shown in the following screenshot, click on **Forward** in case you want to create additional users.

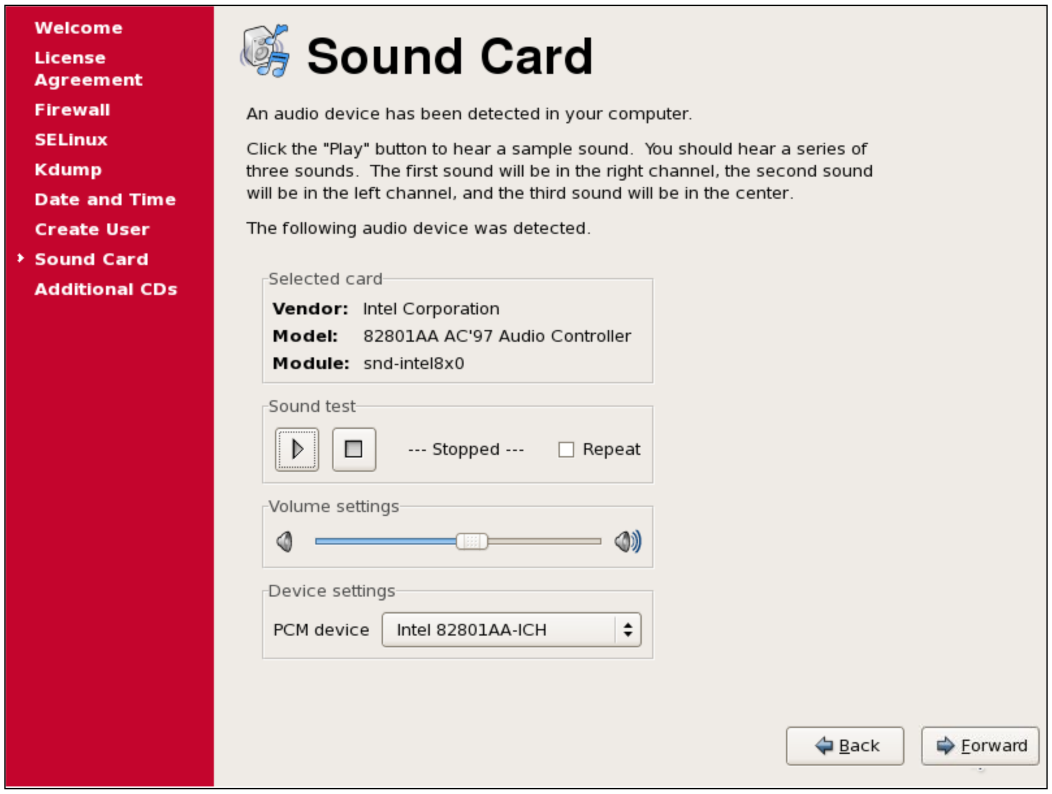


It is recommended that you create a 'username' for regular (non-administrative) use of your system. To create a system 'username,' please provide the information requested below.

Username:
Full Name:
Password:
Confirm Password:

If you need to use network authentication, such as Kerberos or NIS, please click the Use Network Login button.

25. As shown in the following screenshot, accept the default settings and click on **Forward**.



Sound Card

An audio device has been detected in your computer.

Click the "Play" button to hear a sample sound. You should hear a series of three sounds. The first sound will be in the right channel, the second sound will be in the left channel, and the third sound will be in the center.

The following audio device was detected.

Selected card

Vendor: Intel Corporation

Model: 82801AA AC'97 Audio Controller

Module: snd-intel8x0

Sound test

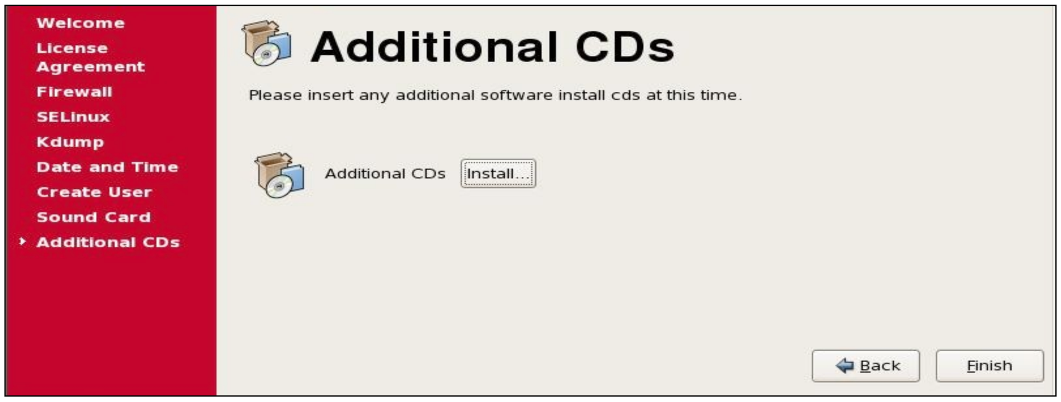
--- Stopped --- Repeat

Volume settings

Device settings

PCM device Intel 82801AA-ICH

26. On the following screenshot, no additional CD is required, so click on **Finish**.

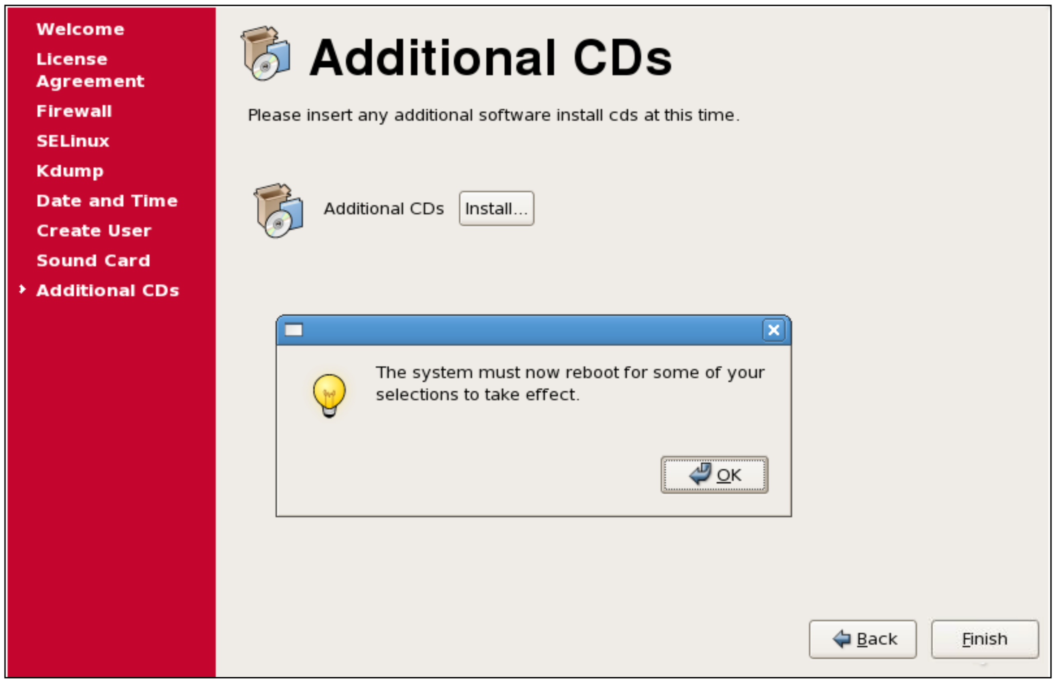


Additional CDs

Please insert any additional software install cds at this time.

Additional CDs Install...

27. As shown in the following screenshot, installation is now complete. Clicking on **OK** will reboot the system and you will be prompted to log in after restart.



Please insert any additional software install cds at this time.

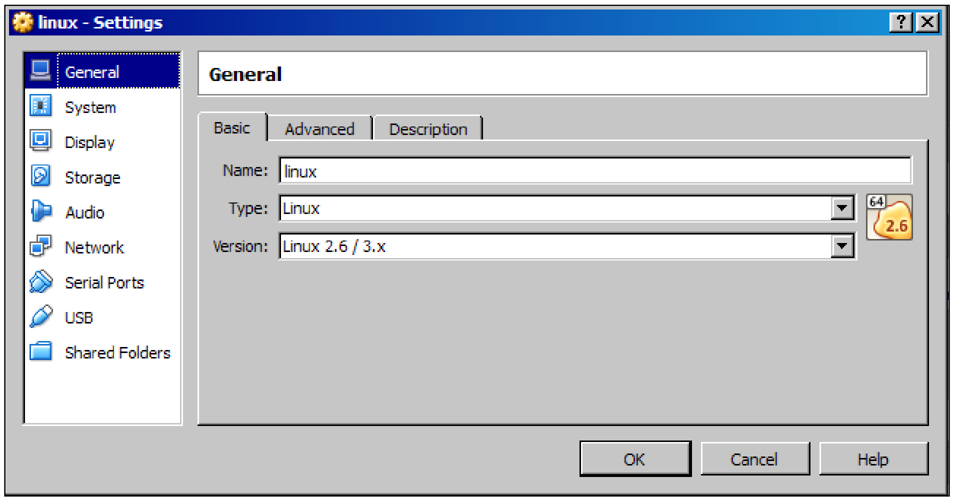
The system must now reboot for some of your selections to take effect.

Once the VM is installed, it can use the virtualized hardware provided by its configuration.

The VM uses the following hardware:

- Input devices
- Graphics
- Storage
- Networking
- Audio

You can change the preceding settings by selecting any of the following tabs from the VM settings. Please refer to the following screenshot:



Name: linux
Type: Linux
Version: Linux 2.6 / 3.x

The different settings are as follows:

- **General** settings
- **System** settings
- **Display** settings
- **Storage** setting
- **Audio** settings
- **Network** settings
- **Serial Ports**
- **USB**
- **Shared Folders**

A detailed explanation of each of the settings is out of the scope of this book.

Installing VirtualBox Guest Additions

VirtualBox **Guest Additions** is an extra add-on software that not only helps to improve the interactive performance of guest systems, but also provides a closer integration between the host and guest VM. Guest Additions needs to be installed once the guest VM is installed. This software contains device drivers and applications that optimize the performance and usability of the guest OS.

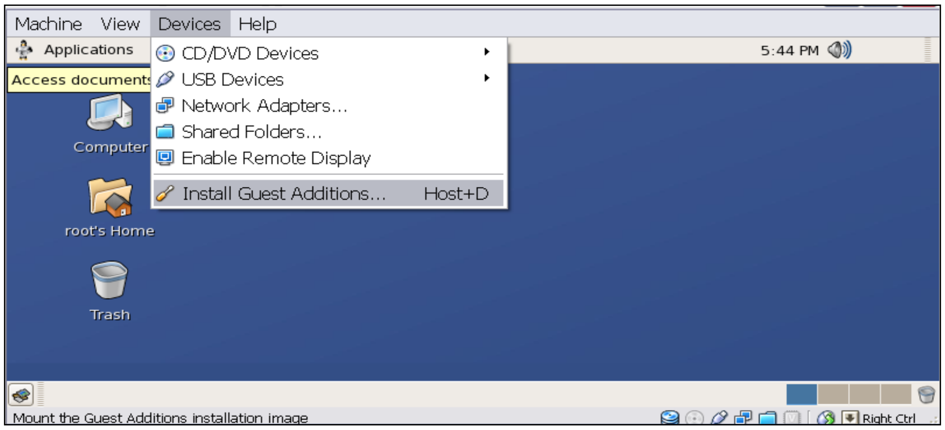
The following features can be leveraged once you install Guest Addition:

- **Mouse pointer integration**: Guest Additions helps to improve and enhance the mouse support. When you install the add-on, it installs a special mouse driver in the guest VM, which communicates with the actual mouse driver on the host OS to help move the mouse pointer.
- **Shared folders**: This enables the feature that helps to exchange files between the guest VM and the host machine. To use this feature you need to select or identify the host directory, which needs to be shared across all or multiple guest VMs.
- Better video support and display: When you install this add-on, it installs a specific video driver that provides very high and nonstandard video modes, which also accelerate video performance.
- **Seamless windows**: This feature helps to map the guest VMs individual desktop Windows on the host desktop. It seems like the application is actually running on the host.
- **Time synchronization**: This feature synchronizes the guest VM's time with the host.
- **Shared clipboard**: This feature helps to share the clipboard of the guest OS with the host OS.

Ways to install VirtualBox Guest Additions

Guest Additions can be installed in two ways, either through GUI or from the command line.

As shown in the following screenshot, select the VM and click on **Devices**, and then click on **Install Guest Additions** to install the Guest Additions.



Mount the Guest Additions installation image

Now open a terminal, switch to the root user, and run the following script:

```
# su - root
# sh /media/VBOX*/VBoxLinuxAdditions.run
```

Once the installation is complete, you will need to reboot again before the Guest Additions software takes effect.

Using shared folders

Shared folders help you to access files placed in your host system from a guest system. It is similar to network sharing, except that the shared folders do not need a network. They must physically reside on the host and are then to be shared with the guest. A special file system driver gets installed while installing the add-on. Shared folders are implemented as network redirectors in the case of a Windows guest, and for Linux and Solaris guest VMs, Guest Additions provide a virtual filesystem.

To share a folder with a guest VM in VirtualBox, you must specify the path of that folder and choose for it a "share name"that the guest can use to access it. Hence, first create the shared folder on the host, then in the guest connect to the host.

The ways by which shared folders can be configured are as follows:

- If the guest VM is running, select **Shared Folders** from the **Devices** menu, or click on the folder icon on the status bar in the bottom right-hand corner
- If the guest VM is not running, go to the **Settings** tab of the VM to configure the shared folder
- You can use the `VBoxManage` command to create the shared folder

Mounting the folders

Folders can be mounted or unmounted manually or automatically for both Windows and Linux.

For Microsoft Windows go to Windows Explorer and look for the shared folder by navigating to **My Networking Places** | **Entire Network** | **VirtualBox Shared Folders**. Right-click on the shared folder and select **Map network drive** from the pop-up screen; you can assign a drive letter to that shared folder.

Starting from VirtualBox 4.0 this folder can be automounted, to automount a folder in Windows the shared folder will receive its own individual drive letter depending on the free drive letters remaining in the guest.

In the Linux guest, use the following command:

```
mount -t vboxsf [-o OPTIONS] sharename mountpoint
```

To mount a shared folder automatically during boot, add the following entry to `/etc/fstab`, or put the following mount command in the `/etc/rc.local` file:

```
sharename mountpoint vboxsf defaults 0 0
```

Memory management techniques

In an environment where multiple VMs exist, Guest Additions can be used to share physical host memory between several guest VMs by reducing their usage of the total amount of memory. Memory overcommitment is a hypervisor feature that allows a VM to use more memory space than what is available in the physical host.

Memory ballooning

To change or reduce the amount of memory allocated to the guest VM, we need to shut down the VM and modify the settings to reduce the memory. With the help of the memory ballooning feature, memory that is allocated for a VM can be given to another machine without shutting down the guest VM.

When memory ballooning is triggered, the VirtualBox Guest Additions allocates physical memory from the guest operating system on the kernel level and locks this memory in the guest. This not only ensures that the guest no longer uses that claimed memory, but also that no other guest application or guest kernel uses it. Then VirtualBox can re-use this reclaimed memory and give it to another VM.

Currently, memory ballooning is only supported through the `VBoxManage` command.

To increase or decrease the size of the memory, use the following command:

```
VBoxManage controlvm "VM name” guestmemoryballoon <n>
```

Here `VM name` is the name of the guest VM (or you can use the universally unique identifier [UUID] of the VM) and `n` is the size of the memory to be allocated in MB from the guest VM.

Page fusion

Page fusion is a technique that transparently and securely shares identical memory pages between VMs, thus eliminating redundant copies of memory pages. This feature helps to avoid memory duplication between several similar, running VMs.

Use the following `VBoxManage` command to enable page fusion:

```
VBoxManage modifyvm "VM name" --pagefusion on
```

`VM name`: name of the guest VM (or you can use the UUID of the VM).

Summary

In this chapter, we have covered installing Oracle Enterprise Linux 6 on VirtualBox, installing Guest Additions, sharing folders, and using some memory management techniques.

In the next chapter we will learn the different networking options available in VirtualBox.

5 Virtual Networking

This chapter provides detailed information on the different networking options available in VirtualBox and how they work.

Virtual networking hardware

Up to eight virtual PCI Ethernet cards are supported in VirtualBox in each guest VM. Out of the eight virtual Ethernet cards, four of them can be configured in the **Network** section of the **Settings** dialog in the VirtualBox GUI.

VirtualBox can virtualize six types of networking hardware, listed as follows:

- AMD PCNet FAST III (Am79C973, which is the default)
- AMD PCNet PCI II (Am79C970A)
- Intel PRO/1000 MT Server (82545EM)
- Intel PRO/1000 T Server (82543GC)
- Intel PRO/1000 MT Desktop (82540EM)
- Paravirtualized network adapter (virtio-net)

AMD PCNet FAST III is the default hardware because it is supported by most of the current operating systems out of the box, and is also supported by the GNU GRUB boot manager. Operating systems such as Microsoft Vista don't ship by default with this driver; that's the reason why Intel PRO/1000 family adapters are chosen for this type of guest OS. The Intel PRO/1000 MT desktop driver works with operating systems such as Microsoft Vista and later versions.

For operating systems such as Microsoft XP, the T Server variant of the Intel PRO/1000 card works out of the box without additional driver installation. The MT Server variant helps to import open virtualization formats from other platforms.

Starting with Version 3.1, VirtualBox provides support for the industry-standard virtio networking drivers, which are a part of the open-source Kernel-based Virtual Machine (KVM) project. If you select this driver, VirtualBox does not virtualize common networking hardware; rather, it expects a special software interface for virtualized environments to be provided by the guest, which avoids the complexity of emulating networking hardware and improving network performance.

The virtio networking drivers are available for Linux kernel versions 2.6.25 or later.

Networking modes

VirtualBox allows you to choose different networking modes for the network adapter used in the VM. The following is the list of available options:

- **Not attached**: This mode implies that the guest VM has an Ethernet card but no network connection; it's like an Ethernet cable that is not connected to the network card.
- **Network Address Translation (NAT)**: This is the default networking mode, and it requires no configuration on either the host or the guest VM. This is great if you just want to browse the web within the guest.
- **Host-only**: In this mode, the guest VMs can interact with each other as well as with the host, but they cannot talk to the outside world. In this mode, a virtual interface is created on the host, which provides connectivity across the guest VM and the host.
- **Bridged networking**: This networking mode helps you to set up routing between the guest and the rest of the network. It does so by using a virtual network interface, which filters data from the physical network adapter, so that the guest and the host can talk to each other, and also with other VMs on the network and in the outside world.
- **Internal networking**. This mode is similar to a bridged network mode when it comes to communicating with the other VMs on the same host, but it is more secure and does not let you communicate with an external network. This mode has two sub-modes:

- **UDP tunnel**: This sub-mode helps to interconnect guest VMs running on different physical hosts directly and transparently over the existing network infrastructure.
- **VDE (Virtual Distributed Ethernet) networking**: This mode helps to connect to a Virtual Distributed Ethernet switch on a Linux or a FreeBSD host. To enable this mode VirtualBox needs compilation because this feature does not come by default with VirtualBox.

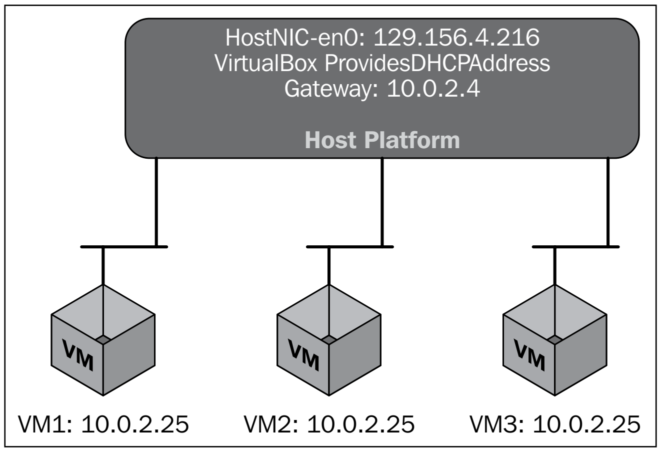
Network Address Translation (NAT)

If you want to access an external network from a guest, VM **Network Address Translation** (**NAT**) helps to do so. There is no extra configuration required, neither from the host nor the guest system, to enable this mode; this works out of the box. This is the default networking mode in VirtualBox.

A guest machine with the NAT mode enabled works similar to a physical host connected to a router. The only exception in this case is in the scenario in which the VirtualBox networking engine routes traffic from and to the guest VM transparently. In VirtualBox, this router is placed between each guest VM and t he host. This separation ensures security, as the default VMs cannot talk to each other. Guest VMs placed in the NAT mode are invisible and unreachable from the outside network, which means that unless you set up a port forwarding request, you can't reach it. We will discuss detailed port forwarding in the next section.

When a guest VM OS sends network traffic, the network frames are received by VirtualBox's NAT engine. Once it receives the frames, the NAT engine extracts the TCP/IP data and resends it using the host OS. So for an application on the same host, or another in the same network, it looks like the data was sent by the VirtualBox application on the host, using an IP address belonging to the same host. VirtualBox listens for replies to the packages sent, and repacks and resends them to the guest machine on its private network. The VM receives its IP and configuration on the private network from a DHCP server built in VirtualBox. The IP address assigned to the VM by the DHCP server is usually on a completely different network than the host. More than one card of a VM can be set up to use NAT.

Let's look at the following example:



When the guest OS boots, DHCP provides the IP. VirtualBox takes care of the DHCP request and also assigns an IP address and the gateway address for routing outbound connections. In this mode, every VM is assigned the same IP address (for example, 10.0.2.25) because each VM thinks it is on its own isolated network. And when it sends the traffic via the gateway (for example, 10.0.2.2), VirtualBox rewrites the packets to make them appear as if they originated from the host rather than the guest. This ensures that the guest VM will work even if the host moves from one network to another.

Port forwarding with NAT

As already discussed, in the NAT mode the guest VM is connected to the private network, which is internal to VirtualBox and not visible to the host. So the network services in the guest VM are not accessible to the host machine or other machines in the same network. With the help of port forwarding, the selected services can be exposed to the outside world. When you configure port forwarding, VirtualBox lists specific ports on the hosts and resends all packets, which are received by the guest OS on the same or a different port.

For an application on the same or a different host on the network, it looks as though the service being proxied is actually running on the host. This also means that you cannot run the same service on the same ports on the host. The **VBoxManage** command line can be used to configure a proxy; also, GUI can be leveraged to configure it as well.

To configure port forwarding, you need to know on which port the service uses for the guest and which ports to use for the host.

For example, if you want to configure port forwarding for TCP traffic received on port 1234 in the host to port 22 in the guest, you can use the following command:

```
VBoxManage modifyvm "VM name" --natpf1 "Testssh,tcp,1234 ,22"
```

To remove this forwarding rule again, use the following command:

```
VBoxManage modifyvm "VM name" --natpf1 delete "Testssh"
```

To specify the guest IP (static), use the following command:

```
VBoxManage modifyvm "VM name" --natpf1 "Testssh,tcp,1234,<IP Address> ,22"
```

To forward all incoming traffic arriving on a local interface from port 1234 to port 22 in the guest, use the following command:

```
VBoxManage modifyvm "VM name" --natpf1 "Testssh,tcp,127.0.0.1,1234 ,22"
```

Despite all the advantages, it has some limitations. For example, it does not support any other protocol (such as PPTP) other than TCP and UDP. And for Linux- and Unix-based hosts, it is not possible to bind to ports below 1024 from applications that are not run by the root. In this case, if you try to configure port forwarding, the guest VM will refuse to start.

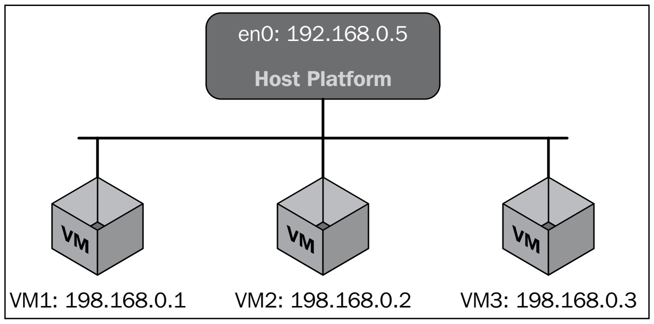
Bridged networking

In this mode, VirtualBox uses a device driver called **net filter** on the host system that filters data from the host's physical network adapter. Net filter helps VirtualBox to translate data from a physical network and inject data into it. Actually, it creates a software network interface.

By using this software adapter, the host sends data to the guest and receives data from it. This means it acts as a bridge between the host and the guest VM. This interface can be used for routing between the guest VM and the network.

A bridge network can be configured from the VirtualBox GUI. To do this, open the **Settings** dialog of a virtual machine, click on the **Network** tab, and select **Bridged Adapter**. In this mode, just like the host, each guest VM can access the physical network. It can access any external services, such as NTP, DNS, and so on.

This network is shown in the following figure:

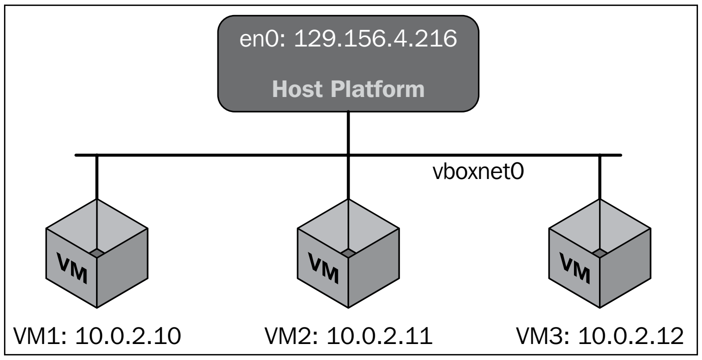


The disadvantage of this mode is that, if your host has multiple physical NICs, then you have to reconfigure the bridge when you move across networks.

Internal networking

If you want to keep one or more guest VMs in an internal network or isolated network, then this mode is the most suitable one. In this mode, VirtualBox ensures that all traffic on the network stays within the host and is only visible to the guest VMs on that virtual network. The network is shown in the following figure.
This mode is similar to bridged networking, apart from the fact that it's secured compared to the bridged mode, because in the bridged mode all the traffic passes through the same physical interface of the host system. This mode is great if you have a requirement to keep two or more guest VMs on the same machine and only communicate with each other, or even hide their data from the host system where it's running and also from the outside world.

The following figure depicts how it works:



This mode allows the guest VMs to work even if the host computer is not connected to any network. There are two ways you can configure this mode:

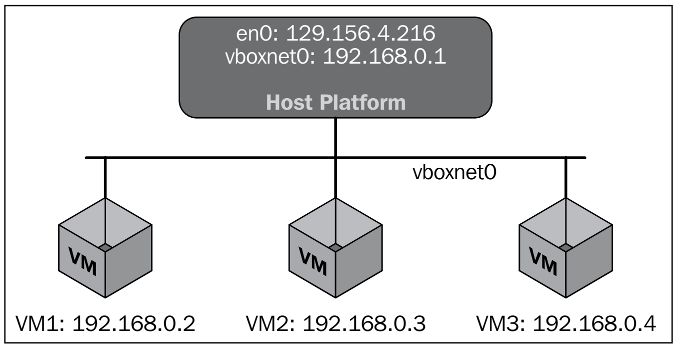
- On the VirtualBox GUI, click on the guest VM **Settings** dialog. In the **Networking** category, select **Internal Networking** from the drop-down list of networking modes. Now select the name of an existing internal network from the list.
- The following command can also be used to configure this:

  ```
  VBoxManage modifyvm "VM name" --nic<x> intnet
  ```

Host-only networking

In this mode, you define which network the guest sits on, and the guest VMs in that network can talk to each other. This is a kind of hybrid between the internal and bridge networking modes.

To configure it, click on the **Network** tab, select **Host-only Adapter** from the drop-down list of networking modes, and select the name. This name is the network name, and the guest VMs in that network can see each other. But the external machines can't talk to these guest VMs.



In this example, vboxnet0 is the network on which the guest VMs are.

Bandwidth limitation for network I/O

VirtualBox helps in limiting the maximum bandwidth used for network transmission. The VBoxManage command can be used to set the limit.

The following example creates a bandwidth group named `BandwidthLimit`, sets the limit to 10 Mbit/s, and assigns the group to the first and second network adapters of the VM named `LinuxVM`:

```
VBoxManage bandwidthctl "LinuxVM" add  BandwidthLimit --type network
--limit 10m
VBoxManage modifyvm " LinuxVM " --nicbandwidthgroup1 BandwidthLimit
VBoxManage modifyvm " LinuxVM " --nicbandwidthgroup2 Limit BandwidthLimit
```

The following command can be used to change the limits for each group while VM is running:

```
VBoxManage bandwidthctl "LinuxVM" set Limit --limit 50k
```

To disable shaping for the second adapter of VM, use the following command:

```
VBoxManage modifyvm "LinuxVM" –nicbandwidthgroup2 none
```

Summary

In this chapter, we have covered the different networking modes available in VirtualBox and how they are used. In the next chapter, we will learn different storage options available for VirtualBox.

6
Virtual Storage

This chapter provides detailed information on the different virtual storage options available in VirtualBox and how they work.

Virtual disk

When you create a guest VM, it expects to see a hard disk where it stores the data. VirtualBox needs to present real storage to this guest VM, and this guest VM sees the storage as a virtual hard disk.

There are three ways to achieve this:

- Large image files can be created on the physical hard disk and these image files will be used by VirtualBox, which in turn presents them to the guest VM as a virtual hard disk
- You can present an iSCSI storage to the VirtualBox
- There is an advanced functionality by which you can present the host disks directly to the guest VM

Each of the preceding options will be described in detail.

Each of the virtual storage device types presented in this section needs to be connected to the virtual hard disk controller, which VirtualBox presents to a virtual machine as its hard disk. There are different disk controller standards, and VirtualBox supports most of the commonly used disk controllers. In the next sections, we will discuss the different disk controllers.

Hard disk controllers

If we look at the physical servers or computers, all the devices (for example, DVD drives and hard drives) are connected to the physical hard disk controller, which is responsible for all the I/O operations of these devices. In contrast, VirtualBox emulated these hard disk controllers. VirtualBox emulates the most commonly used hard disk controllers such as IDE, SATA, SCSI, and SAS, which are described as follows:

- **IDE** (**ATA**): This was the most widely used controller standard, but it has certain limitations and is now being replaced by the SATA standard. These controllers are backward compatible. So, if a guest operating system doesn't support SCSI or SATA, it would fall back to use IDE (ATA). For each guest VM in VirtualBox, there might be one IDE controller enabled that supports or allows up to four storage devices to be connected to a guest VM. It does so by using the primary master, secondary master, primary slave, and secondary slave slots. Out of these four storage devices, the secondary device master slot is preconfigured for CD/DVD devices. If additional IDE disks are required, the VirtualBox SATA controller for the VM must be enabled in the SATA IDE compatibility mode.

  If you migrate a guest VM from another virtualization solution to VirtualBox, then you are expected to provide the exact controller type. In VirtualBox, you can select the PIIX3, PIIX4, or ICH6 controller type.

- **Serial ATA** (**SATA/AHCI**): This is now the most widely used and latest drive controller standard. SATA provides higher levels of storage performance, unlike the IDE standard. It does not impose restrictions on the number of devices that can be connected or attached to a single disk controller. Another big advantage of using SATA is that disks can be added or removed on the fly without shutting down the system. These disk controllers can be configured to run in IDE compatibility mode to allow access from guest operating systems that lack **Advanced Host Controller Interface** (**AHCI**) support. A guest VM can be configured with up to 30 SATA ports in VirtualBox. The first four of these SATA ports are configured by default to operate in the IDE compatibility mode. Considering all the advantages and features, VirtualBox uses the SATA controller as the default controller for any newly created VM.

  VirtualBox creates one SATA controller by default, and the default disk that is created with the new guest VM is attached to this controller.

  If it's not enabled for some guest VMs, you can enable it by clicking the **Storage** tab under the **Settings** dialog. Then, click on the **Add controller** button and select **Add SATA Controller**.

- **Small Computer System Interface** (**SCSI**): This is now a popular storage connection standard. This standard is based on a bus topology allowing up to 15 storage devices to be connected to a disk controller in a daisy chain configuration. Each of the devices is assigned a unique SCSI identifier, which distinguishes the devices from each other on the SCSI bus. VirtualBox provides emulation for BusLogic and LSI Logic SCSI disk controllers.

  To enable a SCSI controller, select the VM under **Settings**, click on the **Storage** tab, click on the **Add Controller** button under the **Storage Tree** section, and then select **Add SCSI Controller**.

- **Serial Attached SCSI** (**SAS**): Like SCSI, this standard uses the SCSI command set. In this type of connection, serial cables are used instead of parallel cables. This results in simplified physical device connections. It is more reliable and faster compared to SCSI.

  To support high-end guests that require SAS controllers, VirtualBox emulates a LSI Logic SAS controller. This can be enabled in the same way as a SCSI controller. This standard supports up to eight devices, which can be connected to the SAS controller.

To sum up, VirtualBox provides the following categories of virtual storage slots, provided they are enabled. In this case, they are supported by the guest operating system:

- Four slots attached to the traditional IDE controller; avoid this controller unless this is the only controller supported by your guest OS
- 30 slots supported by the SATA controller
- 15 slots supported by the SCSI controller
- Eight slots supported by the SAS controller

Disk image files (VDI, VMDK, VHD, and HDD)

Guest VMs accesses virtual disk images that are stored on the physical hard disk in the host computer. When a guest VM tries to access these disk images, the read/write disk access is redirected by VirtualBox to the virtual disk image.

A virtual disk image is similar to a physical hard disk. It has size that is basically specified when you create the VM. Upon creation, the entire size of the virtual disk might be used for the image or the image might be dynamically expanded. These two types refer to thick and thin provisioning.

There are four types of disk images supported by VirtualBox:

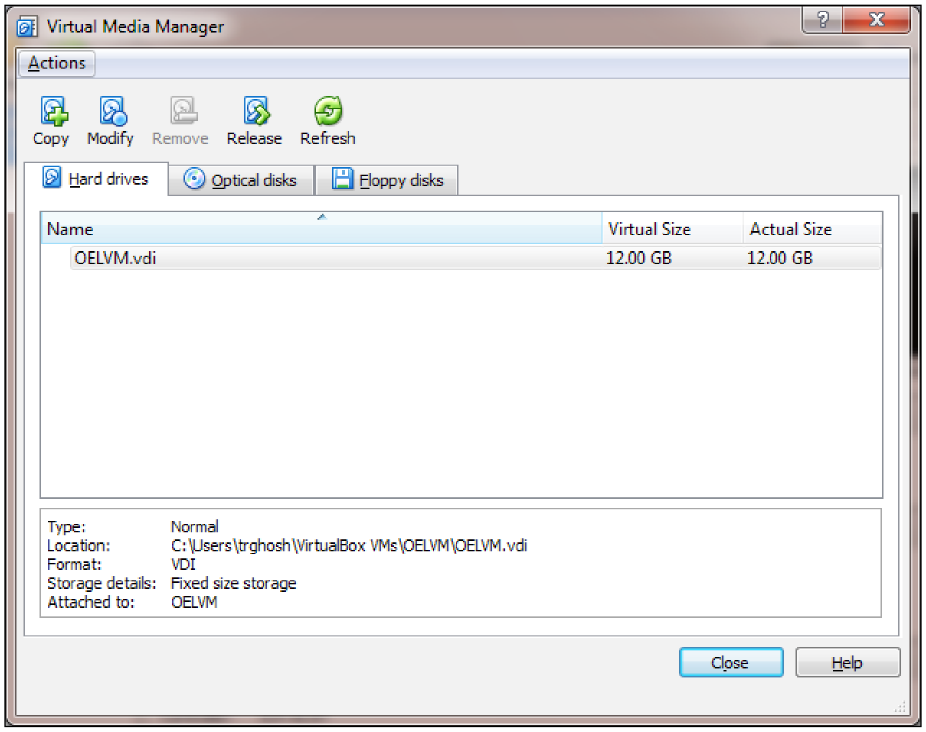
- **VDI**: Also known as **Virtual Disk Image**
- **VMDK**: The VMware format of a virtual hard disk
- **VHD**: The Microsoft format of a virtual hard disk
- **HDD**: This is the image file of Parallels Version 2 format; VirtualBox also supports this format

You can create two kinds of disk images: fixed size images or dynamically allocated images:

- **Dynamically expanding**: When you select this option, the disk image will be created with a minimal size, but it grows automatically when more space is needed by the guest operating system. However, during the creation of a dynamically expanding disk, you specify a maximum capacity, which is the maximum capacity the disk will expand to. The main advantage of using this feature is that you are not wasting the disk space on the host before it is required, but there is a disadvantage associated with it in that it results in a slow performance if disk expansion happens pretty frequently.
- **Fixed size**: When you select this option, a disk image file is created with the specified size. This option improves the write performance, but the disadvantage is that it will take a long time to create the disk image in the first place and you might end up wasting the disk space if it's not required.

VirtualBox keeps track of all the disk types, such as hard disk, CD/DVD ROM, and floppy disks, which are used by the guest VM. These are known as **Known Media**. **Virtual Media Manager** is used to either view or change these known media. VMM can be accessed from the **File** menu in the VirtualBox main window.

Virtual Media Manager looks like the following screenshot:



Name	Virtual Size	Actual Size
OELVM.vdi	12.00 GB	12.00 GB

Type:	Normal
Location:	C:\Users\trghosh\VirtualBox VMs\OELVM\OELVM.vdi
Format:	VDI
Storage details:	Fixed size storage
Attached to:	OELVM

Known media are grouped into three categories as follows:

- Hard disk images can be either a VDI native format for VirtualBox or any third-party format, such as VMDK
- CD/DVD images in the ISO format
- Floppy disk images in the standard RAW format

Virtual Media Manager also allows you to copy, modify, remove, or release a disk image.

Image write modes

We have already learned about fixed and dynamic disk images, and apart from this there are multiple options available that dictate the way data is written to the disk drive.

The options are listed as follows:

- **Normal**: When you create a disk image, the default setting is **Normal**. In this mode, any operation such as read or write performed by the guest VM is performed on the disk image itself. Disk images in the normal mode can only be associated with one running guest VM. Also, any data written to the disk is permanent even if you power off the guest VM.
- **Immutable**: This mode ensures that the data written on the disk image is read-only. But when a guest VM writes any data to the disk, it gets stored in a separate differencing disk image. When the guest VM is reset or powered off, all the data written in the differencing disk will be lost. So, the immutable disk is untouched. This approach is used when you want to share the same disk image across multiple VMs, each being assigned its own differencing disk during run time.
- **Multi-attach**: This setting is very much similar to how an **Immutable** image works, apart from the fact that the differencing image is not reset every time the guest virtual machine starts.
- **Writethrough**: These disk images allow read/write operations to be performed on the disk image. When the VM is powered off or reset, the state of the disk image is not saved; therefore it reverts to its original state next time the VM starts.
- **Sharable**: These are a variant of write through hard disks but the state is not saved when a snapshot is taken and can't be restored when the snapshot is restored. You can attach this disk to multiple guest VMs that run concurrently and access this file concurrently. But only fixed sized images are candidates to play the role of a shareable hard disk. If you try to attach dynamic images, they will be rejected.
- **Read-only image**: These are used by a CD-ROM/DVD image by default as they are read-only in nature.

Cloning a virtual disk image

Cloning a disk helps to duplicate hard disk image files on the same host to quickly reproduce a second VM as the primary image.

VirtualBox assigns a **Unique Identity Number** (**UUID**) to each disk image, which is also stored inside the image. So if you have multiple disks with the same UUID number, VirtualBox will refuse to work.

To clone the disk, use the following command:

```
VBoxManage clonehd <source image file> <target image file> -format
<format>
```

To check the disk ID, the following command can be used:

```
hdparm -i  <device>
```

Bandwidth control for disk images

Starting with Version 4.0, VirtualBox allows for limiting the maximum bandwidth used for asynchronous I/O. Additionally, it supports sharing limits through bandwidth groups for several images. It is possible to have more than one such limit.

Limits are configured through the `VBoxManage` command-line interface. The following example creates a bandwidth group named `Limit`, sets the limit to `20` MB/s, and assigns the group to the attached disks of the VM:

```
VBoxManage bandwidthctl "VM name" add Limit --type disk --limit 20M

VBoxManage storageattach "VM name" --controller "SATA" --port 0 --device
0 --type hdd --medium disk1.vdi --bandwidthgroup Limit

VBoxManage storageattach "VM name" --controller "SATA" --port 1 --device
0 --type hdd --medium disk2.vdi --bandwidthgroup Limit
```

All disks in a group share the bandwidth limit, meaning that in the preceding example the bandwidth of both images combined can never exceed `20` MB/s. However, if one disk doesn't require bandwidth, the other can use the remaining bandwidth of its group.

The limits for each group can be changed while the VM is running, with changes being picked up immediately. The following example changes the limit for the group created in the preceding example to `10` MB/s:

```
VBoxManage bandwidthctl "VM name" set Limit --limit 10M
```

Summary

In this chapter, we have covered the different storage options available and their types, settings, and usage. We have also discussed how to clone a disk and control bandwidth by the command-line utility.

This is the last chapter of the book, and I am sure this book will help you to understand virtualization better. After reading this book, you will be able to easily install, configure, manage, and work with VirtualBox, which is a true free, open-source, and powerful virtualization tool.

Index