
# Oracle® VM VirtualBox®

## User Manual

Version 6.0.8_Debian



http://www.virtualbox.org

# Contents

# Preface

The *Oracle VM VirtualBox User Manual* provides an introduction to using Oracle VM VirtualBox. The manual provides information on how to install Oracle VM VirtualBox and use it to create and configure virtual machines.

## Audience

This document is intended for both new and existing users of Oracle VM VirtualBox. It is assumed that readers are familiar with Web technologies and have a general understanding of Windows and UNIX platforms.

## Related Documents

The documentation for this product is available at:
`https://www.oracle.com/technetwork/server-storage/virtualbox/documentation/index.html`

## Conventions

The following text conventions are used in this document:

- **boldface**: Boldface type indicates graphical user interface elements associated with an action, or terms defined in text or the glossary.
- *italic*: Italic type indicates book titles, emphasis, or placeholder variables for which you supply particular values.
- `monospace`: Monospace type indicates commands within a paragraph, URLs, code in examples, text that appears on the screen, or text that you enter.

## Documentation Accessibility

For information about Oracle's commitment to accessibility, visit the Oracle Accessibility Program website at
`http://www.oracle.com/pls/topic/lookup?ctx=acc&id=docacc`.

## Access to Oracle Support

Oracle customers that have purchased support have access to electronic support through My Oracle Support. For information, visit
`http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info` or visit `http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs` if you are hearing impaired.

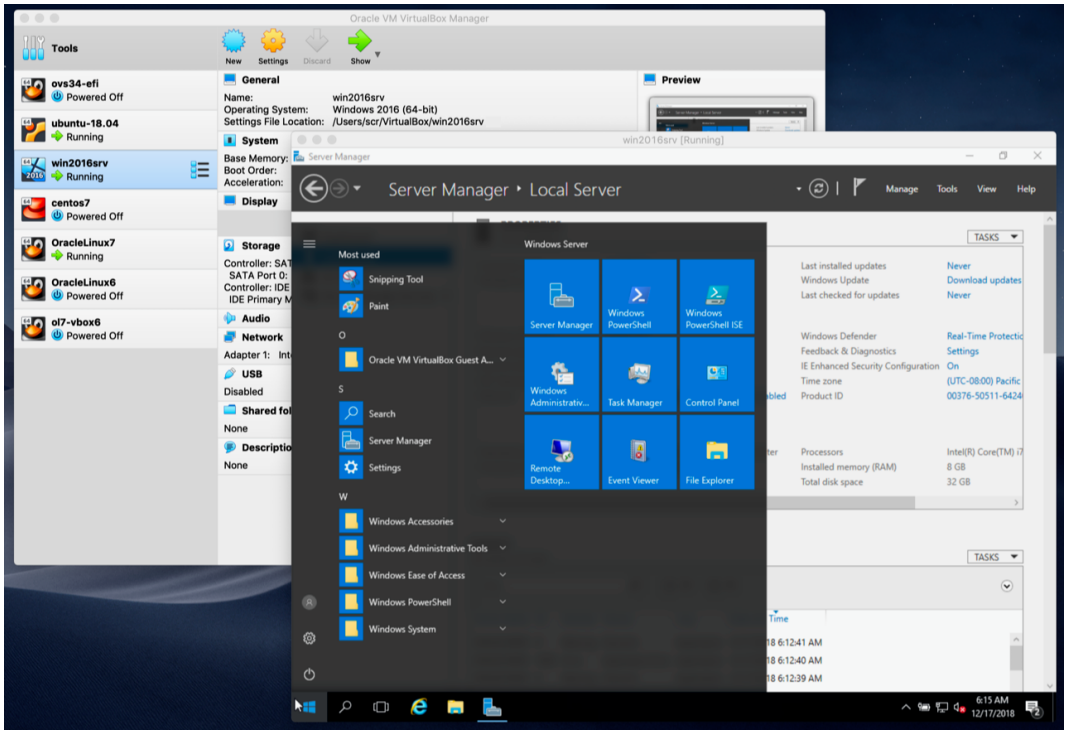
# 1 First Steps

Welcome to Oracle VM VirtualBox.

Oracle VM VirtualBox is a cross-platform virtualization application. What does that mean? For one thing, it installs on your existing Intel or AMD-based computers, whether they are running Windows, Mac OS X, Linux, or Oracle Solaris operating systems (OSes). Secondly, it extends the capabilities of your existing computer so that it can run multiple OSes, inside multiple virtual machines, at the same time. As an example, you can run Windows and Linux on your Mac, run Windows Server 2016 on your Linux server, run Linux on your Windows PC, and so on, all alongside your existing applications. You can install and run as many virtual machines as you like. The only practical limits are disk space and memory.

Oracle VM VirtualBox is deceptively simple yet also very powerful. It can run everywhere from small embedded systems or desktop class machines all the way up to datacenter deployments and even Cloud environments.

The following screenshot shows how Oracle VM VirtualBox, installed on an Apple Mac OS X computer, is running Windows Server 2016 in a virtual machine window.




In this User Manual, we will begin simply with a quick introduction to virtualization and how to get your first virtual machine running with the easy-to-use Oracle VM VirtualBox graphical user interface. Subsequent chapters will go into much more detail covering more powerful tools and features, but fortunately, it is not necessary to read the entire User Manual before you can use Oracle VM VirtualBox.

You can find a summary of Oracle VM VirtualBox's capabilities in chapter 1.3, *Features Overview*, page 3. For existing Oracle VM VirtualBox users who just want to find out what is new in this release, see the chapter 15, *Change Log*, page 305.

## 1.1 Why is Virtualization Useful?

The techniques and features that Oracle VM VirtualBox provides are useful in the following scenarios:

- **Running multiple operating systems simultaneously.** Oracle VM VirtualBox enables you to run more than one OS at a time. This way, you can run software written for one OS on another, such as Windows software on Linux or a Mac, without having to reboot to use it. Since you can configure what kinds of *virtual* hardware should be presented to each such OS, you can install an old OS such as DOS or OS/2 even if your real computer's hardware is no longer supported by that OS.

- **Easier software installations.** Software vendors can use virtual machines to ship entire software configurations. For example, installing a complete mail server solution on a real machine can be a tedious task. With Oracle VM VirtualBox, such a complex setup, often called an *appliance*, can be packed into a virtual machine. Installing and running a mail server becomes as easy as importing such an appliance into Oracle VM VirtualBox.

- **Testing and disaster recovery.** Once installed, a virtual machine and its virtual hard disks can be considered a *container* that can be arbitrarily frozen, woken up, copied, backed up, and transported between hosts.

  On top of that, with the use of another Oracle VM VirtualBox feature called *snapshots*, one can save a particular state of a virtual machine and revert back to that state, if necessary. This way, one can freely experiment with a computing environment. If something goes wrong, such as prolems after installing software or infecting the guest with a virus, you can easily switch back to a previous snapshot and avoid the need of frequent backups and restores.

  Any number of snapshots can be created, allowing you to travel back and forward in virtual machine time. You can delete snapshots while a VM is running to reclaim disk space.

- **Infrastructure consolidation.** Virtualization can significantly reduce hardware and electricity costs. Most of the time, computers today only use a fraction of their potential power and run with low average system loads. A lot of hardware resources as well as electricity is thereby wasted. So, instead of running many such physical computers that are only partially used, one can pack many virtual machines onto a few powerful hosts and balance the loads between them.

## 1.2 Some Terminology

When dealing with virtualization, and also for understanding the following chapters of this documentation, it helps to acquaint oneself with a bit of crucial terminology, especially the following terms:

- *Host operating system (host OS)*. This is the OS of the physical computer on which Oracle VM VirtualBox was installed. There are versions of Oracle VM VirtualBox for Windows, Mac OS X, Linux, and Oracle Solaris hosts. See chapter 1.4, *Supported Host Operating Systems*, page 5.

  Most of the time, this manual discusses all Oracle VM VirtualBox versions together. There may be platform-specific differences which we will point out where appropriate.

- *Guest operating system (guest OS)*. This is the OS that is running inside the virtual machine. Theoretically, Oracle VM VirtualBox can run any x86 OS. such as DOS, Windows, OS/2, FreeBSD, and OpenBSD. But to achieve near-native performance of the guest code on your machine, we had to go through a lot of optimizations that are specific to certain OSes. So

while your favorite OS *may* run as a guest, we officially support and optimize for a select few, which include the most common OSes.

See chapter 3.1, *Supported Guest Operating Systems*, page 40.

- *Virtual machine (VM)*. This is the special environment that Oracle VM VirtualBox creates for your guest OS while it is running. In other words, you run your guest OS *in* a VM. Normally, a VM will be shown as a window on your computer's desktop, but depending on which of the various frontends of Oracle VM VirtualBox you use, it can be displayed in full screen mode or remotely on another computer.

  In a more abstract way, internally, Oracle VM VirtualBox thinks of a VM as a set of parameters that determine its behavior. They include hardware settings, such as: how much memory the VM should have, what hard disks Oracle VM VirtualBox should virtualize through which container files, what CDs are mounted. They also include state information, such as: whether the VM is currently running, saved, if the VM has snapshots. These settings are mirrored in the VirtualBox Manager window, as well as the `VBoxManage` command. See chapter 8, *VBoxManage*, page 120. In other words, a VM is also what you can see in its **Settings** dialog.

- *Guest Additions*. This refers to special software packages which are shipped with Oracle VM VirtualBox but designed to be installed *inside* a VM to improve performance of the guest OS and to add extra features. See chapter 4, *Guest Additions*, page 62.

## 1.3 Features Overview

The following is a brief outline of Oracle VM VirtualBox's main features:

- **Portability.** Oracle VM VirtualBox runs on a large number of 32-bit and 64-bit host OS. See chapter 1.4, *Supported Host Operating Systems*, page 5.

  Oracle VM VirtualBox is a so-called *hosted* hypervisor, sometimes referred to as a *type 2* hypervisor. Whereas a *bare-metal* or *type 1* hypervisor would run directly on the hardware, Oracle VM VirtualBox requires an existing OS to be installed. It can thus run alongside existing applications on that host.

  To a very large degree, Oracle VM VirtualBox is functionally identical on all of the host platforms, and the same file and image formats are used. This enables you to run virtual machines created on one host on another host with a different host OS. For example, you can create a virtual machine on Windows and then run it under Linux.

  In addition, virtual machines can easily be imported and exported using the Open Virtualization Format (OVF), an industry standard created for this purpose. You can even import OVFs that were created with a different virtualization software. See chapter 1.15, *Importing and Exporting Virtual Machines*, page 21.

- **No hardware virtualization required.** For many scenarios, Oracle VM VirtualBox does not require the processor features built into newer hardware like Intel VT-x or AMD-V. As opposed to many other virtualization solutions, you can therefore use Oracle VM VirtualBox even on older hardware where these features are not present. See chapter 10.3, *Hardware vs. Software Virtualization*, page 270.

- **Guest Additions: shared folders, seamless windows, 3D virtualization.** The Oracle VM VirtualBox Guest Additions are software packages which can be installed *inside* of supported guest systems to improve their performance and to provide additional integration and communication with the host system. After installing the Guest Additions, a virtual machine will support automatic adjustment of video resolutions, seamless windows, accelerated 3D graphics and more. See chapter 4, *Guest Additions*, page 62.

In particular, Guest Additions provide for "shared folders", which let you access files from the host system from within a guest machine. See chapter 4.3, *Shared Folders*, page 69.

- **Great hardware support.** Among others, Oracle VM VirtualBox supports the following:
  - **Guest multiprocessing (SMP).** Oracle VM VirtualBox can present up to 32 virtual CPUs to each virtual machine, irrespective of how many CPU cores are physically present on your host.
  - **USB device support.** Oracle VM VirtualBox implements a virtual USB controller and enables you to connect arbitrary USB devices to your virtual machines without having to install device-specific drivers on the host. USB support is not limited to certain device categories. See chapter 3.11.1, *USB Settings*, page 56.
  - **Hardware compatibility.** Oracle VM VirtualBox virtualizes a vast array of virtual devices, among them many devices that are typically provided by other virtualization platforms. That includes IDE, SCSI and SATA hard disk controllers, several virtual network cards and sound cards, virtual serial and parallel ports and an Input/Output Advanced Programmable Interrupt Controller (I/O APIC), which is found in many modern PC systems. This eases cloning of PC images from real machines and importing of third-party virtual machines into Oracle VM VirtualBox.
  - **Full ACPI support.** The Advanced Configuration and Power Interface (ACPI) is fully supported by Oracle VM VirtualBox. This eases cloning of PC images from real machines or third-party virtual machines into Oracle VM VirtualBox. With its unique *ACPI power status support*, Oracle VM VirtualBox can even report to ACPI-aware guest OSes the power status of the host. For mobile systems running on battery, the guest can thus enable energy saving and notify the user of the remaining power, for example in full screen modes.
  - **Multiscreen resolutions.** Oracle VM VirtualBox virtual machines support screen resolutions many times that of a physical screen, allowing them to be spread over a large number of screens attached to the host system.
  - **Built-in iSCSI support.** This unique feature enables you to connect a virtual machine directly to an iSCSI storage server without going through the host system. The VM accesses the iSCSI target directly without the extra overhead that is required for virtualizing hard disks in container files. See chapter 5.10, *iSCSI Servers*, page 95.
  - **PXE Network boot.** The integrated virtual network cards of Oracle VM VirtualBox fully support remote booting using the Preboot Execution Environment (PXE).
- **Multigeneration branched snapshots.** Oracle VM VirtualBox can save arbitrary snapshots of the state of the virtual machine. You can go back in time and revert the virtual machine to any such snapshot and start an alternative VM configuration from there, effectively creating a whole snapshot tree. See chapter 1.11, *Snapshots*, page 17. You can create and delete snapshots while the virtual machine is running.
- **VM groups.** Oracle VM VirtualBox provides a groups feature that enables the user to organize and control virtual machines collectively, as well as individually. In addition to basic groups, it is also possible for any VM to be in more than one group, and for groups to be nested in a hierarchy. This means you can have groups of groups. In general, the operations that can be performed on groups are the same as those that can be applied to individual VMs: Start, Pause, Reset, Close (Save state, Send Shutdown, Poweroff), Discard Saved State, Show in File System, Sort.
- **Clean architecture and unprecedented modularity.** Oracle VM VirtualBox has an extremely modular design with well-defined internal programming interfaces and a clean separation of client and server code. This makes it easy to control it from several interfaces at once. For example, you can start a VM simply by clicking on a button in the Oracle VM

VirtualBox graphical user interface and then control that machine from the command line, or even remotely. See chapter 1.17, *Alternative Front-Ends*, page 28.

Due to its modular architecture, Oracle VM VirtualBox can also expose its full functionality and configurability through a comprehensive **software development kit (SDK),** which enables integration of Oracle VM VirtualBox with other software systems. See chapter 11, *Oracle VM VirtualBox Programming Interfaces*, page 276.

- **Remote machine display.** The VirtualBox Remote Desktop Extension (VRDE) enables high-performance remote access to any running virtual machine. This extension supports the Remote Desktop Protocol (RDP) originally built into Microsoft Windows, with special additions for full client USB support.

  The VRDE does not rely on the RDP server that is built into Microsoft Windows. Instead, the VRDE is plugged directly into the virtualization layer. As a result, it works with guest OSes other than Windows, even in text mode, and does not require application support in the virtual machine either. The VRDE is described in detail in chapter 7.1, *Remote Display (VRDP Support)*, page 110.

  On top of this special capacity, Oracle VM VirtualBox offers you more unique features:

  - **Extensible RDP authentication.** Oracle VM VirtualBox already supports Winlogon on Windows and PAM on Linux for RDP authentication. In addition, it includes an easy-to-use SDK which enables you to create arbitrary interfaces for other methods of authentication. See chapter 7.1.5, *RDP Authentication*, page 115.
  - **USB over RDP.** Using RDP virtual channel support, Oracle VM VirtualBox also enables you to connect arbitrary USB devices locally to a virtual machine which is running remotely on a Oracle VM VirtualBox RDP server. See chapter 7.1.4, *Remote USB*, page 114.

## 1.4 Supported Host Operating Systems

Currently, Oracle VM VirtualBox runs on the following host OSes:

- **Windows hosts (64-bit):**
  - Windows 7
  - Windows 8
  - Windows 8.1
  - Windows 10 RTM (1507) build 10240
  - Windows 10 November Update (1511) build 10586
  - Windows 10 Anniversary Update (1607) build 14393
  - Windows 10 Creators Update (1703) build 15063
  - Windows 10 Fall Creators Update (1709) build 16299
  - Windows 10 April 2018 Update (1803) build 17134
  - Windows 10 October 2018 Update (1809) build 17763
  - Windows Server 2008 R2
  - Windows Server 2012
  - Windows Server 2012 R2
  - Windows Server 2016
  - Windows Server 2019

- **Mac OS X hosts (64-bit):**
  - 10.12 (Sierra)
  - 10.13 (High Sierra)
  - 10.14 (Mojave)

  Intel hardware is required. See also chapter 14, *Known Limitations*, page 301.

- **Linux hosts (64-bit).** Includes the following:
  - Ubuntu 16.04 LTS, 18.04 LTS and 18.10
  - Debian GNU/Linux 9 ("Stretch")
  - Oracle Linux 6 and 7
  - Redhat Enterprise Linux 6 and 7
  - Fedora 28 and 29
  - Gentoo Linux
  - SUSE Linux Enterprise server 12 and 15
  - openSUSE Leap 42.3 and 15.0

  It should be possible to use Oracle VM VirtualBox on most systems based on Linux kernel 2.6 or 3.x using either the Oracle VM VirtualBox installer or by doing a manual installation. See chapter 2.3, *Installing on Linux Hosts*, page 32. However, the formally tested and supported Linux distributions are those for which we offer a dedicated package.

  Note that Linux 2.4-based host OSes are no longer supported.

- **Oracle Solaris hosts (64-bit only).** The following versions are supported with the restrictions listed in chapter 14, *Known Limitations*, page 301:
  - Oracle Solaris 11

Note that the above list is informal. Oracle support for customers who have a support contract is limited to a subset of the listed host OSes. Also, any feature which is marked as *experimental* is not supported. Feedback and suggestions about such features are welcome.

## 1.5 Host CPU Requirements

SSE2 is required, starting with Oracle VM VirtualBox version 5.2.10 and version 5.1.24.

## 1.6 Installing Oracle VM VirtualBox and Extension Packs

Oracle VM VirtualBox comes in many different packages, and installation depends on your host OS. If you have installed software before, installation should be straightforward. On each host platform, Oracle VM VirtualBox uses the installation method that is most common and easy to use. If you run into trouble or have special requirements, see chapter 2, *Installation Details*, page 29 for details about the various installation methods.

Oracle VM VirtualBox is split into the following components:

- **Base package.** The base package consists of all open source components and is licensed under the GNU General Public License V2.
- **Extension packs.** Additional extension packs can be downloaded which extend the functionality of the Oracle VM VirtualBox base package. Currently, Oracle provides a single extension pack, available from: `http://www.virtualbox.org`. The extension pack provides the following added functionality:

1. The virtual USB 2.0 (EHCI) device. See chapter 3.11.1, *USB Settings*, page 56.
2. The virtual USB 3.0 (xHCI) device. See chapter 3.11.1, *USB Settings*, page 56.
3. VirtualBox Remote Desktop Protocol (VRDP) support. See chapter 7.1, *Remote Display (VRDP Support)*, page 110.
4. Host webcam passthrough. See chapter 9.6, *Webcam Passthrough*, page 215.
5. Intel PXE boot ROM.
6. Experimental support for PCI passthrough on Linux hosts. See chapter 9.5, *PCI Passthrough*, page 213.
7. Disk image encryption with AES algorithm. See chapter 9.30, *Encryption of Disk Images*, page 251.

Oracle VM VirtualBox extension packages have a `.vbox-extpack` file name extension. To install an extension, simply double-click on the package file and a **Network Operations Manager** window is shown to guide you through the required steps.

To view the extension packs that are currently installed, start the VirtualBox Manager, as shown in chapter 1.7, *Starting Oracle VM VirtualBox*, page 7. From the **File** menu, select **Preferences**. In the window that displays, go to the **Extensions** category. This shows you the extensions which are currently installed, and enables you to remove a package or add a new package.

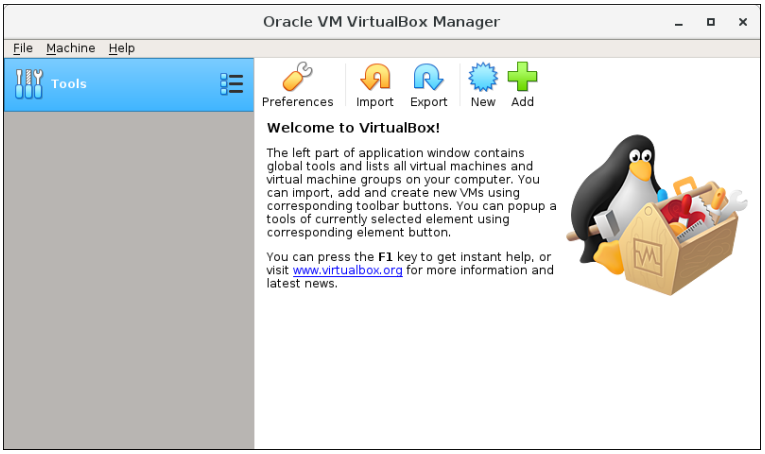
Alternatively, you can use the `VBoxManage` command line. See chapter 8.43, *VBoxManage extpack*, page 201.

## 1.7 Starting Oracle VM VirtualBox

After installation, you can start Oracle VM VirtualBox as follows:

- On a Windows host, in the **Programs** menu, click on the item in the **VirtualBox** group. On Vista or Windows 7, you can also enter `VirtualBox` in the search box of the **Start** menu.
- On a Mac OS X host, in the Finder, double-click on the **VirtualBox** item in the Applications folder. You may want to drag this item onto your Dock.
- On a Linux or Oracle Solaris host, depending on your desktop environment, an Oracle VM VirtualBox item may have been placed in either the System or System Tools group of your **Applications** menu. Alternatively, you can enter `VirtualBox` in a terminal window.

When you start Oracle VM VirtualBox for the first time, a window like the following is displayed:

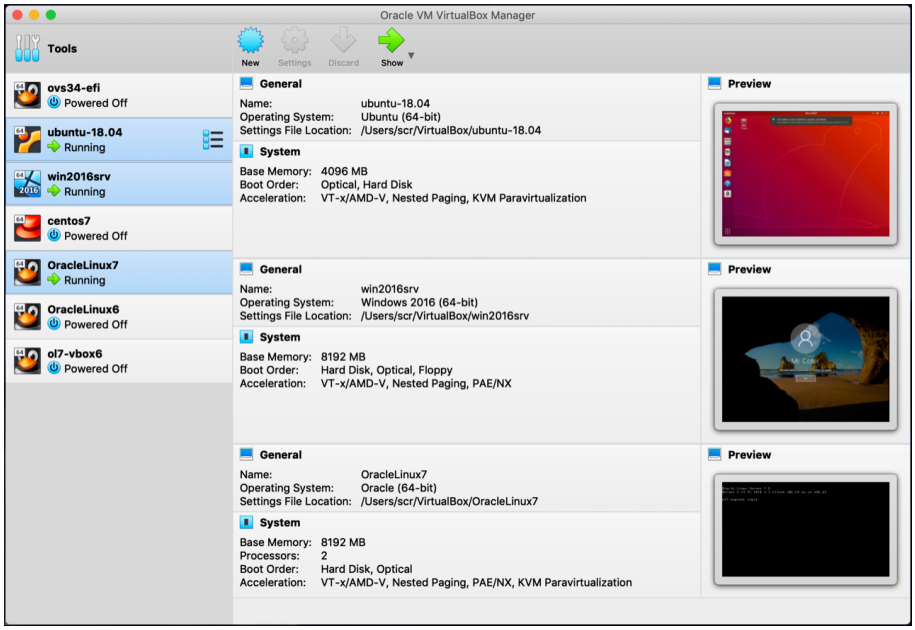


This window is called the **VirtualBox Manager**. The left pane will later list all your virtual machines. Since you have not yet created any virtual machines, this list is empty. The **Tools** button provides access to user tools, such as the Virtual Media Manager.

The pane on the right displays the properties of the currently selected virtual machine. Since you do not have any machines yet, the pane displays a welcome message.

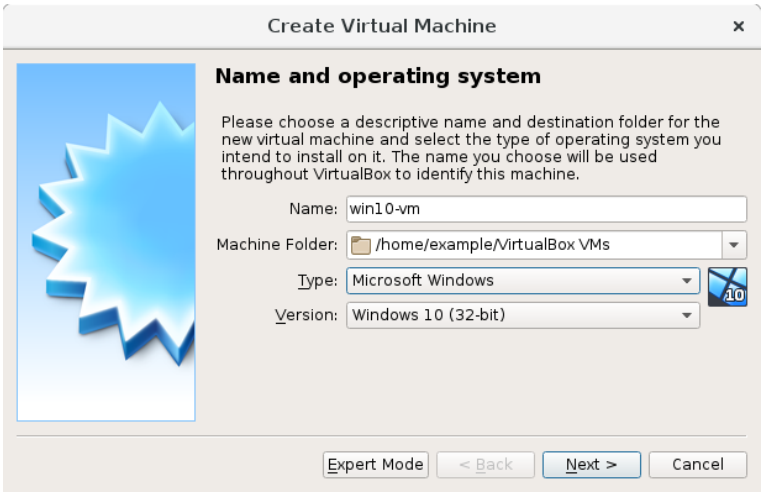
The buttons on the right pane are used to create and work with VMs.

The following figure gives an idea of what Oracle VM VirtualBox might look like after you have created some VMs.




## 1.8 Creating Your First Virtual Machine

Click **New** in the VirtualBox Manager window. A wizard is shown, to guide you through setting up a new virtual machine (VM).




On the following pages, the wizard will ask you for the bare minimum of information that is needed to create a VM, in particular:

1. The **Name** of the VM will later be shown in the machine list of the VirtualBox Manager window, and it will be used for the VM's files on disk. Even though any name can be used, bear in mind that if you create a few VMs, you will appreciate if you have given your VMs rather informative names."My VM" would thus be less useful than "Windows XP SP2 with OpenOffice", for example.

2. The **Machine Folder** is the location where VMs are stored on your computer. The default folder location is shown.

3. For **Operating System Type** select the OS that you want to install later. The supported OSes are grouped. If you want to install something very unusual that is not listed, select **Other**. Depending on your selection, Oracle VM VirtualBox will enable or disable certain VM settings that your guest OS may require. This is particularly important for 64-bit guests. See chapter 3.1.2, *64-bit Guests*, page 42. It is therefore recommended to always set it to the correct value.

4. On the next page, select the **Memory (RAM)** that Oracle VM VirtualBox should allocate every time the virtual machine is started. The amount of memory given here will be taken away from your host machine and presented to the guest OS, which will report this size as the virtual computer's installed RAM.

   Choose this setting carefully. The memory you give to the VM will not be available to your host OS while the VM is running, so do not specify more than you can spare. For example, if your host machine has 1 GB of RAM and you enter 512 MB as the amount of RAM for a particular virtual machine, while that VM is running, you will only have 512 MB left for all the other software on your host. If you run two VMs at the same time, even more memory will be allocated for the second VM, which may not even be able to start if that memory is not available. On the other hand, you should specify as much as your guest OS and your applications will require to run properly.

   A Windows XP guest will require at least a few hundred MB of RAM to run properly, and Windows Vista will not install with less than 512 MB. If you want to run graphics-intensive applications in your VM, you may require even more RAM.

   As a rule of thumb, if you have 1 GB of RAM or more in your host computer, it is usually safe to allocate 512 MB to each VM. In any case, make sure you always have at least 256 to 512 MB of RAM left on your host OS. Otherwise you may cause your host OS to excessively swap out memory to your hard disk, effectively bringing your host system to a standstill.

   As with the other settings, you can change this setting later, after you have created the VM.

5. Next, you must specify a **Virtual Hard Disk** for your VM.

   There are many and potentially complicated ways in which Oracle VM VirtualBox can provide hard disk space to a VM, see chapter 5, *Virtual Storage*, page 83, but the most common way is to use a large image file on your "real" hard disk, whose contents Oracle VM VirtualBox presents to your VM as if it were a complete hard disk. This file represents an entire hard disk then, so you can even copy it to another host and use it with another Oracle VM VirtualBox installation.

   The wizard displays the following window:


At this screen, you have the following options:

- To create a new, empty virtual hard disk, click the **Create** button.
- You can pick an *existing* disk image file.

  The drop-down list presented in the window lists all disk images which are currently remembered by Oracle VM VirtualBox. These disk images are currently attached to a virtual machine, or have been attached to a virtual machine.

  Alternatively, click on the small **folder icon** next to the drop-down list. In the displayed file dialog, you can click **Add** to select any disk image file on your host disk.

If you are using Oracle VM VirtualBox for the first time, you will want to create a new disk image. Click the **Create** button.

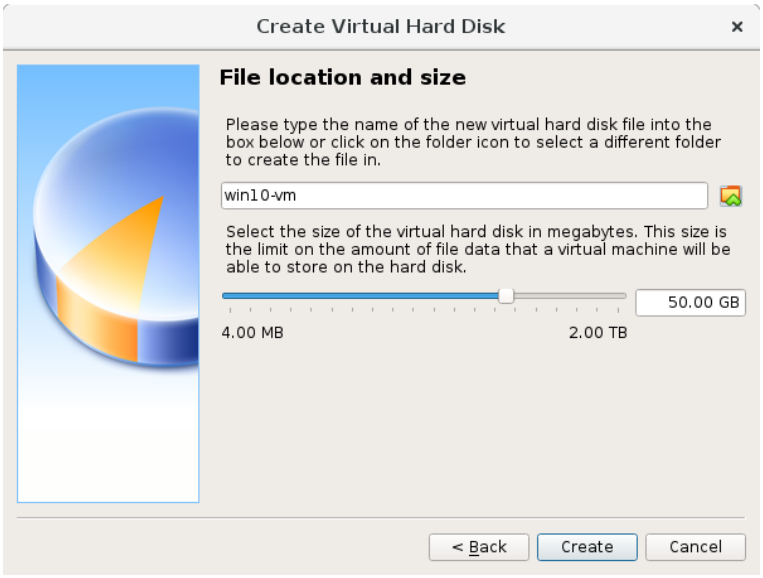
This displays another window, the **Create Virtual Hard Disk Wizard** wizard. This wizard helps you to create a new disk image file in the new virtual machine's folder.

Oracle VM VirtualBox supports the following types of image files:

- A **dynamically allocated file** will only grow in size when the guest actually stores data on its virtual hard disk. It will therefore initially be small on the host hard drive and only later grow to the size specified as it is filled with data.
- A **fixed-size file** will immediately occupy the file specified, even if only a fraction of the virtual hard disk space is actually in use. While occupying much more space, a fixed-size file incurs less overhead and is therefore slightly faster than a dynamically allocated file.

For details about the differences, see chapter 5.2, *Disk Image Files (VDI, VMDK, VHD, HDD)*, page 86.

To prevent your physical hard disk from running full, Oracle VM VirtualBox limits the size of the image file. Still, it needs to be large enough to hold the contents of your OS and the applications you want to install. For a modern Windows or Linux guest, you will probably need several gigabytes for any serious use. The limit of the image file size can be changed later, see chapter 8.24, *VBoxManage modifymedium*, page 167.




After having selected or created your image file, click **Next** to go to the next page.

6. Click **Create**, to create your new virtual machine. The virtual machine is displayed in the list on the left side of the VirtualBox Manager window, with the name that you entered initially.

**Note:** After becoming familiar with the use of wizards, consider using the Expert Mode available in some wizards. Where available, this is selectable using a button, and speeds up the process of using wizards.

## 1.9 Running Your Virtual Machine

To start a virtual machine, you have several options:

- Double-click on the VM's entry in the list in the VirtualBox Manager window.
- Select the VM's entry in the list in the VirtualBox Manager window, and click **Start** at the top of the window.
- Go to the `VirtualBox VMs` folder in your system user's home directory. Find the subdirectory of the machine you want to start and double-click on the machine settings file. This file has a `.vbox` file extension.

Starting a virtual machine displays a new window, and the virtual machine which you selected will boot up. Everything which would normally be seen on the virtual system's monitor is shown in the window. See the screenshot image in chapter 1, *First Steps*, page 1.

In general, you can use the virtual machine as you would use a real computer. There are couple of points worth mentioning however.

### 1.9.1 Starting a New VM for the First Time

When a VM is started for the first time, the **First Start Wizard**, is displayed. This wizard helps you to select an installation medium. Since the VM is created empty, it would otherwise behave just like a real computer with no OS installed. It will do nothing and display an error message that no bootable OS was found.

For this reason, the wizard helps you to select a medium to install an OS from.

- If you have physical CD or DVD media from which you want to install your guest OS, such as a Windows installation CD or DVD, put the media into your host's CD or DVD drive.

  In the wizard's drop-down list of installation media, select **Host Drive** with the correct drive letter. In the case of a Linux host, choose a device file. This will allow your VM to access the media in your host drive, and you can proceed to install from there.

- If you have downloaded installation media from the Internet in the form of an ISO image file such as with a Linux distribution, you would normally burn this file to an empty CD or DVD and proceed as described above. With Oracle VM VirtualBox however, you can skip this step and mount the ISO file directly. Oracle VM VirtualBox will then present this file as a CD or DVD-ROM drive to the virtual machine, much like it does with virtual hard disk images.

  In this case, the wizard's drop-down list contains a list of installation media that were previously used with Oracle VM VirtualBox.

  If your medium is not in the list, especially if you are using Oracle VM VirtualBox for the first time, click the small folder icon next to the drop-down list to display a standard file dialog. Here you can pick an image file on your host disks.

After completing the choices in the wizard, you will be able to install your OS.

### 1.9.2 Capturing and Releasing Keyboard and Mouse

Oracle VM VirtualBox provides a virtual USB tablet device to new virtual machines through which mouse events are communicated to the guest OS. If you are running a modern guest OS that can handle such devices, mouse support may work out of the box without the mouse being *captured* as described below. See chapter 3.5.1, *Motherboard Tab*, page 47.

Otherwise, if the virtual machine only sees standard PS/2 mouse and keyboard devices, since the OS in the virtual machine does not know that it is not running on a real computer, it expects to have exclusive control over your keyboard and mouse. But unless you are running the VM in full screen mode, your VM needs to share keyboard and mouse with other applications and possibly other VMs on your host.

After installing a guest OS and before you install the Guest Additions, described later, either your VM or the rest of your computer can "own" the keyboard and the mouse. Both cannot own the keyboard and mouse at the same time. You will see a *second* mouse pointer which is always confined to the limits of the VM window. You activate the VM by clicking inside it.

To return ownership of keyboard and mouse to your host OS, Oracle VM VirtualBox reserves a special key on your keyboard: the *Host key*. By default, this is the *right Ctrl key* on your keyboard. On a Mac host, the default Host key is the left Command key. You can change this default in the Oracle VM VirtualBox Global Settings. See chapter 1.16, *Global Settings*, page 27. The current setting for the Host key is always displayed at the bottom right of your VM window.

This means the following:

- Your **keyboard** is owned by the VM if the VM window on your host desktop has the keyboard focus. If you have many windows open in your guest OS, the window that has the focus in your VM is used. This means that if you want to enter text within your VM, click on the title bar of your VM window first.

  To release keyboard ownership, press the Host key. As explained above, this is typically the right Ctrl key.

  Note that while the VM owns the keyboard, some key sequences, such as Alt-Tab, will no longer be seen by the host, but will go to the guest instead. After you press the Host key to reenable the host keyboard, all key presses will go through the host again, so that sequences such as Alt-Tab will no longer reach the guest. For technical reasons it may not be possible for the VM to get all keyboard input even when it does own the keyboard. Examples of this are the Ctrl-Alt-Del sequence on Windows hosts or single keys grabbed by other applications on X11 hosts like the GNOME desktop's "Control key highlights mouse pointer" functionality.

- Your **mouse** is owned by the VM only after you have clicked in the VM window. The host mouse pointer will disappear, and your mouse will drive the guest's pointer instead of your normal mouse pointer.

  Note that mouse ownership is independent of that of the keyboard. Even after you have clicked on a titlebar to be able to enter text into the VM window, your mouse is not necessarily owned by the VM yet.

  To release ownership of your mouse by the VM, press the Host key.

As this behavior can be inconvenient, Oracle VM VirtualBox provides a set of tools and device drivers for guest systems called the Oracle VM VirtualBox Guest Additions which make VM keyboard and mouse operation a lot more seamless. Most importantly, the Additions will get rid of the second "guest" mouse pointer and make your host mouse pointer work directly in the guest. See chapter 4, *Guest Additions*, page 62.

### 1.9.3 Typing Special Characters

OSes expect certain key combinations to initiate certain procedures. Some of these key combinations may be difficult to enter into a virtual machine, as there are three candidates as to who receives keyboard input: the host OS, Oracle VM VirtualBox, or the guest OS. Which of these three receives keypresses depends on a number of factors, including the key itself.

- Host OSes reserve certain key combinations for themselves. For example, it is impossible to enter the **Ctrl+Alt+Delete** combination if you want to reboot the guest OS in your virtual machine, because this key combination is usually hard-wired into the host OS, both Windows and Linux intercept this, and pressing this key combination will therefore reboot your *host*.

  On Linux and Oracle Solaris hosts, which use the X Window System, the key combination **Ctrl+Alt+Backspace** normally resets the X server and restarts the entire graphical user interface. As the X server intercepts this combination, pressing it will usually restart your *host* graphical user interface and kill all running programs, including Oracle VM VirtualBox, in the process.

  On Linux hosts supporting virtual terminals, the key combination **Ctrl+Alt+Fx**, where Fx is one of the function keys from F1 to F12, normally enables you to switch between virtual terminals. As with Ctrl+Alt+Delete, these combinations are intercepted by the host OS and therefore always switch terminals on the *host*.

  If, instead, you want to send these key combinations to the *guest* OS in the virtual machine, you will need to use one of the following methods:

  - Use the items in the **Input**, **Keyboard** menu of the virtual machine window. This menu includes the settings **Insert Ctrl+Alt+Delete** and **Ctrl+Alt+Backspace**. The latter will only have an effect with Linux or Oracle Solaris guests, however.

    This menu also includes an option for inserting the Host key combination.
  - Use special key combinations with the Host key, normally the right Control key. Oracle VM VirtualBox will then translate these key combinations for the virtual machine:
    * **Host key + Del** to send Ctrl+Alt+Del to reboot the guest.
    * **Host key + Backspace** to send Ctrl+Alt+Backspace to restart the graphical user interface of a Linux or Oracle Solaris guest.
    * **Host key + Function key**. For example, to simulate Ctrl+Alt+Fx to switch between virtual terminals in a Linux guest.

- For some other keyboard combinations such as **Alt-Tab** to switch between open windows, Oracle VM VirtualBox enables you to configure whether these combinations will affect the host or the guest, if a virtual machine currently has the focus. This is a global setting for all virtual machines and can be found under **File**, **Preferences**, **Input**.

### 1.9.4 Changing Removable Media

While a virtual machine is running, you can change removable media in the **Devices** menu of the VM's window. Here you can select in detail what Oracle VM VirtualBox presents to your VM as a CD, DVD, or floppy drive.

The settings are the same as those available for the VM in the **Settings** dialog of the Oracle VM VirtualBox main window. But as the **Settings** dialog is disabled while the VM is in the Running or Saved state, the **Devices** menu saves you from having to shut down and restart the VM every time you want to change media.

Using the **Devices** menu, you can attach the host drive to the guest or select a floppy or DVD image, as described in chapter 3.7, *Storage Settings*, page 51.

The **Devices** menu also includes an option for creating a virtual ISO (VISO) from selected files on the host.

### 1.9.5 Resizing the Machine's Window

You can resize the virtual machine's window when it is running. In that case, one of the following things will happen:

1. If you have **scaled mode** enabled, then the virtual machine's screen will be scaled to the size of the window. This can be useful if you have many machines running and want to have a look at one of them while it is running in the background. Alternatively, it might be useful to enlarge a window if the VM's output screen is very small, for example because you are running an old OS in it.

   To enable scaled mode, press **Host key + C**, or select **Scaled Mode** from the **View** menu in the VM window. To leave scaled mode, press **Host key + C** again.

   The aspect ratio of the guest screen is preserved when resizing the window. To ignore the aspect ratio, press **Shift** during the resize operation.

   See chapter 14, *Known Limitations*, page 301 for additional remarks.

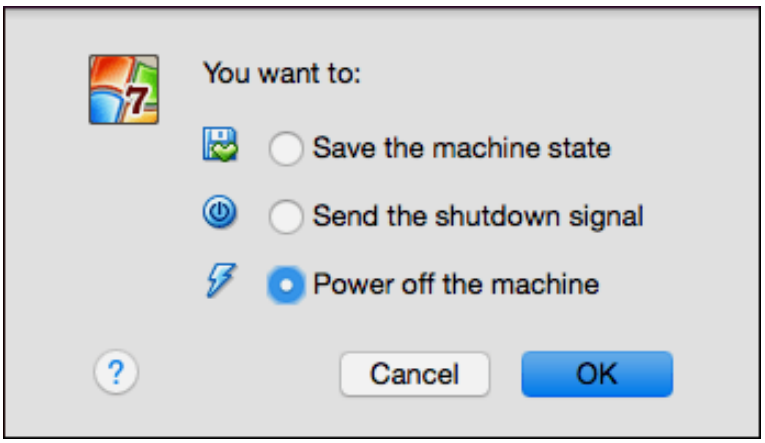
2. If you have the Guest Additions installed and they support automatic **resizing**, the Guest Additions will automatically adjust the screen resolution of the guest OS. For example, if you are running a Windows guest with a resolution of 1024x768 pixels and you then resize the VM window to make it 100 pixels wider, the Guest Additions will change the Windows display resolution to 1124x768.

See chapter 4, *Guest Additions*, page 62.

3. Otherwise, if the window is bigger than the VM's screen, the screen will be centered. If it is smaller, then scroll bars will be added to the machine window.

### 1.9.6 Saving the State of the Machine

When you click on the **Close** button of your virtual machine window, at the top right of the window, just like you would close any other window on your system, Oracle VM VirtualBox asks you whether you want to save or power off the VM. As a shortcut, you can also press **Host key** + **Q**.




The difference between the three options is crucial. They mean the following:

- **Save the machine state:** With this option, Oracle VM VirtualBox *freezes* the virtual machine by completely saving its state to your local disk.

  When you start the VM again later, you will find that the VM continues exactly where it was left off. All your programs will still be open, and your computer resumes operation. Saving the state of a virtual machine is thus in some ways similar to suspending a laptop computer by closing its lid.

- **Send the shutdown signal.** This will send an ACPI shutdown signal to the virtual machine, which has the same effect as if you had pressed the power button on a real computer. So long as the VM is running a fairly modern OS, this should trigger a proper shutdown mechanism from within the VM.

- **Power off the machine:** With this option, Oracle VM VirtualBox also stops running the virtual machine, but *without* saving its state.

  **Warning:** This is equivalent to pulling the power plug on a real computer without shutting it down properly. If you start the machine again after powering it off, your OS will have to reboot completely and may begin a lengthy check of its virtual system disks. As a result, this should not normally be done, since it can potentially cause data loss or an inconsistent state of the guest system on disk.

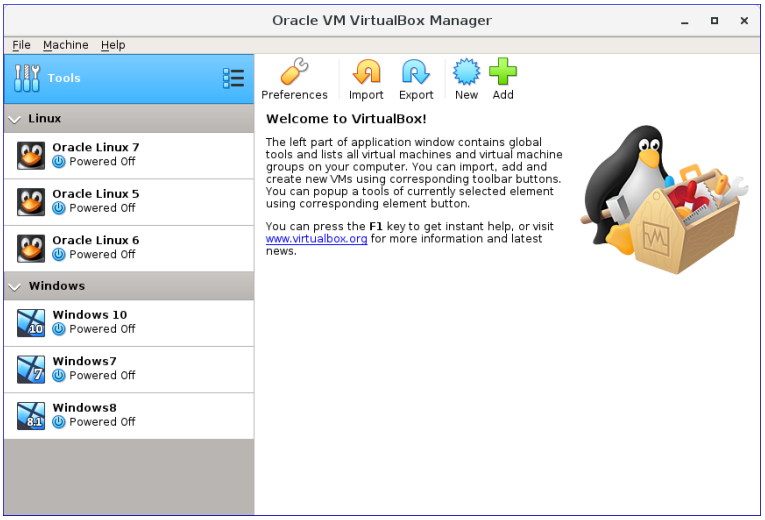
  As an exception, if your virtual machine has any snapshots, see chapter 1.11, *Snapshots*, page 17, you can use this option to quickly **restore the current snapshot** of the virtual machine. In that case, powering off the machine will not disrupt its state, but any changes made since that snapshot was taken will be lost.

The **Discard** button in the VirtualBox Manager window discards a virtual machine's saved state. This has the same effect as powering it off, and the same warnings apply.

## 1.10 Using VM Groups

VM groups enable the user to create ad hoc groups of VMs, and to manage and perform functions on them collectively, as well as individually.

The following figure shows VM groups displayed in VirtualBox Manager.




The following features are available for groups:

- Create a group using the VirtualBox Manager. Do one of the following:
  - Drag one VM on top of another VM.
  - Select multiple VMs and select **Group** from the right-click menu.
- Create and manage a group using the command line. Do one of the following:
  - Create a group and assign a VM. For example:

    ```
    VBoxManage modifyvm "vm01" --groups "/TestGroup"
    ```

    This command creates a group “TestGroup” and attaches the VM “vm01” to that group.
  - Detach a VM from the group, and delete the group if empty. For example:

    ```
    VBoxManage modifyvm "vm01" --groups ""
    ```

    This command detaches all groups from the VM “vm01” and deletes the empty group.
- Create multiple groups. For example:

  ```
  VBoxManage modifyvm "vm01" --groups "/TestGroup,/TestGroup2"
  ```

  This command creates the groups “TestGroup” and “TestGroup2”, if they do not exist, and attaches the VM “vm01” to both of them.
- Create nested groups, having a group hierarchy. For example:

  ```
  VBoxManage modifyvm "vm01" --groups "/TestGroup/TestGroup2"
  ```

  This command attaches the VM “vm01” to the subgroup “TestGroup2” of the “TestGroup” group.
- The following is a summary of group commands: Start, Pause, Reset, Close (save state, send shutdown signal, poweroff), Discard Saved State, Show in File System, Sort.

## 1.11 Snapshots

With snapshots, you can save a particular state of a virtual machine for later use. At any later time, you can revert to that state, even though you may have changed the VM considerably since then. A snapshot of a virtual machine is thus similar to a machine in Saved state, but there can be many of them, and these saved states are preserved.

To see the snapshots of a virtual machine, click on the machine name in VirtualBox Manager. Then click the **List** icon next to the machine name, and select **Snapshots**. Until you take a snapshot of the machine, the list of snapshots will be empty except for the **Current State** item, which represents the "now" point in the lifetime of the virtual machine.

### 1.11.1 Taking, Restoring, and Deleting Snapshots

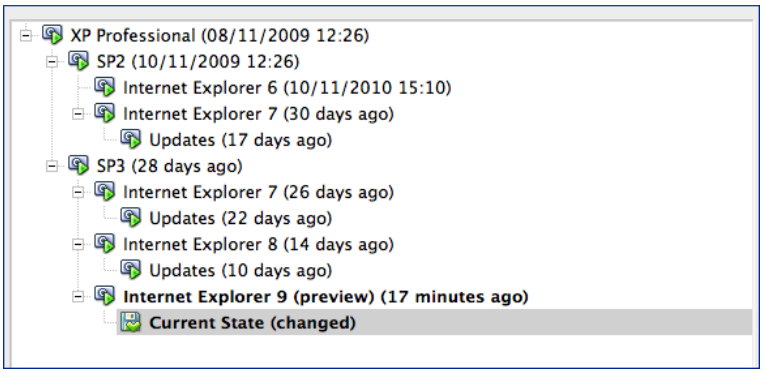
There are three operations related to snapshots, as follows:

1. **Take a snapshot**. This makes a copy of the machine's current state, to which you can go back at any given time later.

   - If your VM is running, select **Take Snapshot** from the **Machine** pull-down menu of the VM window.
   - If your VM is in either the Saved or the Powered Off state, as displayed next to the VM name in the Oracle VM VirtualBox main window, click the **List** icon next to the machine name and select **Snapshots**. The snapshots window is shown. Do one of the following:
     - Click the **Take** icon.
     - Right-click on the **Current State** item in the list and select **Take**.

   In either case, a window is displayed prompting you for a snapshot name. This name is purely for reference purposes to help you remember the state of the snapshot. For example, a useful name would be "Fresh installation from scratch, no Guest Additions", or "Service Pack 3 just installed". You can also add a longer text in the **Description** field.

   Your new snapshot will then appear in the snapshots list. Underneath your new snapshot, you will see an item called **Current State**, signifying that the current state of your VM is a variation based on the snapshot you took earlier. If you later take another snapshot, you will see that they are displayed in sequence, and that each subsequent snapshot is derived from an earlier one.




   Oracle VM VirtualBox imposes no limits on the number of snapshots you can take. The only practical limitation is disk space on your host. Each snapshot stores the state of the virtual machine and thus occupies some disk space. See chapter 1.11.2, *Snapshot Contents*, page 18 for details on what is stored in a snapshot.

2. **Restore a snapshot**. In the list of snapshots, right-click on any snapshot you have taken and select **Restore**. By restoring a snapshot, you go back or forward in time. The current state of the machine is lost, and the machine is restored to the exact state it was in when the snapshot was taken.

   **Note:** Restoring a snapshot will affect the virtual hard drives that are connected to your VM, as the entire state of the virtual hard drive will be reverted as well. This means also that all files that have been created since the snapshot and all other file changes *will be lost*. In order to prevent such data loss while still making use of the snapshot feature, it is possible to add a second hard drive in *write-through* mode using the `VBoxManage` interface and use it to store your data. As write-through hard drives are *not* included in snapshots, they remain unaltered when a machine is reverted. See chapter 5.4, *Special Image Write Modes*, page 89.

   To avoid losing the current state when restoring a snapshot, you can create a new snapshot before the restore operation.

   By restoring an earlier snapshot and taking more snapshots from there, it is even possible to create a kind of alternate reality and to switch between these different histories of the virtual machine. This can result in a whole tree of virtual machine snapshots, as shown in the screenshot above.

3. **Delete a snapshot**. This does not affect the state of the virtual machine, but only releases the files on disk that Oracle VM VirtualBox used to store the snapshot data, thus freeing disk space. To delete a snapshot, right-click on the snapshot name in the snapshots tree and select **Delete**. Snapshots can be deleted even while a machine is running.

   **Note:** Whereas taking and restoring snapshots are fairly quick operations, deleting a snapshot can take a considerable amount of time since large amounts of data may need to be copied between several disk image files. Temporary disk files may also need large amounts of disk space while the operation is in progress.

   There are some situations which cannot be handled while a VM is running, and you will get an appropriate message that you need to perform this snapshot deletion when the VM is shut down.

### 1.11.2 Snapshot Contents

Think of a snapshot as a point in time that you have preserved. More formally, a snapshot consists of the following:

- The snapshot contains a complete copy of the VM settings, including the hardware configuration, so that when you restore a snapshot, the VM settings are restored as well. For example, if you changed the hard disk configuration or the VM's system settings, that change is undone when you restore the snapshot.

  The copy of the settings is stored in the machine configuration, an XML text file, and thus occupies very little space.

- The complete state of all the virtual disks attached to the machine is preserved. Going back to a snapshot means that all changes that had been made to the machine's disks, file by file and bit by bit, will be undone as well. Files that were since created will disappear, files that were deleted will be restored, changes to files will be reverted.

Strictly speaking, this is only true for virtual hard disks in "normal" mode. You can configure disks to behave differently with snapshots, see chapter 5.4, *Special Image Write Modes*, page 89. In technical terms, it is not the virtual disk itself that is restored when a snapshot is restored. Instead, when a snapshot is taken, Oracle VM VirtualBox creates differencing images which contain only the changes since the snapshot were taken. When the snapshot is restored, Oracle VM VirtualBox throws away that differencing image, thus going back to the previous state. This is both faster and uses less disk space. For the details, which can be complex, see chapter 5.5, *Differencing Images*, page 90.

Creating the differencing image as such does not occupy much space on the host disk initially, since the differencing image will initially be empty and grow dynamically later with each write operation to the disk. The longer you use the machine after having created the snapshot, however, the more the differencing image will grow in size.

- If you took a snapshot while the machine was running, the memory state of the machine is also saved in the snapshot. This is in the same way that memory can be saved when you close a VM window. When you restore such a snapshot, execution resumes at exactly the point when the snapshot was taken.

  The memory state file can be as large as the memory size of the virtual machine and will therefore occupy quite some disk space as well.

## 1.12 Virtual Machine Configuration

When you select a virtual machine from the list in the VirtualBox Manager window, you will see a summary of that machine's settings on the right.

Clicking on **Settings** displays a window, where you can configure many of the properties of the selected VM. But be careful when changing VM settings. It is possible to change all VM settings after installing a guest OS, but certain changes might prevent a guest OS from functioning correctly if done after installation.

> **Note:** The **Settings** button is disabled while a VM is either in the Running or Saved state. This is because the **Settings** dialog enables you to change fundamental characteristics of the virtual machine that is created for your guest OS. For example, the guest OS may not perform well if half of its memory is taken away. As a result, if the **Settings** button is disabled, shut down the current VM first.

Oracle VM VirtualBox provides a wide range of parameters that can be changed for a virtual machine. The various settings that can be changed in the **Settings** window are described in detail in chapter 3, *Configuring Virtual Machines*, page 40. Even more parameters are available when using the `VBoxManage` command line interface. See chapter 8, *VBoxManage*, page 120.

## 1.13 Removing and Moving Virtual Machines

You can remove a VM from Oracle VM VirtualBox or move the VM and its associated files, such as disk images, to another location on the host.

- **Removing a VM.** To remove a VM, right-click on the VM in the VirtualBox Manager's machine list and select **Remove**.

  The confirmation dialog enables you to specify whether to only remove the VM from the list of machines or to remove the files associated with the VM.

  Note that the **Remove** menu item is disabled while a VM is running.

- **Moving a VM.** To move a VM to a new location on the host, right-click on the VM in the VirtualBox Manager's machine list and select **Move**.

  The file dialog prompts you to specify a new location for the VM.

  When you move a VM, Oracle VM VirtualBox configuration files are updated automatically to use the new location on the host.

  Note that the **Move** menu item is disabled while a VM is running.

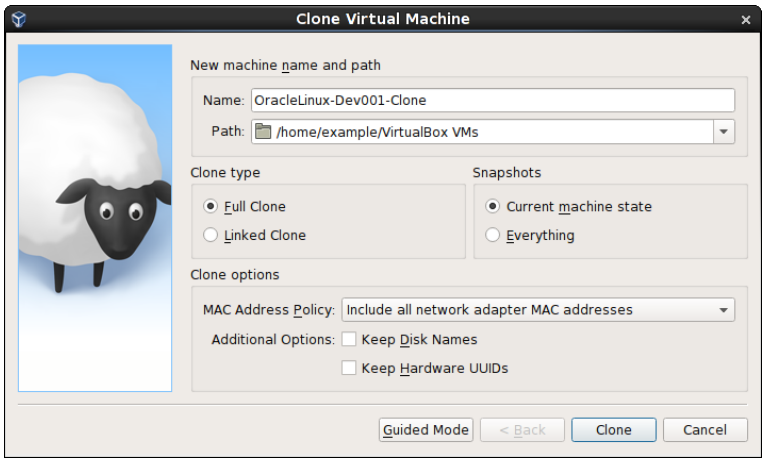
  You can also use the `VBoxManage movevm` command to move a VM. See chapter 8.10, *VBoxManage movevm*, page 148.

For information about removing or moving a disk image file from Oracle VM VirtualBox, see chapter 5.3, *The Virtual Media Manager*, page 87.

## 1.14 Cloning Virtual Machines

You can create a full copy or a linked copy of an existing VM. This copy is called a *clone*. You might use a cloned VM to experiment with a VM configuration, to test different guest OS levels, or to back up a VM.

The **Clone Virtual Machine** wizard guides you through the cloning process.




Start the wizard by clicking **Clone** in the right-click menu of the VirtualBox Manager's machine list or in the **Snapshots** view of the selected VM.

Specify a new **Name** for the clone. You can choose a **Path** for the cloned virtual machine, otherwise Oracle VM VirtualBox uses the default machines folder.

The **Clone Type** option specifies whether to create a clone linked to the source VM or to create a fully independent clone:

- **Full Clone:** Copies all dependent disk images to the new VM folder. A full clone can operate fully without the source VM.
- **Linked Clone:** Creates new differencing disk images based on the source VM disk images. If you select the current state of the source VM as the clone point, Oracle VM VirtualBox creates a new snapshot.

The **Snapshots** option specifies whether to create a clone of the current machine state only or of everything.

- **Everything:** Clones the current machine state and all its snapshots.

- **Current Machine State and All Children:**. Clones a VM snapshot and all its child snapshots.

The following clone options are available:

- **MAC Address Policy:** Specifies how to retain network card MAC addresses when cloning the VM.

  For example, the **Generate New MAC Addresses For All Network Adapters** value assigns a new MAC address to each network card during cloning. This is the default setting. This is the best option when both the source VM and the cloned VM must operate on the same network. Other values enable you to retain the existing MAC addresses in the cloned VM.

- **Keep Disk Names:** Retains the disk image names when cloning the VM.

- **Keep Hardware UUIDs:** Retains the hardware universally unique identifiers (UUIDs) when cloning the VM.

The duration of the clone operation depends on the size and number of attached disk images. In addition, the clone operation saves all the differencing disk images of a snapshot.

Note that the **Clone** menu item is disabled while a machine is running.

You can also use the `VBoxManage clonevm` command to clone a VM. See chapter 8.9, *VBoxManage clonevm*, page 147.

## 1.15 Importing and Exporting Virtual Machines

Oracle VM VirtualBox can import and export virtual machines in the following formats:

- **Open Virtualization Format (OVF).** This is the industry-standard format. See chapter 1.15.1, *About the OVF Format*, page 21.

- **Cloud service formats.** Export to cloud services such as Oracle Cloud Infrastructure is supported. Import is not supported. See chapter 1.15.4, *Exporting an Appliance to Oracle Cloud Infrastructure*, page 24.

### 1.15.1 About the OVF Format

OVF is a cross-platform standard supported by many virtualization products which enables the creation of ready-made virtual machines that can then be imported into a hypervisor such as Oracle VM VirtualBox. Oracle VM VirtualBox makes OVF import and export easy to do, using the VirtualBox Manager window or the command-line interface.

Using OVF enables packaging of *virtual appliances*. These are disk images, together with configuration settings that can be distributed easily. This way one can offer complete ready-to-use software packages, including OSes with applications, that need no configuration or installation except for importing into Oracle VM VirtualBox.

> **Note:** The OVF standard is complex, and support in Oracle VM VirtualBox is an ongoing process. In particular, no guarantee is made that Oracle VM VirtualBox supports all appliances created by other virtualization software. For a list of known limitations, see chapter 14, *Known Limitations*, page 301.

Appliances in OVF format can appear in the following variants:

- They can come in several files, as one or several disk images, typically in the widely-used VMDK format. See chapter 5.2, *Disk Image Files (VDI, VMDK, VHD, HDD)*, page 86. They also include a textual description file in an XML dialect with an `.ovf` extension. These files must then reside in the same directory for Oracle VM VirtualBox to be able to import them.
- Alternatively, the above files can be packed together into a single archive file, typically with an `.ova` extension. Such archive files use a variant of the TAR archive format and can therefore be unpacked outside of Oracle VM VirtualBox with any utility that can unpack standard TAR files.

**Note:** OVF cannot describe snapshots that were taken for a virtual machine. As a result, when you export a virtual machine that has snapshots, only the current state of the machine will be exported. The disk images in the export will have a *flattened* state identical to the current state of the virtual machine.

### 1.15.2 Importing an Appliance in OVF Format

The following steps show how to import an appliance in OVF format.

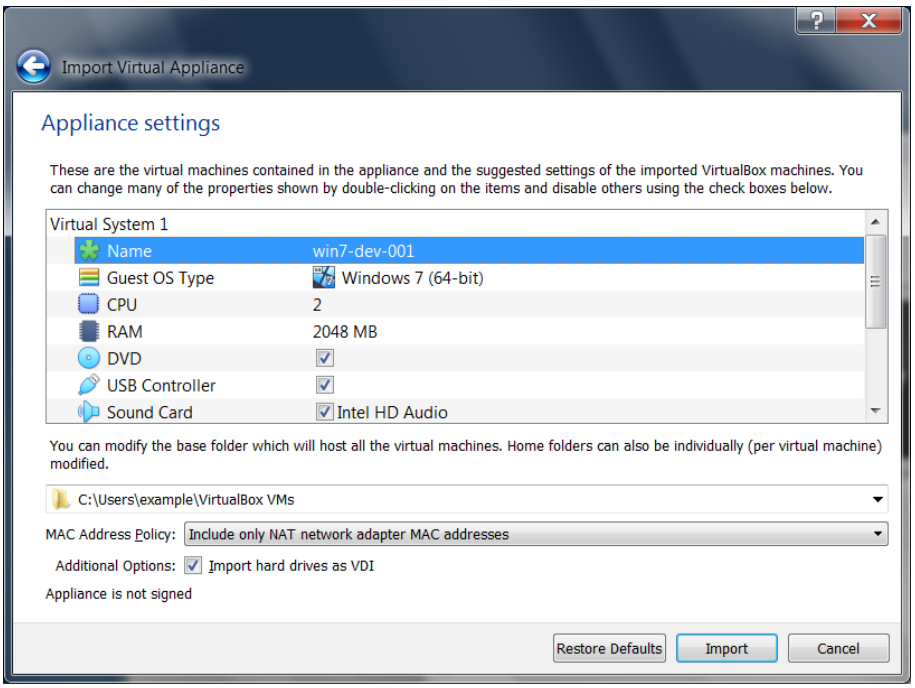
1. Double-click on the OVF or OVA file.

   Oracle VM VirtualBox creates file type associations automatically for any OVF and OVA files on your host OS.

2. Select **File**, **Import Appliance** from the VirtualBox Manager window.

   From the file dialog, go to the file with either the `.ovf` or the `.ova` file extension.

   Click **Import** to open the **Appliance Settings** screen.



This screen shows the VMs described in the OVF or OVA file and enables you to change the VM settings.

By default, membership of VM groups is preserved on import for VMs that were initially exported from Oracle VM VirtualBox. You can change this behavior by using the **Primary Group** setting for the VM.

The following global settings apply to all of the VMs that you import:

- **Base Folder:** Specifies the directory on the host in which to store the imported VMs.

  If an appliance has multiple VMs, you can specify a different directory for each VM by editing the **Base Folder** setting for the VM.
- **MAC Address Policy:** Reinitializes the MAC addresses of network cards in your VMs prior to import, by default. You can override the default behavior and preserve the MAC addresses on import.
- **Import Hard Drives as VDI:** Imports hard drives in the VDI format rather that in the defalut VMDK format.

3. Click **Import** to import the appliance.

   Oracle VM VirtualBox copies the disk images and creates local VMs with the settings described on the **Appliance Settings** screen. The imported VMs are shown in the list of VMs in VirtualBox Manager.

   Because disk images are large, the VMDK images that are included with virtual appliances are shipped in a compressed format that cannot be used directly by VMs. So, the images are first unpacked and copied, which might take several minutes.

You can use the `VBoxManage import` command to import an appliance. See chapter 8.11, *VBoxManage import*, page 148.

### 1.15.3 Exporting an Appliance in OVF Format

The following steps show how to export an appliance in OVF format.

1. Select **File**, **Export Appliance** to open the **Export Virtual Appliance** wizard.

   From the initial window, you can combine several VMs into an OVF appliance.

   Select one or more VMs to export, and click **Next**.

2. The **Appliance Settings** screen enables you to select the following settings:

   - **Format:** Selects the **Open Virtualization Format** value for the output files.

     The **Oracle Cloud Infrastructure** value exports export to Oracle Cloud Infrastructure. See chapter 1.15.4, *Exporting an Appliance to Oracle Cloud Infrastructure*, page 24.
   - **File:** Selects the location in which to store the exported files.
   - **MAC Address Policy:** Specifies whether to retain or reassign network card MAC addresses on export.
   - **Write Manifest File:** Enables you to include a manifest file in the exported archive file.
   - **Include ISO Image Files:** Enables you to include ISO image files in the exported archive file.

3. Click **Next** to show the **Virtual System Settings** screen.

   You can edit settings for the virtual appliance. For example, you can change the name of the virtual appliance or add product information, such as vendor details or license text.

   Double-click the appropriate field to change its value.

4. Click **Export** to begin the export process. Note that this operation might take several minutes.

You can use the `VBoxManage export` command to export an appliance. See chapter 8.12, *VBoxManage export*, page 150.

### 1.15.4 Exporting an Appliance to Oracle Cloud Infrastructure

Oracle VM VirtualBox supports the export of VMs to an Oracle Cloud Infrastructure service.

Before you can export a VM to Oracle Cloud Infrastructure, ensure that you perform the following configuration steps:

- Generate an API signing key pair that is used for API requests to Oracle Cloud Infrastructure.
  - The key pair is usually installed in the `.oci` folder in your home directory. For example, `~/.oci` on a Linux system.
  - Upload the public key of the key pair to the cloud service.

  For step-by-step instructions for creating and uploading an API signing key for Oracle Cloud Infrastructure, see:

  `https://docs.cloud.oracle.com/iaas/Content/API/Concepts/apisigningkey.htm#How`

- Create a profile for your cloud account.

  The cloud profile contains resource identifiers for your cloud account, such as your user OCID, and the fingerprint for your public key. You can create a cloud profile in the following ways:

  - Automatically by using the **Cloud Profile Manager**. See chapter 1.15.5, *The Cloud Profile Manager*, page 25.
  - Manually by creating an `oci_config` file in your Oracle VM VirtualBox global configuration directory. For example, this is `$HOME/.config/VirtualBox/oci_config` on a Linux host.
  - Manually by creating a `config` file in your Oracle Cloud Infrastructure configuration directory. For example, this is `$HOME/.oci/config` on a Linux host.

    This is the same file that is used by the Oracle Cloud Infrastructure command line interface.

    Oracle VM VirtualBox automatically uses the `config` file if no cloud profile file is present in your global configuration directory. Alternatively, you can import this file manually into the Cloud Profile Manager.

  For more information about the cloud profile settings used by Oracle Cloud Infrastructure see:

  `https://docs.cloud.oracle.com/iaas/Content/API/Concepts/sdkconfig.htm`

- Ensure that the subnets that are used by source VMs are available in the target compartment on the cloud service.

Perform the following steps to export a VM to Oracle Cloud Infrastructure:

1. Select **File**, **Export Appliance** to open the **Export Virtual Appliance** wizard.

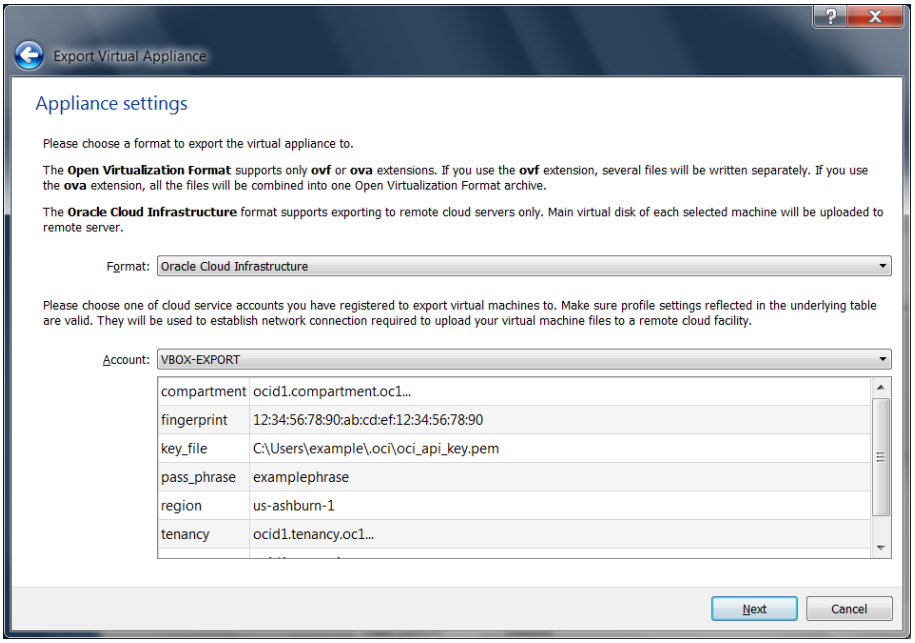
   Select a VM to export and click **Next** to open the **Appliance Settings** screen.

2. From the **Format** drop-down list, select **Oracle Cloud Infrastructure**.

   In the **Account** drop-down list, select your Oracle Cloud Infrastructure account.

   You can set up Oracle Cloud Infrastructure accounts by using the Cloud Profile Manager.

   The list after the **Account** field shows the profile settings for your cloud account.




   Click **Next** to make an API request to the Oracle Cloud Infrastructure service and open the **Virtual System Settings** screen.

3. Optionally edit settings used for the virtual machine on Oracle Cloud Infrastructure.

   For example, you can edit the Disk Size and Shape used for the VM instance.

   Click **Export** to export the VMs to the cloud service.

   The VMs are uploaded to Oracle Cloud Infrastructure.

   Instances are created for the uploaded VMs.

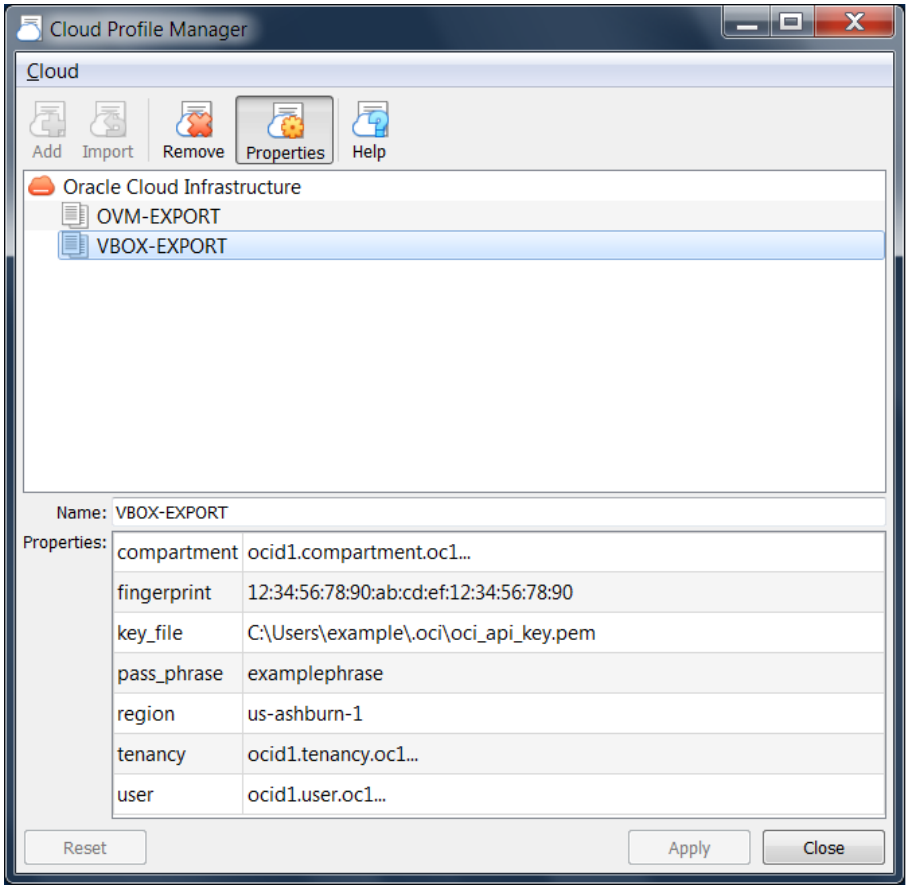
   By default, the VM instance is started after upload to Oracle Cloud Infrastructure.

4. Monitor the export process by using the Oracle Cloud Infrastructure Console.

You can also use the `VBoxManage export` command to export a VM to Oracle Cloud Infrastructure. See chapter 8.12.2, *Export to Oracle Cloud Infrastructure*, page 150.

### 1.15.5 The Cloud Profile Manager

The Cloud Profile Manager is a component of Oracle VM VirtualBox that enables you to create, edit, and manage cloud profiles for your cloud service accounts.




To open the Cloud Profile Manager select **File**, **Cloud Profile Manager** from the VirtualBox Manager window.

Use the Cloud Profile Manager to create a new cloud profile automatically. Or, create a cloud profile by importing settings from your Oracle Cloud Infrastructure configuration file into the Cloud Profile Manager.

Perform the following steps to create a new cloud profile:

1. Click the **Add** icon and specify a **Name** for the profile.
2. Click **Properties** and specify the following property values for the profile:
   - Compartment OCID
   - Fingerprint of the public key
   - Location of the private key on the client device
   - (Optional) Passphrase for the private key, if the key is encrypted
   - Region OCID
   - Tenancy OCID
   - User OCID

   Some of these are settings for your Oracle Cloud Infrastructure account, which you can view from the Oracle Cloud Infrastructure Console.
3. Click **Apply** to save your changes.

   The cloud profile settings are saved in the `oci_config` file in your Oracle VM VirtualBox global settings directory.

Perform the following steps to import an existing Oracle Cloud Infrastructure configuration file:

1. Ensure that a `config` file is present in your Oracle Cloud Infrastructure configuration directory. For example, this is `$HOME/.oci/config` on a Linux host.
2. Click the **Import** icon to open a dialog that prompts you to import cloud profiles from external files.

   **Warning:** This action overwrites any cloud profiles that are in your Oracle VM VirtualBox global settings directory.

3. Click **Import**.

   Your cloud profile settings are saved to the `oci_config` file in your Oracle VM VirtualBox global settings directory.
4. Click **Properties** to show the cloud profile settings.

   Double-click on the appropriate field to change the value.
5. Click **Apply** to save your changes.

## 1.16 Global Settings

The **Global Settings** dialog can be displayed using the **File** menu, by clicking the **Preferences** item. This dialog offers a selection of settings, most of which apply to all virtual machines of the current user. The **Extensions** option applies to the entire system.

The following settings are available:

- **General.** Enables the user to specify the default folder/directory for VM files, and the VRDP Authentication Library.
- **Input.** Enables the user to specify the Host key. It identifies the key that toggles whether the cursor is in the focus of the VM or the Host OS windows, see chapter 1.9.2, *Capturing and Releasing Keyboard and Mouse*, page 12, and which is also used to trigger certain VM actions, see chapter 1.9.3, *Typing Special Characters*, page 13.
- **Language.** Enables the user to specify the GUI language.
- **Display.** Enables the user to specify the screen resolution, and its width and height. A default scale factor can be specified for all guest screens.
- **Network.** Enables the user to configure the details of Host Only Networks.
- **Extensions.** Enables the user to list and manage the installed extension packages.
- **Proxy.** Enables the user to configure a HTTP Proxy Server.

## 1.17 Alternative Front-Ends

As briefly mentioned in chapter 1.3, *Features Overview*, page 3, Oracle VM VirtualBox has a very flexible internal design that enables you to use multiple interfaces to control the same virtual machines. For example, you can start a virtual machine with the VirtualBox Manager window and then stop it from the command line. With Oracle VM VirtualBox's support for the Remote Desktop Protocol (RDP), you can even run virtual machines remotely on a headless server and have all the graphical output redirected over the network.

The following front-ends are shipped in the standard Oracle VM VirtualBox package:

- **VirtualBox.** This is the VirtualBox Manager, a graphical user interface that uses the Qt toolkit. This interface is described throughout this manual. While this is the simplest and easiest front-end to use, some of the more advanced Oracle VM VirtualBox features are not included.
- **VBoxManage.** A command-line interface for automated and detailed control of every aspect of Oracle VM VirtualBox. See chapter 8, *VBoxManage*, page 120.
- **VBoxHeadless.** A front-end that produces no visible output on the host at all, but can act as a RDP server if the VirtualBox Remote Desktop Extension (VRDE) is installed and enabled for the VM. As opposed to the other graphical interfaces, the headless front-end requires no graphics support. This is useful, for example, if you want to host your virtual machines on a headless Linux server that has no X Window system installed. See chapter 7.1.2, *VBoxHeadless, the Remote Desktop Server*, page 112.

If the above front-ends still do not satisfy your particular needs, it is possible to create yet another front-end to the complex virtualization engine that is the core of Oracle VM VirtualBox, as the Oracle VM VirtualBox core neatly exposes all of its features in a clean API. See chapter 11, *Oracle VM VirtualBox Programming Interfaces*, page 276.

# 2 Installation Details

As installation of Oracle VM VirtualBox varies depending on your host operating system, the following sections provide installation instructions for Windows, Mac OS X, Linux, and Oracle Solaris.

## 2.1 Installing on Windows Hosts

### 2.1.1 Prerequisites

For the various versions of Windows that are supported as host operating systems, please refer to chapter 1.4, *Supported Host Operating Systems*, page 5.

In addition, Windows Installer 1.1 or later must be present on your system. This should be the case if you have all recent Windows updates installed.

### 2.1.2 Performing the Installation

The Oracle VM VirtualBox installation can be started in either of the following ways:

- By double-clicking on the executable file, which contains both 32-bit and 64-bit architectures.
- By entering the following command:

  ```
  VirtualBox-<version>-<revision>-Win.exe -extract
  ```

  This will extract both installers into a temporary directory, along with .MSI files. Run the following command to to perform the installation:

  ```
  msiexec /i VirtualBox-<version>-<revision>-MultiArch_<x86|amd64>.msi
  ```

Using either way displays the installation **Welcome** dialog and enables you to choose where to install Oracle VM VirtualBox, and which components to install. In addition to the Oracle VM VirtualBox application, the following components are available:

- **USB support.** This package contains special drivers for your Windows host that Oracle VM VirtualBox requires to fully support USB devices inside your virtual machines.
- **Networking.** This package contains extra networking drivers for your Windows host that Oracle VM VirtualBox needs to support Bridged Networking. This enables your VM's virtual network cards to be accessed from other machines on your physical network.
- **Python support.** This package contains Python scripting support for the Oracle VM VirtualBox API, see chapter 11, *Oracle VM VirtualBox Programming Interfaces*, page 276. For this to work, an already working Windows Python installation on the system is required.

  See, for example: `http://www.python.org/download/windows/`.

**Note:** Python version at least 2.6 is required. Since Oracle VM VirtualBox 5.1, Python 3 is also supported.

Depending on your Windows configuration, you may see warnings about unsigned drivers, or similar. Click **Continue** for these warnings, as otherwise Oracle VM VirtualBox might not function correctly after installation.

The installer will create a Oracle VM VirtualBox group in the Windows **Start** menu, which enables you to launch the application and access its documentation.

With standard settings, Oracle VM VirtualBox will be installed for all users on the local system. If this is not wanted, you must invoke the installer by first extracting as follows:

```
VirtualBox.exe -extract
```

Then, run either of the following commands on the extracted .MSI files. This will install Oracle VM VirtualBox only for the current user.

```
VirtualBox.exe -msiparams ALLUSERS=2

msiexec /i VirtualBox-<version>-MultiArch_<x86|amd64>.msi ALLUSERS=2
```

If you do not want to install all features of Oracle VM VirtualBox, you can set the optional `ADDLOCAL` parameter to explicitly name the features to be installed. The following features are available:

#### VBoxApplication

Main binaries of Oracle VM VirtualBox.

**Note:** This feature must not be absent, since it contains the minimum set of files to have working Oracle VM VirtualBox installation.

#### VBoxUSB

USB support.

#### VBoxNetwork

All networking support. This includes the VBoxNetworkFlt and VBoxNetworkAdp features.

#### VBoxNetworkFlt

Bridged networking support.

#### VBoxNetworkAdp

Host-only networking support

#### VBoxPython

Python support

**Note:** Python version at least 2.6 is required. Since Oracle VM VirtualBox 5.1, Python 3 is also supported.

For example, to only install USB support along with the main binaries, run either of the following commands:

```
VirtualBox.exe -msiparams ADDLOCAL=VBoxApplication,VBoxUSB
```

```
msiexec /i VirtualBox-<version>-MultiArch_<x86|amd64>.msi ADDLOCAL=VBoxApplication,VBoxUSB
```

The user is able to choose between NDIS5 and NDIS6 host network filter drivers during the installation. This is done using a command line parameter, `NETWORKTYPE`. The NDIS6 driver is default for Windows Vista and later. For older Windows versions, the installer will automatically select the NDIS5 driver and this cannot be changed. For Windows Vista and later the user can force an install of the legacy NDIS5 host network filter driver by using `NETWORKTYPE=NDIS5`. For example, to install the NDIS5 driver on Windows 7 use either of the following commands:

```
VirtualBox.exe -msiparams NETWORKTYPE=NDIS5
```

```
msiexec /i VirtualBox-<version>-MultiArch_<x86|amd64>.msi NETWORKTYPE=NDIS5
```

### 2.1.3 Uninstallation

As Oracle VM VirtualBox uses the standard Microsoft Windows installer, Oracle VM VirtualBox can be safely uninstalled at any time. Click the program entry in the **Add/Remove Programs** list in the Windows Control Panel.

### 2.1.4 Unattended Installation

Unattended installations can be performed using the standard MSI support.

### 2.1.5 Public Properties

Public properties can be specified with the MSI API, to control additional behavior and features of the Windows host installer. Use either of the following commands:

```
VirtualBox.exe -msiparams NAME=VALUE [...]
```

```
msiexec /i VirtualBox-<version>-MultiArch_<x86|amd64>.msi NAME=VALUE [...]
```

The following public properties are available.

- VBOX_INSTALLDESKTOPSHORTCUT

  Specifies whether or not an Oracle VM VirtualBox icon on the desktop should be created.

  Set to `1` to enable, `0` to disable. Default is 1.

- VBOX_INSTALLQUICKLAUNCHSHORTCUT

  Specifies whether or not an Oracle VM VirtualBox icon in the Quick Launch Bar should be created.

  Set to `1` to enable, `0` to disable. Default is 1.

- VBOX_REGISTERFILEEXTENSIONS

  Specifies whether or not the file extensions .vbox, .vbox-extpack, .ovf, .ova, .vdi, .vmdk, .vhd and .vdd should be associated with Oracle VM VirtualBox. Files of these types then will be opened with Oracle VM VirtualBox.

  Set to `1` to enable, `0` to disable. Default is 1.

- VBOX_START

  Specifies whether to start Oracle VM VirtualBox right after successful installation.

  Set to `1` to enable, `0` to disable. Default is 1.

## 2.2 Installing on Mac OS X Hosts

### 2.2.1 Performing the Installation

For Mac OS X hosts, Oracle VM VirtualBox ships in a `dmg` disk image file. Perform the following steps to install on a Mac OS X host:

1. Double-click on the `dmg` file, to mount the contents.
2. A window opens, prompting you to double-click on the `VirtualBox.pkg` installer file displayed in that window.
3. This will start the installer, which enables you to select where to install Oracle VM VirtualBox.

After installation, you can find an Oracle VM VirtualBox icon in the "Applications" folder in the Finder.

### 2.2.2 Uninstallation

To uninstall Oracle VM VirtualBox, open the disk image `dmg` file and double-click on the uninstall icon shown.

### 2.2.3 Unattended Installation

To perform a non-interactive installation of Oracle VM VirtualBox you can use the command line version of the installer application.

Mount the `dmg` disk image file, as described in the installation procedure, or use the following command line:

```
hdiutil attach /path/to/VirtualBox-xyz.dmg
```

Open a terminal session and run the following command:

```
sudo installer -pkg /Volumes/VirtualBox/VirtualBox.pkg -target /Volumes/Macintosh\ HD
```

## 2.3 Installing on Linux Hosts

### 2.3.1 Prerequisites

For the various versions of Linux that are supported as host operating systems, see chapter 1.4, *Supported Host Operating Systems*, page 5.

You will need to install the following packages on your Linux system before starting the installation. Some systems will do this for you automatically when you install Oracle VM VirtualBox.

- Qt 5.3.2 or later. Qt 5.6.2 or later is recommended.
- SDL 1.2.7 or later. This graphics library is typically called `libsdl` or similar.

**Note:** These packages are only required if you want to run the Oracle VM VirtualBox graphical user interfaces. In particular, `VirtualBox`, the graphical VirtualBox Manager, requires both Qt and SDL. If you only want to run `VBoxHeadless`, neither Qt nor SDL are required.

### 2.3.2 The Oracle VM VirtualBox Driver Modules

In order to run other operating systems in virtual machines alongside your main operating system, Oracle VM VirtualBox needs to integrate very tightly into the system. To do this it installs a driver module called `vboxdrv` which does a lot of that work into the system kernel, which is the part of the operating system which controls your processor and physical hardware. Without this kernel module, you can still use the VirtualBox Manager to configure virtual machines, but they will not start. It also installs network drivers called `vboxnetflt` and `vboxnetadp` which enable virtual machines to make more use of your computer's network capabilities and are needed for any virtual machine networking beyond the basic NAT mode.

Since distributing driver modules separately from the kernel is not something which Linux supports well, the install process creates the modules on the system where they will be used. This usually means first installing software packages from the distribution which are needed for the build process. Normally, these will be the GNU compiler (GCC), GNU Make (make) and packages containing header files for your kernel, as well as making sure that all system updates are installed and that the system is running the most up-to-date kernel included in the distribution. *The running kernel and the header files must be updated to matching versions*. The following list includes some instructions for common distributions. For most of them you may want to start by finding the version name of your kernel, using the command `uname -r` in a terminal. The instructions assume that you have not changed too much from the original installation, particularly not installed a different kernel type. If you have, then you will need to determine yourself what to set up.

- With Debian and Ubuntu-based distributions, you must install the correct version of the `linux-headers`, usually whichever of `linux-headers-generic`, `linux-headers-amd64`, `linux-headers-i686` or `linux-headers-i686-pae` best matches the kernel version name. Also, the `linux-kbuild` package if it exists. Basic Ubuntu releases should have the correct packages installed by default.
- On Fedora, Redhat, Oracle Linux and many other RPM-based systems, the kernel version sometimes has a code of letters or a word close to the end of the version name. For example "uek" for the Oracle Enterprise kernel or "default" or "desktop" for the standard SUSE kernels. In this case, the package name is `kernel-uek-devel` or equivalent. If there is no such code, it is usually `kernel-devel`.
- On older SUSE and openSUSE Linux, you must install the `kernel-source` and `kernel-syms` packages.

If you suspect that something has gone wrong with module installation, check that your system is set up as described above and try running the following command, as root:

```
rcvboxdrv setup
```

### 2.3.3 Performing the Installation

Oracle VM VirtualBox is available in a number of package formats native to various common Linux distributions. See chapter 1.4, *Supported Host Operating Systems*, page 5. In addition, there is an alternative generic installer (.run) which should work on most Linux distributions. The generic installer packages are built on EL5 systems and thus require reasonably old versions of glibc, such as version 2.5, and other system libraries.

#### 2.3.3.1 Installing Oracle VM VirtualBox from a Debian/Ubuntu Package

Download the appropriate package for your distribution. The following examples assume that you are installing to a 32-bit Ubuntu Wily system. Use `dpkg` to install the Debian package,as follows:

```
sudo dpkg -i virtualbox-5.0_<version-number>_Ubuntu_wily_i386.deb
```

The installer will also try to build kernel modules suitable for the current running kernel. If the build process is not successful you will be shown a warning and the package will be left unconfigured. Look at `/var/log/vbox-install.log` to find out why the compilation failed. You may have to install the appropriate Linux kernel headers, see chapter 2.3.2, *The Oracle VM VirtualBox Driver Modules*, page 33. After correcting any problems, run the following command:

```
sudo rcvboxdrv setup
```

This will start a second attempt to build the module.

If a suitable kernel module was found in the package or the module was successfully built, the installation script will attempt to load that module. If this fails, please see chapter 12.8.1, *Linux Kernel Module Refuses to Load*, page 291 for further information.

Once Oracle VM VirtualBox has been successfully installed and configured, you can start it by clicking **VirtualBox** in your **Start** menu or from the command line. See chapter 2.3.5, *Starting Oracle VM VirtualBox on Linux*, page 37.

#### 2.3.3.2 Using the Alternative Generic Installer (VirtualBox.run)

The alternative generic installer performs the following steps:

- Unpacks the application files to the target directory `/opt/VirtualBox/`, which cannot be changed.
- Builds and installs the Oracle VM VirtualBox kernel modules: `vboxdrv`, `vboxnetflt`, and `vboxnetadp`.
- Creates `/sbin/rcvboxdrv`, an init script to start the Oracle VM VirtualBox kernel module.
- Creates a new system group called `vboxusers`.
- Creates symbolic links in `/usr/bin` to a shell script `/opt/VirtualBox/VBox` which does some sanity checks and dispatches to the actual executables: `VirtualBox`, `VBoxVRDP`, `VBoxHeadless` and `VBoxManage`.
- Creates `/etc/udev/rules.d/60-vboxdrv.rules`, a description file for udev, if that is present, which makes the USB devices accessible to all users in the `vboxusers` group.
- Writes the installation directory to `/etc/vbox/vbox.cfg`.

The installer must be executed as root with either `install` or `uninstall` as the first parameter. For example:

```
sudo ./VirtualBox.run install
```

Or if you do not have the `sudo` command available, run the following as root instead:

```
./VirtualBox.run install
```

Add every user who needs to access USB devices from a VirtualBox guests to the group `vboxusers`. Either use the GUI user management tools or run the following command as root:

```
sudo usermod -a -G vboxusers username
```

**Note:** The `usermod` command of some older Linux distributions does not support the `-a` option, which adds the user to the given group without affecting membership of other groups. In this case, find out the current group memberships with the `groups` command and add all these groups in a comma-separated list to the command line after the `-G` option. For example: `usermod -G group1,group2,vboxusers username`.

#### 2.3.3.3 Performing a Manual Installation

If you cannot use the shell script installer described in chapter 2.3.3.2, *Using the Alternative Generic Installer (VirtualBox.run)*, page 34, you can perform a manual installation. Run the installer as follows:

```
./VirtualBox.run --keep --noexec
```

This will unpack all the files needed for installation in the directory `install` under the current directory. The Oracle VM VirtualBox application files are contained in `VirtualBox.tar.bz2` which you can unpack to any directory on your system. For example:

```
sudo mkdir /opt/VirtualBox
sudo tar jxf ./install/VirtualBox.tar.bz2 -C /opt/VirtualBox
```

To run the same example as root, use the following commands:

```
mkdir /opt/VirtualBox
tar jxf ./install/VirtualBox.tar.bz2 -C /opt/VirtualBox
```

The sources for Oracle VM VirtualBox's kernel module are provided in the `src` directory. To build the module, change to the directory and use the following command:

```
make
```

If everything builds correctly, run the following command to install the module to the appropriate module directory:

```
sudo make install
```

In case you do not have sudo, switch the user account to root and run the following command:

```
make install
```

The Oracle VM VirtualBox kernel module needs a device node to operate. The above `make` command will tell you how to create the device node, depending on your Linux system. The procedure is slightly different for a classical Linux setup with a `/dev` directory, a system with the now deprecated `devfs` and a modern Linux system with `udev`.

On certain Linux distributions, you might experience difficulties building the module. You will have to analyze the error messages from the build system to diagnose the cause of the problems. In general, make sure that the correct Linux kernel sources are used for the build process.

Note that the `/dev/vboxdrv` kernel module device node must be owned by root:root and must be read/writable only for the user.

Next, you install the system initialization script for the kernel module and activate the initialization script using the right method for your distribution, as follows:

```
cp /opt/VirtualBox/vboxdrv.sh /sbin/rcvboxdrv
```

This example assumes you installed Oracle VM VirtualBox to the `/opt/VirtualBox` directory.

Create a configuration file for Oracle VM VirtualBox, as follows:

```
mkdir /etc/vbox
echo INSTALL_DIR=/opt/VirtualBox > /etc/vbox/vbox.cfg
```

Create the following symbolic links:

```
ln -sf /opt/VirtualBox/VBox.sh /usr/bin/VirtualBox
ln -sf /opt/VirtualBox/VBox.sh /usr/bin/VBoxManage
ln -sf /opt/VirtualBox/VBox.sh /usr/bin/VBoxHeadless
```

#### 2.3.3.4 Updating and Uninstalling Oracle VM VirtualBox

Before updating or uninstalling Oracle VM VirtualBox, you must terminate any virtual machines which are currently running and exit the Oracle VM VirtualBox or VBoxSVC applications. To update Oracle VM VirtualBox, simply run the installer of the updated version. To uninstall Oracle VM VirtualBox, run the installer as follows:

```
sudo ./VirtualBox.run uninstall
```

As root, you can use the following command:

```
./VirtualBox.run uninstall
```

You can uninstall the .run package as follows:

```
/opt/VirtualBox/uninstall.sh
```

To manually uninstall Oracle VM VirtualBox, perform the manual installation steps in reverse order.

#### 2.3.3.5 Automatic Installation of Debian Packages

The Debian packages will request some user feedback when installed for the first time. The debconf system is used to perform this task. To prevent any user interaction during installation, default values can be defined. A file `vboxconf` can contain the following debconf settings:

```
virtualbox virtualbox/module-compilation-allowed boolean true
virtualbox virtualbox/delete-old-modules boolean true
```

The first line enables compilation of the vboxdrv kernel module if no module was found for the current kernel. The second line enables the package to delete any old vboxdrv kernel modules compiled by previous installations.

These default settings can be applied prior to the installation of the Oracle VM VirtualBox Debian package, as follows:

```
debconf-set-selections vboxconf
```

In addition there are some common configuration options that can be set prior to the installation. See chapter 2.3.3.7, *Automatic Installation Options*, page 36.

#### 2.3.3.6 Automatic Installation of RPM Packages

The RPM format does not provide a configuration system comparable to the debconf system. See chapter 2.3.3.7, *Automatic Installation Options*, page 36 for how to set some common installation options provided by Oracle VM VirtualBox.

#### 2.3.3.7 Automatic Installation Options

To configure the installation process for .deb and .rpm packages, you can create a response file named `/etc/default/virtualbox`. The automatic generation of the udev rule can be prevented with the following setting:

```
INSTALL_NO_UDEV=1
```

The creation of the group vboxusers can be prevented as follows:

```
INSTALL_NO_GROUP=1
```

If the following line is specified, the package installer will not try to build the `vboxdrv` kernel module if no module fitting the current kernel was found.

```
INSTALL_NO_VBOXDRV=1
```

### 2.3.4 The vboxusers Group

The Linux installers create the system user group `vboxusers` during installation. Any system user who is going to use USB devices from Oracle VM VirtualBox guests must be a member of that group. A user can be made a member of the group `vboxusers` through the GUI user/group management or using the following command:

```
sudo usermod -a -G vboxusers username
```

### 2.3.5 Starting Oracle VM VirtualBox on Linux

The easiest way to start a Oracle VM VirtualBox program is by running the program of your choice (`VirtualBox`, `VBoxManage`, or `VBoxHeadless`) from a terminal. These are symbolic links to `VBox.sh` that start the required program for you.

The following detailed instructions should only be of interest if you wish to execute Oracle VM VirtualBox without installing it first. You should start by compiling the `vboxdrv` kernel module and inserting it into the Linux kernel. Oracle VM VirtualBox consists of a service daemon, `VBoxSVC`, and several application programs. The daemon is automatically started if necessary. All Oracle VM VirtualBox applications will communicate with the daemon through UNIX local domain sockets. There can be multiple daemon instances under different user accounts and applications can only communicate with the daemon running under the user account as the application. The local domain socket resides in a subdirectory of your system's directory for temporary files called `.vbox-<username>-ipc`. In case of communication problems or server startup problems, you may try to remove this directory.

All Oracle VM VirtualBox applications (`VirtualBox`, `VBoxManage`, and `VBoxHeadless`) require the Oracle VM VirtualBox directory to be in the library path, as follows:

```
LD_LIBRARY_PATH=. ./VBoxManage showvminfo "Windows XP"
```

## 2.4 Installing on Oracle Solaris Hosts

For the specific versions of Oracle Solaris that are supported as host operating systems, see chapter 1.4, *Supported Host Operating Systems*, page 5.

If you have a previously installed instance of Oracle VM VirtualBox on your Oracle Solaris host, please uninstall it first before installing a new instance. See chapter 2.4.4, *Uninstallation*, page 38 for uninstall instructions.

### 2.4.1 Performing the Installation

Oracle VM VirtualBox is available as a standard Oracle Solaris package. Download the Oracle VM VirtualBox SunOS package which includes the 64-bit versions of Oracle VM VirtualBox. *The installation must be performed as root and from the global zone* as the Oracle VM VirtualBox installer loads kernel drivers which cannot be done from non-global zones. To verify which zone you are currently in, execute the `zonename` command. Execute the following commands:

```
gunzip -cd VirtualBox-<version-number>-SunOS.tar.gz | tar xvf -
```

The Oracle VM VirtualBox kernel package is no longer a separate package and has been integrated into the main package. Install the Oracle VM VirtualBox package as follows:

```
pkgadd -d VirtualBox-<version-number>-SunOS.pkg
```

The installer will then prompt you to enter the package you wish to install. Choose **1** or **all** and proceed. Next the installer will ask you if you want to allow the postinstall script to be executed. Choose **y** and proceed, as it is essential to execute this script which installs the Oracle

VM VirtualBox kernel module. Following this confirmation the installer will install Oracle VM VirtualBox and execute the postinstall setup script.

Once the postinstall script has been executed your installation is now complete. You may now safely delete the uncompressed package and `autoresponse` files from your system. Oracle VM VirtualBox is installed in `/opt/VirtualBox`.

> **Note:** If you need to use Oracle VM VirtualBox from non-global zones, see chapter 2.4.6, *Configuring a Zone for Running Oracle VM VirtualBox*, page 39.

### 2.4.2 The vboxuser Group

The installer creates the system user group `vboxuser` during installation for Oracle Solaris hosts that support the USB features required by Oracle VM VirtualBox. Any system user who is going to use USB devices from Oracle VM VirtualBox guests must be a member of this group. A user can be made a member of this group through the GUI user/group management or at the command line by executing as root:

```
usermod -G vboxuser username
```

Note that adding an active user to that group will require that user to log out and back in again. This should be done manually after successful installation of the package.

### 2.4.3 Starting Oracle VM VirtualBox on Oracle Solaris

The easiest way to start a Oracle VM VirtualBox program is by running the program of your choice (`VirtualBox`, `VBoxManage`, or `VBoxHeadless`) from a terminal. These are symbolic links to `VBox.sh` that start the required program for you.

Alternatively, you can directly invoke the required programs from `/opt/VirtualBox`. Using the links provided is easier as you do not have to enter the full path.

You can configure some elements of the `VirtualBox` Qt GUI, such as fonts and colours, by running `VBoxQtconfig` from the terminal.

### 2.4.4 Uninstallation

Uninstallation of Oracle VM VirtualBox on Oracle Solaris requires root permissions. To perform the uninstallation, start a root terminal session and run the following command:

```
pkgrm SUNWvbox
```

After confirmation, this will remove Oracle VM VirtualBox from your system.

If you are uninstalling Oracle VM VirtualBox version 3.0 or lower, you need to remove the Oracle VM VirtualBox kernel interface package, as follows:

```
pkgrm SUNWvboxkern
```

### 2.4.5 Unattended Installation

To perform a non-interactive installation of Oracle VM VirtualBox there is a response file named `autoresponse`, that the installer will use for responses to inputs rather than ask them from you.

Extract the tar.gz package as described in the normal installation instructions. Then open a root terminal session and run the following command:

```
pkgadd -d VirtualBox-<version-number>-SunOS-x86 -n -a autoresponse SUNWvbox
```

To perform a non-interactive uninstallation, open a root terminal session and run the following command:

```
pkgrm -n -a /opt/VirtualBox/autoresponse SUNWvbox
```

### 2.4.6 Configuring a Zone for Running Oracle VM VirtualBox

Assuming that Oracle VM VirtualBox has already been installed into your zone, you need to give the zone access to Oracle VM VirtualBox's device node. This is done by performing the following steps. Start a root terminal and run the following command:

```
zonecfg -z vboxzone
```

Replace "vboxzone" with the name of the zone where you intend to run Oracle VM VirtualBox.
Use`zonecfg` to add the `device` resource and `match` properties to the zone, as follows:

```
zonecfg:vboxzone>add device
zonecfg:vboxzone:device>set match=/dev/vboxdrv
zonecfg:vboxzone:device>end
zonecfg:vboxzone>add device
zonecfg:vboxzone:device>set match=/dev/vboxdrvu
zonecfg:vboxzone:device>end
zonecfg:vboxzone>exit
```

If you are running Oracle VM VirtualBox 2.2.0 or above on Oracle Solaris 11 or above, you may also add a device for `/dev/vboxusbmon`, similar to that shown above. This does not apply to Oracle Solaris 10 hosts, due to lack of USB support.

If you are not using sparse root zones, you will need to loopback mount `/opt/VirtualBox` from the global zone into the non-global zone at the same path. This is specified below using the `dir` attribute and the `special` attribute. For example:

```
zonecfg:vboxzone>add fs
zonecfg:vboxzone:device>set dir=/opt/VirtualBox
zonecfg:vboxzone:device>set special=/opt/VirtualBox
zonecfg:vboxzone:device>set type=lofs
zonecfg:vboxzone:device>end
zonecfg:vboxzone>exit
```

Reboot the zone using `zoneadm` and you should be able to run Oracle VM VirtualBox from within the configured zone.

# 3 Configuring Virtual Machines

This chapter provides detailed steps for configuring an Oracle VM VirtualBox virtual machine (VM). For an introduction to Oracle VM VirtualBox and steps to get your first virtual machine running, see chapter 1, *First Steps*, page 1.

You have considerable latitude when deciding what virtual hardware to provide to the guest. Use virtual hardware to communicate with the host system or with other guests. For example, you can use virtual hardware in the following ways:

- Have Oracle VM VirtualBox present an ISO CD-ROM image to a guest system as if it were a physical CD-ROM.
- Provide a guest system access to the physical network through its virtual network card.
- Provide the host system, other guests, and computers on the Internet access to the guest system.

## 3.1 Supported Guest Operating Systems

Because Oracle VM VirtualBox is designed to provide a generic virtualization environment for x86 systems, it can run operating systems (OSes) of any kind. However, Oracle VM VirtualBox focuses on the following guest systems:

- **Windows NT 4.0:**
  - Fully supports all versions, editions, and service packs. Note that you might encounter issues with some older service packs, so install at least service pack 6a.
  - Guest Additions are available with a limited feature set.
- **Windows 2000, Windows XP, Windows Server 2003, Windows Vista, Windows Server 2008, Windows 7, Windows Server 2008 R2, Windows 8, Windows Server 2012, Windows 8.1, Windows Server 2012 R2, Windows 10 (non-Insider Preview releases), Windows Server 2016, Windows Server 2019:**
  - Fully supports all versions, editions, and service packs, including 64-bit versions.
  - Note that you must enable hardware virtualization when running at least Windows 8.
  - Guest Additions are available.
- **MS-DOS, Windows 3.x, Windows 95, Windows 98, Windows ME:**
  - Limited testing has been performed.
  - Use beyond legacy installation mechanisms is not recommended.
  - Guest Additions are not available.
- **Linux 2.4:**

  Limited support.
- **Linux 2.6:**
  - Fully supports all versions and editions, both 32-bit and 64-bit.

  - For best performance, use at least Linux kernel version 2.6.13.
  - Guest Additions are available.

**Note:** Certain Linux kernel releases have bugs that prevent them from executing in a virtual environment. See chapter 12.4.3, *Buggy Linux 2.6 Kernel Versions*, page 288.

- **Linux 3.x and later:**
  - Fully supports all versions and editions, both 32-bit and 64-bit.
  - Guest Additions are available.
- **Oracle Solaris 10 and Oracle Solaris 11:**
  - Fully supports all versions starting with Oracle Solaris 10 8/08 and Oracle Solaris 11.
  - Supports 64-bit prior to Oracle Solaris 11 11/11, and 32-bit.
  - Guest Additions are available.
- **FreeBSD:**
  - Limited support.
  - Note that you must enable hardware virtualization when running FreeBSD.
  - Guest Additions are not available.
- **OpenBSD:**
  - Supports at least version 3.7.
  - Note that you must enable hardware virtualization when running OpenBSD.
  - Guest Additions are not available.
- **OS/2 Warp 4.5:**
  - Only MCP2 is supported. Other OS/2 versions might not work.
  - Note that you must enable hardware virtualization when running OS/2 Warp 4.5.
  - Guest Additions are available with a limited feature set. See chapter 14, *Known Limitations*, page 301.
- **Mac OS X:**
  - Oracle VM VirtualBox 3.2 added experimental support for Mac OS X guests, with restrictions. See chapter 3.1.1, *Mac OS X Guests*, page 41 and chapter 14, *Known Limitations*, page 301.
  - Guest Additions are not available.

### 3.1.1 Mac OS X Guests

Oracle VM VirtualBox enables you to install and execute unmodified versions of Mac OS X guests on supported host hardware. Note that this feature is experimental and thus unsupported.

Oracle VM VirtualBox is the first product to provide the modern PC architecture expected by OS X without requiring any of the modifications used by competing virtualization solutions. For example, some competing solutions perform modifications to the Mac OS X install DVDs, such as a different boot loader and replaced files.

Be aware of the following important issues before you attempt to install a Mac OS X guest:

- Mac OS X is commercial, licensed software and contains **both license and technical restrictions** that limit its use to certain hardware and usage scenarios. You must understand and comply with these restrictions.

  In particular, Apple prohibits the installation of most versions of Mac OS X on non-Apple hardware.

  These license restrictions are also enforced on a technical level. Mac OS X verifies that it is running on Apple hardware. Most DVDs that accompany Apple hardware check for the exact model. These restrictions are *not* circumvented by Oracle VM VirtualBox and continue to apply.

- Only **CPUs** that are known and tested by Apple are supported. As a result, if your Intel CPU is newer than the Mac OS X build, or if you have a non-Intel CPU, you will likely encounter a panic during bootup with an "Unsupported CPU" exception.

  Ensure that you use the Mac OS X DVD that comes with your Apple hardware.

- The Mac OS X installer expects the hard disk to be *partitioned*. So, the installer will not offer a partition selection to you. Before you can install the software successfully, start the Disk Utility from the Tools menu and partition the hard disk. Close the Disk Utility and proceed with the installation.

- In addition, Mac OS X support in Oracle VM VirtualBox is an experimental feature. See chapter 14, *Known Limitations*, page 301.

### 3.1.2 64-bit Guests

Oracle VM VirtualBox enables you to run 64-bit guest OSes even on a 32-bit host OS. To run a 64-bit guest OS on a 32-bit host system, ensure that you meet the following conditions:

- You need a 64-bit processor that has hardware virtualization support. See chapter 10.3, *Hardware vs. Software Virtualization*, page 270.

- You must enable hardware virtualization for the particular VM that requires 64-bit support. Software virtualization is not supported for 64-bit VMs.

- To use 64-bit guest support on a 32-bit host OS, you must select a 64-bit OS for the particular VM. Since supporting 64 bits on 32-bit hosts incurs additional overhead, Oracle VM VirtualBox only enables this support only upon explicit request.

  64-bit hosts typically come with hardware virtualization support. So, you can install a 64-bit guest OS in the guest regardless of the settings.

**Warning:** Be sure to enable **I/O APIC** for virtual machines that you intend to use in 64-bit mode. This is especially true for 64-bit Windows VMs. See chapter 3.4.2, *Advanced Tab*, page 46. For 64-bit Windows guests, ensure that the VM uses the **Intel networking device** because there is no 64-bit driver support for the AMD PCNet card. See chapter 6.1, *Virtual Networking Hardware*, page 98.

If you use the **Create VM** wizard of the Oracle VM VirtualBox graphical user interface (GUI), Oracle VM VirtualBox automatically uses the correct settings for each selected 64-bit OS type. See chapter 1.8, *Creating Your First Virtual Machine*, page 8.

## 3.2 Unattended Guest Installation

Oracle VM VirtualBox can install a guest OS automatically. You only need to provide the installation medium and a few other parameters, such as the name of the default user.

Performing an unattended guest installation involves the following steps:

- **Create a new VM.** Use one of the following methods:
  - The VirtualBox Manager, see chapter 1.8, *Creating Your First Virtual Machine*, page 8.
  - The `VBoxManage createvm` command, see chapter 8.7, *VBoxManage createvm*, page 134.

  For the new VM, choose the guest OS type and accept the default settings for that OS. The following sections in this chapter describe how to change the settings for a VM.

- **Prepare the VM for unattended guest installation.** Use the `VBoxManage unattended` command, see chapter 8.44, *VBoxManage unattended*, page 203.

  During this step, Oracle VM VirtualBox scans the installation medium and changes certain parameters to ensure a seamless installation as a guest running on Oracle VM VirtualBox.

- **Start the VM.** Use the VirtualBox Manager or the `VBoxManage startvm` command.

  When you start the VM, the unattended installation is performed automatically.

  The installation operation changes the boot device order to boot the virtual hard disk first and then the virtual DVD drive. If the virtual hard disk is empty prior to the automatic installation, the VM boots from the virtual DVD drive and begins the installation.

  If the virtual hard disk contains a bootable OS, the installation operation exits. In this case, change the boot device order manually by pressing F12 during the BIOS splash screen.

chapter 3.2.1, *An Example of Unattended Guest Installation*, page 43 describes how to perform an unattended guest installation for an Oracle Linux guest.

### 3.2.1 An Example of Unattended Guest Installation

The following example shows how to perform an unattended guest installation for an Oracle Linux VM. The example uses various `VBoxManage` commands to prepare the guest VM. The `VBoxManage unattended install` command is then used to install and configure the guest OS.

1. Create the virtual machine.

   ```
   # VM="ol7-autoinstall"
   # VBoxManage list ostypes
   # VBoxManage createvm --name $VM --ostype "Oracle_64" --register
   ```

   Note the following:

   - The $VM variable represents the name of the VM.
   - The `VBoxManage list ostypes` command lists the guest OSes supported by Oracle VM VirtualBox, including the name used for each OS in the `VBoxManage` commands.
   - A 64-bit Oracle Linux 7 VM is created and registered with Oracle VM VirtualBox.
   - The VM has a unique UUID.
   - An XML settings file is generated.

2. Create a virtual hard disk and storage devices for the VM.

```
# VBoxManage createhd --filename /VirtualBox/$VM/$VM.vdi --size 32768
# VBoxManage storagectl $VM --name "SATA Controller" --add sata --controller IntelAHCI
# VBoxManage storageattach $VM --storagectl "SATA Controller" --port 0 --device 0 \
--type hdd --medium /VirtualBox/$VM/$VM.vdi
# VBoxManage storagectl $VM --name "IDE Controller" --add ide
# VBoxManage storageattach $VM --storagectl "IDE Controller" --port 0 --device 0 \
--type dvddrive --medium /u01/Software/OL/OracleLinux-R7-U6-Server-x86_64-dvd.iso
```

The previous commands do the following:

- Create a 32768 MB virtual hard disk.
- Create a SATA storage controller and attach the virtual hard disk.
- Create an IDE storage controller for a virtual DVD drive and attach an Oracle Linux installation ISO.

3. (Optional) Configure some settings for the VM.

```
# VBoxManage modifyvm $VM --ioapic on
# VBoxManage modifyvm $VM --boot1 dvd --boot2 disk --boot3 none --boot4 none
# VBoxManage modifyvm $VM --memory 8192 --vram 128
```

The previous commands do the following:

- Enable I/O APIC for the motherboard of the VM.
- Configure the boot device order for the VM.
- Allocate 8192 MB of RAM and 128 MB of video RAM to the VM.

4. Perform an unattended install of the OS.

```
# VBoxManage unattended install $VM \
--iso=/u01/Software/OL/OracleLinux-R7-U6-Server-x86_64-dvd.iso \
--user=<login> --full-user-name=<name> --password <password> \
--install-additions --time-zone=CET
```

The previous command does the following:

- Specifies an Oracle Linux ISO as the installation ISO.
- Specifies a login name, full name, and login password for a default user on the guest OS.

  Note that the specified password is also used for the root user account on the guest.
- Installs the Guest Additions on the VM.
- Sets the time zone for the guest OS to Central European Time (CET).

5. Start the virtual machine.

   This step completes the unattended installation process.

```
# VBoxManage startvm $VM --type headless
```

The VM starts in headless mode, which means that the VirtualBox Manager window does not open.

6. (Optional) Update the guest OS to use the latest Oracle Linux packages.

   On the guest VM, run the following command:

```
# yum update
```

## 3.3 Emulated Hardware

Oracle VM VirtualBox virtualizes nearly all hardware of the host. Depending on a VM's configuration, the guest will see the following virtual hardware:

- **Input devices.** By default, Oracle VM VirtualBox emulates a standard PS/2 keyboard and mouse. These devices are supported by almost all past and present OSes.

  In addition, Oracle VM VirtualBox can provide virtual USB input devices to avoid having to capture mouse and keyboard, as described in chapter 1.9.2, *Capturing and Releasing Keyboard and Mouse*, page 12.

- **Graphics.** The Oracle VM VirtualBox graphics device, sometimes referred to as a VGA device, is not based on any physical counterpart. This is unlike nearly all other emulated devices. It is a simple, synthetic device which provides compatibility with standard VGA and several extended registers used by the VESA BIOS Extensions (VBE).

- **Storage.** Oracle VM VirtualBox currently emulates the standard ATA interface found on Intel PIIX3/PIIX4 chips, the SATA (AHCI) interface, and two SCSI adapters (LSI Logic and BusLogic). See chapter 5.1, *Hard Disk Controllers: IDE, SATA (AHCI), SCSI, SAS, USB MSD, NVMe*, page 83 for details. Whereas providing one of these would be enough for Oracle VM VirtualBox by itself, this multitude of storage adapters is required for compatibility with other hypervisors. Windows is particularly picky about its boot devices, and migrating VMs between hypervisors is very difficult or impossible if the storage controllers are different.

- **Networking.** See chapter 6.1, *Virtual Networking Hardware*, page 98.

- **USB.** Oracle VM VirtualBox emulates three USB host controllers: xHCI, EHCI, and OHCI. While xHCI handles all USB transfer speeds, only guest OSes released approximately after 2011 support xHCI. Note that for Windows 7 guests, 3rd party drivers must be installed for xHCI support.

  Older OSes typically support OHCI and EHCI. The two controllers are needed because OHCI only handles USB low-speed and full-speed devices (both USB 1.x and 2.0), while EHCI only handles high-speed devices (USB 2.0 only).

  The emulated USB controllers do not communicate directly with devices on the host but rather with a virtual USB layer which abstracts the USB protocol and enables the use of remote USB devices.

- **Audio.** See chapter 3.8, *Audio Settings*, page 53.

## 3.4 General Settings

In the **Settings** window, under **General**, you can configure the most fundamental aspects of the virtual machine such as memory and essential hardware. The following tabs are available.

### 3.4.1 Basic Tab

In the **Basic** tab of the **General** settings category, you can find these settings:

- **Name:** The name under which the VM is shown in the list of VMs in the main window. Under this name, Oracle VM VirtualBox also saves the VM's configuration files. By changing the name, Oracle VM VirtualBox renames these files as well. As a result, you can only use characters which are allowed in your host OS's file names.

  Note that internally, Oracle VM VirtualBox uses unique identifiers (UUIDs) to identify virtual machines. You can display these with `VBoxManage`.

- **Type:** The type of the guest OS for the VM. This is the same setting that is specified in the **New Virtual Machine** wizard. See chapter 1.8, *Creating Your First Virtual Machine*, page 8.

  Whereas the default settings of a newly created VM depend on the selected OS type, changing the type later has no effect on VM settings. This value is purely informational and decorative.

- **Version:** The version of the guest OS for the VM. This is the same setting that is specified in the **New Virtual Machine** wizard. See chapter 1.8, *Creating Your First Virtual Machine*, page 8.

### 3.4.2 Advanced Tab

The following settings are available in the **Advanced** tab:

- **Snapshot Folder:** By default, Oracle VM VirtualBox saves snapshot data together with your other Oracle VM VirtualBox configuration data. See chapter 10.1, *Where Oracle VM VirtualBox Stores its Files*, page 265. With this setting, you can specify any other folder for each VM.

- **Shared Clipboard:** You can select here whether the clipboard of the guest OS should be shared with that of your host. If you select **Bidirectional**, then Oracle VM VirtualBox will always make sure that both clipboards contain the same data. If you select **Host to Guest** or **Guest to Host**, then Oracle VM VirtualBox will only ever copy clipboard data in one direction.

  Clipboard sharing requires that the Oracle VM VirtualBox Guest Additions be installed. In such a case, this setting has no effect. See chapter 4, *Guest Additions*, page 62.

  For security reasons, the shared clipboard is disabled by default. This setting can be changed at any time using the **Shared Clipboard** menu item in the **Devices** menu of the virtual machine.

- **Drag and Drop:** This setting enables support for drag and drop. Select an object, such as a file, from the host or guest and directly copy or open it on the guest or host. Multiple per-VM drag and drop modes allow restricting access in either direction.

  For drag and drop to work the Guest Additions need to be installed on the guest.

  **Note:** Drag and drop is disabled by default. This setting can be changed at any time using the **Drag and Drop** menu item in the **Devices** menu of the virtual machine.

  See chapter 4.4, *Drag and Drop*, page 72.

### 3.4.3 Description Tab

On the **Description** tab you can enter a description for your virtual machine. This has no effect on the functionality of the machine, but you may find this space useful to note down things such as the configuration of a virtual machine and the software that has been installed into it.

To insert a line break into the **Description** text field, press Shift+Enter.

### 3.4.4 Disk Encryption Tab

The **Disk Encryption** tab enables you to encrypt disks that are attached to the virtual machine.

To enable disk encryption, select the **Enable Disk Encryption** check box.

Settings are available to configure the cipher used for encryption and the encryption password.

## 3.5 System Settings

The **System** category groups various settings that are related to the basic hardware that is presented to the virtual machine.

> **Note:** As the activation mechanism of Microsoft Windows is sensitive to hardware changes, if you are changing hardware settings for a Windows guest, some of these changes may trigger a request for another activation with Microsoft.

The following tabs are available.

### 3.5.1 Motherboard Tab

On the **Motherboard** tab, you can configure virtual hardware that would normally be on the motherboard of a real computer.

- **Base Memory:** Sets the amount of RAM that is allocated and given to the VM when it is running. The specified amount of memory will be requested from the host OS, so it must be available or made available as free memory on the host when attempting to start the VM and will not be available to the host while the VM is running. This is the same setting that was specified in the **New Virtual Machine** wizard, as described in chapter 1.8, *Creating Your First Virtual Machine*, page 8.

  Generally, it is possible to change the memory size after installing the guest OS. But you must not reduce the memory to an amount where the OS would no longer boot.

- **Boot Order:** Determines the order in which the guest OS will attempt to boot from the various virtual boot devices. Analogous to a real PC's BIOS setting, Oracle VM VirtualBox can tell a guest OS to start from the virtual floppy, the virtual CD/DVD drive, the virtual hard drive (each of these as defined by the other VM settings), the network, or none of these.

  If you select **Network**, the VM will attempt to boot from a network using the PXE mechanism. This needs to be configured in detail on the command line. See chapter 8.8, *VBoxManage modifyvm*, page 135.

- **Chipset:** You can select which chipset will be presented to the virtual machine. In legacy versions of Oracle VM VirtualBox, PIIX3 was the only available option. For modern guest OSes such as Mac OS X, that old chipset is no longer well supported. As a result, Oracle VM VirtualBox supports an emulation of the more modern ICH9 chipset, which supports PCI express, three PCI buses, PCI-to-PCI bridges and Message Signaled Interrupts (MSI). This enables modern OSes to address more PCI devices and no longer requires IRQ sharing. Using the ICH9 chipset it is also possible to configure up to 36 network cards, up to 8 network adapters with PIIX3. Note that the ICH9 support is experimental and not recommended for guest OSes which do not require it.

- **Pointing Device:** The default virtual pointing devices for older guests is the traditional PS/2 mouse. If set to *USB tablet*, Oracle VM VirtualBox reports to the virtual machine that a USB tablet device is present and communicates mouse events to the virtual machine through this device. The third setting is a *USB Multi-Touch Tablet* which is suited for recent Windows guests.

  Using the virtual USB tablet has the advantage that movements are reported in absolute coordinates, instead of as relative position changes. This enables Oracle VM VirtualBox to translate mouse events over the VM window into tablet events without having to "capture" the mouse in the guest as described in chapter 1.9.2, *Capturing and Releasing Keyboard*

*and Mouse*, page 12. This makes using the VM less tedious even if Guest Additions are not installed.

- **Enable I/O APIC:** Advanced Programmable Interrupt Controllers (APICs) are a newer x86 hardware feature that have replaced old-style Programmable Interrupt Controllers (PICs) in recent years. With an I/O APIC, OSes can use more than 16 interrupt requests (IRQs) and therefore avoid IRQ sharing for improved reliability.

  **Note:** Enabling the I/O APIC is *required* for 64-bit guest OSes, especially Windows Vista. It is also required if you want to use more than one virtual CPU in a virtual machine.

  However, software support for I/O APICs has been unreliable with some OSes other than Windows. Also, the use of an I/O APIC slightly increases the overhead of virtualization and therefore slows down the guest OS a little.

  **Warning:** All Windows OSes starting with Windows 2000 install different kernels, depending on whether an I/O APIC is available. As with ACPI, the I/O APIC therefore *must not be turned off after installation* of a Windows guest OS. Turning it on after installation will have no effect however.

- **Enable EFI:** Enables Extensible Firmware Interface (EFI), which replaces the legacy BIOS and may be useful for certain advanced use cases. See chapter 3.14, *Alternative Firmware (EFI)*, page 58.

- **Hardware Clock in UTC Time:** If selected, Oracle VM VirtualBox will report the system time in UTC format to the guest instead of the local (host) time. This affects how the virtual real-time clock (RTC) operates and may be useful for UNIX-like guest OSes, which typically expect the hardware clock to be set to UTC.

In addition, you can turn off the **Advanced Configuration and Power Interface (ACPI)** which Oracle VM VirtualBox presents to the guest OS by default.

ACPI is the current industry standard to allow OSes to recognize hardware, configure motherboards and other devices and manage power. As all modern PCs contain this feature and Windows and Linux have been supporting it for years, it is also enabled by default in Oracle VM VirtualBox. ACPI can only be turned off using the command line. See chapter 8.8, *VBoxManage modifyvm*, page 135.

**Warning:** All Windows OSes starting with Windows 2000 install different kernels, depending on whether ACPI is available. This means that ACPI *must not be turned off* after installation of a Windows guest OS. However, turning it on after installation will have no effect.

### 3.5.2 Processor Tab

On the **Processor** tab, you can configure settings for the CPU used by the virtual machine.

- **Processor(s):** Sets the number of virtual CPU cores the guest OSes can see. Oracle VM VirtualBox supports symmetrical multiprocessing (SMP) and can present up to 32 virtual CPU cores to each virtual machine.

  You should not configure virtual machines to use more CPU cores than are available physically. This includes real cores, with no hyperthreads.

- **Execution Cap:** Configures the CPU execution cap. This limits the amount of time a host CPU spends to emulate a virtual CPU. The default setting is 100%, meaning that there is no limitation. A setting of 50% implies a single virtual CPU can use up to 50% of a single host CPU. Note that limiting the execution time of the virtual CPUs may cause guest timing problems.

  A warning is displayed at the bottom of the Processor tab if an Execution Cap setting is made that may affect system performance.

- **Enable PAE/NX:** Determines whether the PAE and NX capabilities of the host CPU will be exposed to the virtual machine.

  PAE stands for Physical Address Extension. Normally, if enabled and supported by the OS, then even a 32-bit x86 CPU can access more than 4 GB of RAM. This is made possible by adding another 4 bits to memory addresses, so that with 36 bits, up to 64 GB can be addressed. Some OSes, such as Ubuntu Server, require PAE support from the CPU and cannot be run in a virtual machine without it.

- **Enable Nested VT-x/AMD-V**: Enables nested virtualization, with passthrough of hardware virtualization functions to the guest VM.

  This feature is available on host systems that use an AMD CPU. For Intel CPUs, the option is grayed out.

With virtual machines running modern server OSes, Oracle VM VirtualBox also supports CPU hot-plugging. For details, see chapter 9.4, *CPU Hot-Plugging*, page 212.

### 3.5.3 Acceleration Tab

On this tab, you can configure Oracle VM VirtualBox to use hardware virtualization extensions that your host CPU supports.

- **Paravirtualization Interface:** Oracle VM VirtualBox provides paravirtualization interfaces to improve time-keeping accuracy and performance of guest OSes. The options available are documented under the `paravirtprovider` option in chapter 8.8, *VBoxManage modifyvm*, page 135. For further details on the paravirtualization providers, see chapter 10.4, *Paravirtualization Providers*, page 271.

- **Hardware Virtualization:** You can select for each virtual machine individually whether Oracle VM VirtualBox should use software or hardware virtualization.

  - **Enable VT-x/AMD-V:** Enables Intel VT-x and AMD-V hardware extensions if the host CPU supports them.
  - **Enable Nested Paging:** If the host CPU supports the nested paging (AMD-V) or EPT (Intel VT-x) features, then you can expect a significant performance increase by enabling nested paging in addition to hardware virtualization. For technical details, see chapter 10.7, *Nested Paging and VPIDs*, page 275. For Intel EPT security recommendations, see chapter 13.4.1, *CVE-2018-3646*, page 299.

  Advanced users may be interested in technical details about software versus hardware virtualization. See chapter 10.3, *Hardware vs. Software Virtualization*, page 270.

In most cases, the default settings on the **Acceleration** tab will work well. Oracle VM VirtualBox selects sensible defaults, depending on the OS that you selected when you created the virtual machine. In certain situations, however, you may want to change the preconfigured defaults.

## 3.6 Display Settings

The following tabs are available for configuring the display for a virtual machine.

### 3.6.1 Screen Tab

- **Video Memory:** Sets the size of the memory provided by the virtual graphics card available to the guest, in MB. As with the main memory, the specified amount will be allocated from the host's resident memory. Based on the amount of video memory, higher resolutions and color depths may be available.

  The GUI will show a warning if the amount of video memory is too small to be able to switch the VM into full screen mode. The minimum value depends on the number of virtual monitors, the screen resolution and the color depth of the host display as well as on the use of *3D acceleration* and *2D video acceleration*. A rough estimate is (*color depth* / 8) x *vertical pixels* x *horizontal pixels* x *number of screens* = *number of bytes*. Extra memory may be required if display acceleration is used.

- **Monitor Count:** With this setting, Oracle VM VirtualBox can provide more than one virtual monitor to a virtual machine. If a guest OS supports multiple attached monitors, Oracle VM VirtualBox can pretend that multiple virtual monitors are present. Up to eight such virtual monitors are supported.

  The output of the multiple monitors are displayed on the host in multiple VM windows which are running side by side. However, in full screen and seamless mode, they use the available physical monitors attached to the host. As a result, for full screen and seamless modes to work with multiple monitors, you will need at least as many physical monitors as you have virtual monitors configured, or Oracle VM VirtualBox will report an error.

  You can configure the relationship between guest and host monitors using the **View** menu by pressing Host key + Home when you are in full screen or seamless mode.

  See also chapter 14, *Known Limitations*, page 301.

- **Scale Factor:** Enables scaling of the display size. For multiple monitor displays, you can set the scale factor for individual monitors, or globally for all of the monitors. Use the slider to select a scaling factor up to 200%.

  You can set a default scale factor for all VMs. Use the **Display** tab in the Global Settings dialogs.

- **Enable 3D Acceleration:** If a virtual machine has Guest Additions installed, you can select here whether the guest should support accelerated 3D graphics. See chapter 4.5.1, *Hardware 3D Acceleration (OpenGL and Direct3D 8/9)*, page 74.

- **Enable 2D Video Acceleration:** If a virtual machine with Microsoft Windows has Guest Additions installed, you can select here whether the guest should support accelerated 2D video graphics. See chapter 4.5.2, *Hardware 2D Video Acceleration for Windows Guests*, page 75.

- **Graphics Controller:** Specifies the graphics adapter type used by the guest VM. Note that you must install the Guest Additions on the guest VM to specify the VBoxSVGA or VMSVGA graphics controller. The following options are available:

  – **VBoxSVGA:** The default graphics controller for new VMs that use Windows 7 or later.

    This graphics controller improves performance and 3D support when compared to the legacy VBoxVGA option.

  – **VBoxVGA:** Use this graphics controller for legacy guest OSes. This is the default graphics controller for Windows versions before Windows 7 and for Oracle Solaris.

  - **VMSVGA:** Use this graphics controller to emulate a VMware SVGA graphics device. This is the default graphics controller for Linux guests.
  - **None:** Does not emulate a graphics adapter type.

### 3.6.2 Remote Display Tab

On the **Remote Display** tab, if the VirtualBox Remote Display Extension (VRDE) is installed, you can enable the VRDP server that is built into Oracle VM VirtualBox. This enables you to connect to the console of the virtual machine remotely with any standard RDP viewer, such as `mstsc.exe` that comes with Microsoft Windows. On Linux and Oracle Solaris systems you can use the standard open source `rdesktop` program. These features are described in chapter 7.1, *Remote Display (VRDP Support)*, page 110.

- **Enable Server:** Select this check box and configure settings for the remote display connection.

### 3.6.3 Recording Tab

On the **Recording** tab you can enable video and audio recording for a virtual machine and change related settings. Note that these features can be enabled and disabled while a VM is running.

- **Enable Recording:** Select this check box and select a **Recording Mode** option.
- **Recording Mode:** You can choose to record video, audio, or both video and audio.

  Some settings on the **Recording** tab may be grayed out, depending on the **Recording Mode** setting.
- **File Path:** The file where the recording is saved.
- **Frame Size:** The video resolution of the recorded video, in pixels. The drop-down list enables you to select from common frame sizes.
- **Frame Rate:** Use the slider to set the maximum number of video frames per second (FPS) to record. Frames that have a higher frequency are skipped. Increasing this value reduces the number of skipped frames and increases the file size.
- **Quality:** Use the slider to set the the bit rate of the video in kilobits per second. Increasing this value improves the appearance of the video at the cost of an increased file size.
- **Audio Quality:** Use the slider to set the quality of the audio recording. Increasing this value improves the audio quality at the cost of an increased file size.
- **Screens:** For a multiple monitor display, you can select which screens to record video from.

As you adjust the video and audio recording settings, the approximate output file size for a five minute video is shown.

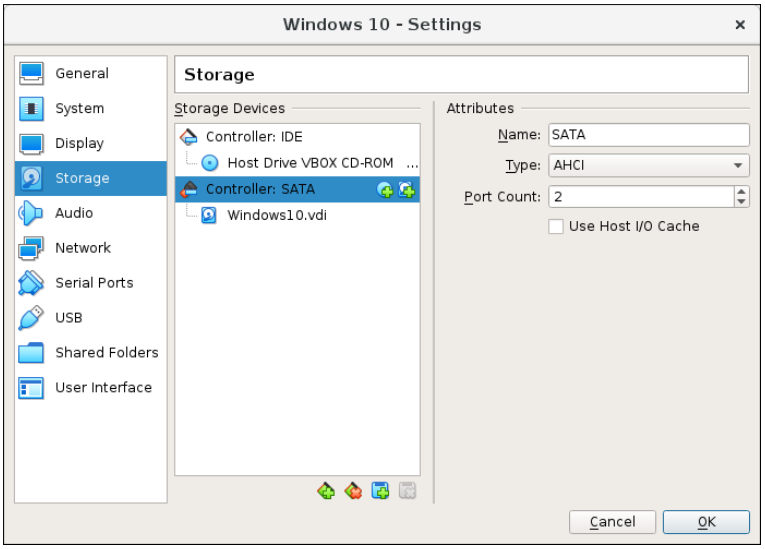
## 3.7 Storage Settings

The **Storage** category in the VM settings enables you to connect virtual hard disk, CD/DVD, and floppy images and drives to your virtual machine.

In a real PC, so-called *storage controllers* connect physical disk drives to the rest of the computer. Similarly, Oracle VM VirtualBox presents virtual storage controllers to a virtual machine. Under each controller, the virtual devices, such as hard disks, CD/DVD or floppy drives, attached to the controller are shown.

**Note:** This section gives a quick introduction to the Oracle VM VirtualBox storage settings. See chapter 5, *Virtual Storage*, page 83 for a full description of the available storage settings in Oracle VM VirtualBox.

If you have used the **Create VM** wizard to create a machine, you will normally see something like the following:




Depending on the guest OS type that you selected when you created the VM, a new VM includes the following storage devices:

- **IDE controller.** A virtual CD/DVD drive is attached to the secondary master port of the IDE controller.
- **SATA controller.** This is a modern type of storage controller for higher hard disk data throughput, to which the virtual hard disks are attached. Initially you will normally have one such virtual disk, but as shown in the previous screenshot, you can have more than one. Each is represented by a disk image file, such as a VDI file in this example.

If you created your VM with an older version of Oracle VM VirtualBox, the default storage layout may differ. You might then only have an IDE controller to which both the CD/DVD drive and the hard disks have been attached. This might also apply if you selected an older OS type when you created the VM. Since older OSes do not support SATA without additional drivers, Oracle VM VirtualBox will make sure that no such devices are present initially. See chapter 5.1, *Hard Disk Controllers: IDE, SATA (AHCI), SCSI, SAS, USB MSD, NVMe*, page 83.

Oracle VM VirtualBox also provides a *floppy controller*. You cannot add devices other than floppy drives to this controller. Virtual floppy drives, like virtual CD/DVD drives, can be connected to either a host floppy drive, if you have one, or a disk image, which in this case must be in RAW format.

You can modify these media attachments freely. For example, if you wish to copy some files from another virtual disk that you created, you can connect that disk as a second hard disk, as in the above screenshot. You could also add a second virtual CD/DVD drive, or change where these items are attached. The following options are available:

- To **add another virtual hard disk, or a CD/DVD or floppy drive**, select the storage controller to which it should be added (IDE, SATA, SCSI, SAS, floppy controller) and then click the **Add Disk** button below the tree. You can then either select **Add CD/DVD Device** or

**Add Hard Disk**. If you clicked on a floppy controller, you can add a floppy drive instead. Alternatively, right-click on the storage controller and select a menu item there.

On the right part of the window, you can then set the following:

1. The **device slot** of the controller that the virtual disk is connected to. IDE controllers have four slots which have traditionally been called primary master, primary slave, secondary master, and secondary slave. By contrast, SATA and SCSI controllers offer you up to 30 slots for attaching virtual devices.
2. The **image file** to use.
   - For virtual hard disks, a button with a drop-down list appears on the right, offering you to either select a **virtual hard disk file** using a standard file dialog or to **create a new hard disk** (image file). The latter option displays the **Create New Disk** wizard, described in chapter 1.8, *Creating Your First Virtual Machine*, page 8.

     For virtual floppy drives, a dialog enables you to create and format a new floppy disk image automatically.

     For details on the image file types that are supported, see chapter 5.2, *Disk Image Files (VDI, VMDK, VHD, HDD)*, page 86.
   - For virtual CD/DVD drives, the image files will typically be in the standard ISO format instead. Most commonly, you will select this option when installing an OS from an ISO file that you have obtained from the Internet. For example, most Linux distributions are available in this way.

     For virtual CD/DVD drives, the following additional options are available:

     * If you select **Host Drive** from the list, then the physical device of the host computer is connected to the VM, so that the guest OS can read from and write to your physical device. This is, for instance, useful if you want to install Windows from a real installation CD. In this case, select your host drive from the drop-down list presented.

       If you want to write, or burn, CDs or DVDs using the host drive, you need to also enable the **Passthrough** option. See chapter 5.9, *CD/DVD Support*, page 94.
     * If you select **Remove Disk from Virtual Drive**, Oracle VM VirtualBox will present an empty CD/DVD drive to the guest into which no media has been inserted.

- To **remove an attachment**, either select it and click on the **Remove** icon at the bottom, or right-click on it and select the menu item.

Removable media, such as CD/DVDs and floppies, can be changed while the guest is running. Since the **Settings** dialog is not available at that time, you can also access these settings from the **Devices** menu of your virtual machine window.

## 3.8 Audio Settings

The **Audio** section in a virtual machine's **Settings** window determines whether the VM will detect a connected sound card, and if the audio output should be played on the host system.

To enable audio for a guest, select the **Enable Audio** check box. The following settings are available:

- **Host Audio Driver:** The audio driver that Oracle VM VirtualBox uses on the host. On a Linux host, depending on your host configuration, you can select between the OSS, ALSA,

or the PulseAudio subsystem. On newer Linux distributions, the PulseAudio subsystem is preferred.

Only OSS is supported on Oracle Solaris hosts. The Oracle Solaris Audio audio backend is no longer supported on Oracle Solaris hosts.

- **Audio Controller:** You can choose between the emulation of an Intel AC'97 controller, an Intel HD Audio controller, or a SoundBlaster 16 card.
- **Enable Audio Output:** Enables audio output only for the VM.
- **Enable Audio Input:** Enables audio input only for the VM.

## 3.9 Network Settings

The **Network** section in a virtual machine's **Settings** window enables you to configure how Oracle VM VirtualBox presents virtual network cards to your VM, and how they operate.

When you first create a virtual machine, Oracle VM VirtualBox by default enables one virtual network card and selects the Network Address Translation (NAT) mode for it. This way the guest can connect to the outside world using the host's networking and the outside world can connect to services on the guest which you choose to make visible outside of the virtual machine.

This default setup is good for the majority of Oracle VM VirtualBox users. However, Oracle VM VirtualBox is extremely flexible in how it can virtualize networking. It supports many virtual network cards per virtual machine. The first four virtual network cards can be configured in detail in the VirtualBox Manager window. Additional network cards can be configured using the `VBoxManage` command.

Many networking options are available. See chapter 6, *Virtual Networking*, page 98 for more information.

## 3.10 Serial Ports

Oracle VM VirtualBox supports the use of virtual serial ports in a virtual machine.

Ever since the original IBM PC, personal computers have been equipped with one or two serial ports, also called COM ports by DOS and Windows. Serial ports were commonly used with modems, and some computer mice used to be connected to serial ports before USB became commonplace.

While serial ports are no longer as common as they used to be, there are still some important uses left for them. For example, serial ports can be used to set up a primitive network over a null-modem cable, in case Ethernet is not available. Also, serial ports are indispensable for system programmers needing to do kernel debugging, since kernel debugging software usually interacts with developers over a serial port. With virtual serial ports, system programmers can do kernel debugging on a virtual machine instead of needing a real computer to connect to.

If a virtual serial port is enabled, the guest OS sees a standard 16550A compatible UART device. Other UART types can be configured using the `VBoxManage modifyvm` command. Both receiving and transmitting data is supported. How this virtual serial port is then connected to the host is configurable, and the details depend on your host OS.

You can use either the Settings tabs or the `VBoxManage` command to set up virtual serial ports. For the latter, see chapter 8.8, *VBoxManage modifyvm*, page 135 for information on the `--uart`, `--uartmode` and `--uarttype` options.

You can configure up to four virtual serial ports per virtual machine. For each device, you must set the following:

1. **Port Number:** This determines the serial port that the virtual machine should see. For best results, use the traditional values as follows:

- COM1: I/O base 0x3F8, IRQ 4
- COM2: I/O base 0x2F8, IRQ 3
- COM3: I/O base 0x3E8, IRQ 4
- COM4: I/O base 0x2E8, IRQ 3

You can also configure a user-defined serial port. Enter an I/O base address and interrupt (IRQ).

See also http://en.wikipedia.org/wiki/COM_(hardware_interface).

2. **Port Mode:** What the virtual port is connected to. For each virtual serial port, you have the following options:

   - **Disconnected:** The guest will see the device, but it will behave as if no cable had been connected to it.
   - **Host Device:** Connects the virtual serial port to a physical serial port on your host. On a Windows host, this will be a name like `COM1`. On Linux or Oracle Solaris hosts, it will be a device node like `/dev/ttyS0`. Oracle VM VirtualBox will then simply redirect all data received from and sent to the virtual serial port to the physical device.
   - **Host Pipe:** Configure Oracle VM VirtualBox to connect the virtual serial port to a software pipe on the host. This depends on your host OS, as follows:
     - On a Windows host, data will be sent and received through a named pipe. The pipe name must be in the format `\\.\pipe\<name>` where `<name>` should identify the virtual machine but may be freely chosen.
     - On a Mac, Linux, or Oracle Solaris host, a local domain socket is used instead. The socket filename must be chosen such that the user running Oracle VM VirtualBox has sufficient privileges to create and write to it. The `/tmp` directory is often a good candidate.

       On Linux there are various tools which can connect to a local domain socket or create one in server mode. The most flexible tool is `socat` and is available as part of many distributions.

     In this case, you can configure whether Oracle VM VirtualBox should create the named pipe, or the local domain socket non-Windows hosts, itself or whether Oracle VM VirtualBox should assume that the pipe or socket exists already. With the `VBoxManage` command-line options, this is referred to as server mode or client mode, respectively.

     For a direct connection between two virtual machines, corresponding to a null-modem cable, simply configure one VM to create a pipe or socket and another to attach to it.
   - **Raw File:** Send the virtual serial port output to a file. This option is very useful for capturing diagnostic output from a guest. Any file may be used for this purpose, as long as the user running Oracle VM VirtualBox has sufficient privileges to create and write to the file.
   - **TCP Socket:** Useful for forwarding serial traffic over TCP/IP, acting as a server, or it can act as a TCP client connecting to other servers. This option enables a remote machine to directly connect to the guest's serial port using TCP.
     - **TCP Server:** Deselect the **Connect to Existing Pipe/Socket** check box and specify the port number in the **Path/Address** field. This is typically 23 or 2023. Note that on UNIX-like systems you will have to use a port a number greater than 1024 for regular users.

       The client can use software such as `PuTTY` or the `telnet` command line tool to access the TCP Server.

  - **TCP Client:** To create a virtual null-modem cable over the Internet or LAN, the other side can connect using TCP by specifying `hostname:port` in the **Path/Address** field. The TCP socket will act in client mode if you select the **Connect to Existing Pipe/Socket** check box.

Up to four serial ports can be configured per virtual machine, but you can pick any port numbers out of the above. However, serial ports cannot reliably share interrupts. If both ports are to be used at the same time, they must use different interrupt levels, for example COM1 and COM2, but not COM1 and COM3.

## 3.11 USB Support

### 3.11.1 USB Settings

The **USB** section in a virtual machine's **Settings** window enables you to configure Oracle VM VirtualBox's sophisticated USB support.

Oracle VM VirtualBox can enable virtual machines to access the USB devices on your host directly. To achieve this, Oracle VM VirtualBox presents the guest OS with a virtual USB controller. As soon as the guest system starts using a USB device, it will appear as unavailable on the host.

**Note:**

- Be careful with USB devices that are currently in use on the host. For example, if you allow your guest to connect to your USB hard disk that is currently mounted on the host, when the guest is activated, it will be disconnected from the host without a proper shutdown. This may cause data loss.
- Oracle Solaris hosts have a few known limitations regarding USB support. See chapter 14, *Known Limitations*, page 301.

In addition to allowing a guest access to your local USB devices, Oracle VM VirtualBox even enables your guests to connect to remote USB devices by use of the VirtualBox Remote Desktop Extension (VRDE). See chapter 7.1.4, *Remote USB*, page 114.

To enable USB for a VM, select the **Enable USB Controller** check box. The following settings are available:

- **USB Controller:** Selects a controller with the specified level of USB support, as follows:
  - OHCI for USB 1.1
  - EHCI for USB 2.0. This also enables OHCI.
  - xHCI for USB 3.0. This supports all USB speeds.

**Note:** The xHCI and EHCI controllers are shipped as an Oracle VM VirtualBox extension package, which must be installed separately. See chapter 1.6, *Installing Oracle VM VirtualBox and Extension Packs*, page 6.

- **USB Device Filters:** When USB support is enabled for a VM, you can determine in detail which devices will be automatically attached to the guest. For this, you can create filters by specifying certain properties of the USB device. USB devices with a matching filter will be automatically passed to the guest once they are attached to the host. USB devices without a matching filter can be passed manually to the guest, for example by using the **Devices**, **USB** menu.

Clicking on the + button to the right of the **USB Device Filters** window creates a new filter. You can give the filter a name, for later reference, and specify the filter criteria. The more criteria you specify, the more precisely devices will be selected. For instance, if you specify only a vendor ID of 046d, all devices produced by Logitech will be available to the guest. If you fill in all fields, on the other hand, the filter will only apply to a particular device model from a particular vendor, and not even to other devices of the same type with a different revision and serial number.

In detail, the following criteria are available:

- **Vendor and Product ID.** With USB, each vendor of USB products carries an identification number that is unique world-wide, called the *vendor ID*. Similarly, each line of products is assigned a *product ID* number. Both numbers are commonly written in hexadecimal, and a colon separates the vendor from the product ID. For example, `046d:c016` stands for Logitech as a vendor, and the M-UV69a Optical Wheel Mouse product.

  Alternatively, you can also specify **Manufacturer** and **Product** by name.

  To list all the USB devices that are connected to your host machine with their respective vendor IDs and product IDs, use the following command:

  ```
  VBoxManage list usbhost
  ```

  On Windows, you can also see all USB devices that are attached to your system in the Device Manager. On Linux, you can use the `lsusb` command.
- **Serial Number.** While vendor ID and product ID are quite specific to identify USB devices, if you have two identical devices of the same brand and product line, you will also need their serial numbers to filter them out correctly.
- **Remote.** This setting specifies whether the device will be local only, remote only, such as over VRDP, or either.

On a Windows host, you will need to unplug and reconnect a USB device to use it after creating a filter for it.

As an example, you could create a new USB filter and specify a vendor ID of 046d for Logitech, Inc, a manufacturer index of 1, and "not remote". Then any USB devices on the host system produced by Logitech, Inc with a manufacturer index of 1 will be visible to the guest system.

Several filters can select a single device. For example, a filter which selects all Logitech devices, and one which selects a particular webcam.

You can deactivate filters without deleting them by deselecting the check box next to the filter name.

### 3.11.2 Implementation Notes for Windows and Linux Hosts

On Windows hosts, a kernel mode device driver provides USB proxy support. It implements both a USB monitor, which enables Oracle VM VirtualBox to capture devices when they are plugged in, and a USB device driver to claim USB devices for a particular virtual machine. As opposed to Oracle VM VirtualBox versions before 1.4.0, system reboots are no longer necessary after installing the driver. Also, you no longer need to replug devices for Oracle VM VirtualBox to claim them.

On newer Linux hosts, Oracle VM VirtualBox accesses USB devices through special files in the file system. When Oracle VM VirtualBox is installed, these are made available to all users in the `vboxusers` system group. In order to be able to access USB from guest systems, make sure that you are a member of this group.

On older Linux hosts, USB devices are accessed using the `usbfs` file system. Therefore, the user executing Oracle VM VirtualBox needs read and write permission to the USB file system. Most

distributions provide a group, such as `usbusers`, which the Oracle VM VirtualBox user needs to be added to. Also, Oracle VM VirtualBox can only proxy to virtual machines USB devices which are not claimed by a Linux host USB driver. The `Driver=` entry in `/proc/bus/usb/devices` will show you which devices are currently claimed. See also chapter 12.8.7, *USB Not Working*, page 293 for details about `usbfs`.

## 3.12 Shared Folders

Shared folders enable you to easily exchange data between a virtual machine and your host. This feature requires that the Oracle VM VirtualBox Guest Additions be installed in a virtual machine and is described in detail in chapter 4.3, *Shared Folders*, page 69.

## 3.13 User Interface

The **User Interface** section enables you to change certain aspects of the user interface of this VM.

- **Menu Bar:** This widget enables you to disable menus by clicking on the menu to release it, menu entries by deselecting the check box of the entry to disable it and the complete menu bar by deselecting the rightmost check box.
- **Mini ToolBar:** In full screen or seamless mode, Oracle VM VirtualBox can display a small toolbar that contains some of the items that are normally available from the virtual machine's menu bar. This toolbar reduces itself to a small gray line unless you move the mouse over it. With the toolbar, you can return from full screen or seamless mode, control machine execution or enable certain devices. If you do not want to see the toolbar, disable this setting.

  The second setting enables you to show the toolbar at the top of the screen, instead of showing it at the bottom.
- **Status Bar:** This widget enables you to disable icons on the status bar by deselecting the check box of an icon to disable it, to rearrange icons by dragging and dropping the icon, and to disable the complete status bar by deselecting the leftmost check box.

## 3.14 Alternative Firmware (EFI)

Oracle VM VirtualBox includes experimental support for the Extensible Firmware Interface (EFI), which is a new industry standard intended to eventually replace the legacy BIOS as the primary interface for bootstrapping computers and certain system services later.

By default, Oracle VM VirtualBox uses the BIOS firmware for virtual machines. To use EFI for a given virtual machine, you can enable EFI in the machine's **Settings** dialog. See chapter 3.5.1, *Motherboard Tab*, page 47. Alternatively, use the `VBoxManage` command line interface as follows:

```
VBoxManage modifyvm "VM name" --firmware efi
```

To switch back to using the BIOS:

```
VBoxManage modifyvm "VM name" --firmware bios
```

One notable user of EFI is Apple Mac OS X. More recent Linux versions and Windows releases, starting with Vista, also offer special versions that can be booted using EFI.

Another possible use of EFI in Oracle VM VirtualBox is development and testing of EFI applications, without booting any OS.

Note that the Oracle VM VirtualBox EFI support is experimental and will be enhanced as EFI matures and becomes more widespread. Mac OS X, Linux, and newer Windows guests are known to work fine. Windows 7 guests are unable to boot with the Oracle VM VirtualBox EFI implementation.

### 3.14.1 Video Modes in EFI

EFI provides two distinct video interfaces: GOP (Graphics Output Protocol) and UGA (Universal Graphics Adapter). Modern OSes, such as Mac OS X, generally use GOP, while some older ones still use UGA. Oracle VM VirtualBox provides a configuration option to control the graphics resolution for both interfaces, making the difference mostly irrelevant for users.

The default resolution is 1024x768. To select a graphics resolution for EFI, use the following `VBoxManage` command:

```
VBoxManage setextradata "VM name" VBoxInternal2/EfiGraphicsResolution HxV
```

Determine the horizontal resolution H and the vertical resolution V from the following list of default resolutions:

**VGA**

640x480, 32bpp, 4:3

**SVGA**

800x600, 32bpp, 4:3

**XGA**

1024x768, 32bpp, 4:3

**XGA+**

1152x864, 32bpp, 4:3

**HD**

1280x720, 32bpp, 16:9

**WXGA**

1280x800, 32bpp, 16:10

**SXGA**

1280x1024, 32bpp, 5:4

**SXGA+**

1400x1050, 32bpp, 4:3

**WXGA+**

1440x900, 32bpp, 16:10

**HD+**

1600x900, 32bpp, 16:9

**UXGA**

1600x1200, 32bpp, 4:3

**WSXGA+**

1680x1050, 32bpp, 16:10

**Full HD**

1920x1080, 32bpp, 16:9

**WUXGA**

1920x1200, 32bpp, 16:10

**DCI 2K**

2048x1080, 32bpp, 19:10

**Full HD+**

2160x1440, 32bpp, 3:2

**Unnamed**

2304x1440, 32bpp, 16:10

**QHD**

2560x1440, 32bpp, 16:9

**WQXGA**

2560x1600, 32bpp, 16:10

**QWXGA+**

2880x1800, 32bpp, 16:10

**QHD+**

3200x1800, 32bpp, 16:9

**WQSXGA**

3200x2048, 32bpp, 16:10

**4K UHD**

3840x2160, 32bpp, 16:9

**WQUXGA**

3840x2400, 32bpp, 16:10

**DCI 4K**

4096x2160, 32bpp, 19:10

**HXGA**

4096x3072, 32bpp, 4:3

**UHD+**

5120x2880, 32bpp, 16:9

**WHXGA**

5120x3200, 32bpp, 16:10

**WHSXGA**

6400x4096, 32bpp, 16:10

**HUXGA**

6400x4800, 32bpp, 4:3

**8K UHD2**

7680x4320, 32bpp, 16:9

If this list of default resolution does not cover your needs, see chapter 9.7.1, *Custom VESA Resolutions*, page 216. Note that the color depth value specified in a custom video mode must be specified. Color depths of 8, 16, 24, and 32 are accepted. EFI assumes a color depth of 32 by default.

The EFI default video resolution settings can only be changed when the VM is powered off.

### 3.14.2 Specifying Boot Arguments

It is currently not possible to manipulate EFI variables from within a running guest. For example, setting the "boot-args" variable by running the `nvram` tool in a Mac OS X guest will not work. As an alternative way, "VBoxInternal2/EfiBootArgs" extradata can be passed to a VM in order to set the "boot-args" variable. To change the "boot-args" EFI variable, use the following command:

```
VBoxManage setextradata "VM name" VBoxInternal2/EfiBootArgs <value>
```

# 4 Guest Additions

The previous chapter covered getting started with Oracle VM VirtualBox and installing operating systems in a virtual machine. For any serious and interactive use, the Oracle VM VirtualBox Guest Additions will make your life much easier by providing closer integration between host and guest and improving the interactive performance of guest systems. This chapter describes the Guest Additions in detail.

## 4.1 Introduction to Guest Additions

As mentioned in chapter 1.2, *Some Terminology*, page 2, the Guest Additions are designed to be installed *inside* a virtual machine after the guest operating system has been installed. They consist of device drivers and system applications that optimize the guest operating system for better performance and usability. See chapter 3.1, *Supported Guest Operating Systems*, page 40 for details on what guest operating systems are fully supported with Guest Additions by Oracle VM VirtualBox.

The Oracle VM VirtualBox Guest Additions for all supported guest operating systems are provided as a single CD-ROM image file which is called `VBoxGuestAdditions.iso`. This image file is located in the installation directory of Oracle VM VirtualBox. To install the Guest Additions for a particular VM, you mount this ISO file in your VM as a virtual CD-ROM and install from there.

The Guest Additions offer the following features:

- **Mouse pointer integration**. To overcome the limitations for mouse support described in chapter 1.9.2, *Capturing and Releasing Keyboard and Mouse*, page 12, this feature provides you with seamless mouse support. You will only have one mouse pointer and pressing the Host key is no longer required to "free" the mouse from being captured by the guest OS. To make this work, a special mouse driver is installed in the guest that communicates with the "real" mouse driver on your host and moves the guest mouse pointer accordingly.

- **Shared folders.** These provide an easy way to exchange files between the host and the guest. Much like ordinary Windows network shares, you can tell Oracle VM VirtualBox to treat a certain host directory as a shared folder, and Oracle VM VirtualBox will make it available to the guest operating system as a network share, irrespective of whether guest actually has a network. See chapter 4.3, *Shared Folders*, page 69.

- **Better video support.** While the virtual graphics card which Oracle VM VirtualBox emulates for any guest operating system provides all the basic features, the custom video drivers that are installed with the Guest Additions provide you with extra high and non-standard video modes, as well as accelerated video performance.

  In addition, with Windows, Linux, and Oracle Solaris guests, you can resize the virtual machine's window if the Guest Additions are installed. The video resolution in the guest will be automatically adjusted, as if you had manually entered an arbitrary resolution in the guest's **Display** settings. See chapter 1.9.5, *Resizing the Machine's Window*, page 14.

  If the Guest Additions are installed, 3D graphics and 2D video for guest applications can be accelerated. See chapter 4.5, *Hardware-Accelerated Graphics*, page 74.

- **Seamless windows.** With this feature, the individual windows that are displayed on the desktop of the virtual machine can be mapped on the host's desktop, as if the underlying application was actually running on the host. See chapter 4.6, *Seamless Windows*, page 75.

- **Generic host/guest communication channels.** The Guest Additions enable you to control and monitor guest execution. The "guest properties" provide a generic string-based mechanism to exchange data bits between a guest and a host, some of which have special meanings for controlling and monitoring the guest. See chapter 4.7, *Guest Properties*, page 76.

  Additionally, applications can be started in a guest from the host. See chapter 4.9, *Guest Control of Applications*, page 79.

- **Time synchronization.** With the Guest Additions installed, Oracle VM VirtualBox can ensure that the guest's system time is better synchronized with that of the host.

  For various reasons, the time in the guest might run at a slightly different rate than the time on the host. The host could be receiving updates through NTP and its own time might not run linearly. A VM could also be paused, which stops the flow of time in the guest for a shorter or longer period of time. When the wall clock time between the guest and host only differs slightly, the time synchronization service attempts to gradually and smoothly adjust the guest time in small increments to either "catch up" or "lose" time. When the difference is too great, for example if a VM paused for hours or restored from saved state, the guest time is changed immediately, without a gradual adjustment.

  The Guest Additions will resynchronize the time regularly. See chapter 9.13.3, *Tuning the Guest Additions Time Synchronization Parameters*, page 226 for how to configure the parameters of the time synchronization mechanism.

- **Shared clipboard.** With the Guest Additions installed, the clipboard of the guest operating system can optionally be shared with your host operating system. See chapter 3.4, *General Settings*, page 45.

- **Automated logins.** Also called credentials passing. See chapter 9.1, *Automated Guest Logins*, page 206.

Each version of Oracle VM VirtualBox, even minor releases, ship with their own version of the Guest Additions. While the interfaces through which the Oracle VM VirtualBox core communicates with the Guest Additions are kept stable so that Guest Additions already installed in a VM should continue to work when Oracle VM VirtualBox is upgraded on the host, for best results, it is recommended to keep the Guest Additions at the same version.

The Windows and Linux Guest Additions therefore check automatically whether they have to be updated. If the host is running a newer Oracle VM VirtualBox version than the Guest Additions, a notification with further instructions is displayed in the guest.

To disable this update check for the Guest Additions of a given virtual machine, set the value of its `/VirtualBox/GuestAdd/CheckHostVersion` guest property to `0`. See chapter 4.7, *Guest Properties*, page 76.

## 4.2 Installing and Maintaining Guest Additions

Guest Additions are available for virtual machines running Windows, Linux, Oracle Solaris, or OS/2. The following sections describe the specifics of each variant in detail.

### 4.2.1 Guest Additions for Windows

The Oracle VM VirtualBox Windows Guest Additions are designed to be installed in a virtual machine running a Windows operating system. The following versions of Windows guests are supported:

- Microsoft Windows NT 4.0 (any service pack)

- Microsoft Windows 2000 (any service pack)
- Microsoft Windows XP (any service pack)
- Microsoft Windows Server 2003 (any service pack)
- Microsoft Windows Server 2008
- Microsoft Windows Vista (all editions)
- Microsoft Windows 7 (all editions)
- Microsoft Windows 8 (all editions)
- Microsoft Windows 10 RTM build 10240
- Microsoft Windows Server 2012

#### 4.2.1.1 Installing the Windows Guest Additions

In the **Devices** menu in the virtual machine's menu bar, Oracle VM VirtualBox has a menu item **Insert Guest Additions CD Image**, which mounts the Guest Additions ISO file inside your virtual machine. A Windows guest should then automatically start the Guest Additions installer, which installs the Guest Additions on your Windows guest.

For other guest operating systems, or if automatic start of software on a CD is disabled, you need to do a manual start of the installer.

> **Note:** For the basic Direct3D acceleration to work in a Windows guest, you have to install the WDDM video driver available for Windows Vista or later.
> For Windows 8 and later, only the WDDM Direct3D video driver is available. For basic Direct3D acceleration to work in Windows XP guests, you have to install the Guest Additions in Safe Mode. See chapter 14, *Known Limitations*, page 301 for details.

If you prefer to mount the Guest Additions manually, you can perform the following steps:

1. Start the virtual machine in which you have installed Windows.
2. Select **Mount CD/DVD-ROM** from the **Devices** menu in the virtual machine's menu bar and then **CD/DVD-ROM Image**. This displays the Virtual Media Manager, described in chapter 5.3, *The Virtual Media Manager*, page 87.
3. In the Virtual Media Manager, click **Add** and browse your host file system for the `VBoxGuestAdditions.iso` file.
   - On a Windows host, this file is in the Oracle VM VirtualBox installation directory, usually in `C:\Program files\Oracle\VirtualBox`.
   - On Mac OS X hosts, this file is in the application bundle of Oracle VM VirtualBox. Right-click on the Oracle VM VirtualBox icon in Finder and choose **Show Package Contents**. The file is located in the `Contents/MacOS` folder.
   - On a Linux host, this file is in the `additions` folder where you installed Oracle VM VirtualBox, usually `/opt/VirtualBox/`.
   - On Oracle Solaris hosts, this file is in the `additions` folder where you installed Oracle VM VirtualBox, usually `/opt/VirtualBox`.
4. In the Virtual Media Manager, select the ISO file and click **Select** button. This mounts the ISO file and presents it to your Windows guest as a CD-ROM.

Unless you have the Autostart feature disabled in your Windows guest, Windows will now autostart the Oracle VM VirtualBox Guest Additions installation program from the Additions ISO. If the Autostart feature has been turned off, choose `VBoxWindowsAdditions.exe` from the CD/DVD drive inside the guest to start the installer.

The installer will add several device drivers to the Windows driver database and then invoke the hardware detection wizard.

Depending on your configuration, it might display warnings that the drivers are not digitally signed. You must confirm these in order to continue the installation and properly install the Additions.

After installation, reboot your guest operating system to activate the Additions.

#### 4.2.1.2 Updating the Windows Guest Additions

Windows Guest Additions can be updated by running the installation program again. This replaces the previous Additions drivers with updated versions.

Alternatively, you can also open the Windows Device Manager and select **Update Driver...** for the following devices:

1. Oracle VM VirtualBox Graphics Adapter
2. Oracle VM VirtualBox System Device

For each, choose the option to provide your own driver, click **Have Disk** and navigate to the CD-ROM drive with the Guest Additions.

#### 4.2.1.3 Unattended Installation

To avoid popups when performing an unattended installation of the Oracle VM VirtualBox Guest Additions, the code signing certificates used to sign the drivers needs to be installed in the correct certificate stores on the guest operating system. Failure to do this will cause a typical Windows installation to display multiple dialogs asking whether you want to install a particular driver.

> **Note:** On some Windows versions, such as Windows 2000 and Windows XP, the user intervention popups mentioned above are always displayed, even after importing the Oracle certificates.

Installing the code signing certificates on a Windows guest can be done automatically. Use the `VBoxCertUtil.exe` utility from the `cert` folder on the Guest Additions installation CD.

Use the following steps:

1. Log in as Administrator on the guest.
2. Mount the Oracle VM VirtualBox Guest Additions .ISO.
3. Open a command line window on the guest and change to the `cert` folder on the Oracle VM VirtualBox Guest Additions CD.
4. Run the following command:

   ```
   VBoxCertUtil.exe add-trusted-publisher vbox*.cer --root vbox*.cer
   ```

   This command installs the certificates to the certificate store. When installing the same certificate more than once, an appropriate error will be displayed.

To allow for completely unattended guest installations, you can specify a command line parameter to the install launcher:

```
VBoxWindowsAdditions.exe /S
```

This automatically installs the right files and drivers for the corresponding platform, either 32-bit or 64-bit.

**Note:** By default on an unattended installation on a Vista or Windows 7 guest, there will be the XPDM graphics driver installed. This graphics driver does not support Windows Aero / Direct3D on the guest. Instead, the WDDM graphics driver needs to be installed. To select this driver by default, add the command line parameter `/with_wddm` when invoking the Windows Guest Additions installer. This is only required for Vista and Windows 7.

**Note:** For Windows Aero to run correctly on a guest, the guest's VRAM size needs to be configured to at least 128 MB.

For more options regarding unattended guest installations, consult the command line help by using the command:

```
VBoxWindowsAdditions.exe /?
```

#### 4.2.1.4 Manual File Extraction

If you would like to install the files and drivers manually, you can extract the files from the Windows Guest Additions setup as follows:

```
VBoxWindowsAdditions.exe /extract
```

To explicitly extract the Windows Guest Additions for another platform than the current running one, such as 64-bit files on a 32-bit system, you must use the appropriate platform installer. Use `VBoxWindowsAdditions-x86.exe` or `VBoxWindowsAdditions-amd64.exe` with the `/extract` parameter.

### 4.2.2 Guest Additions for Linux

Like the Windows Guest Additions, the Oracle VM VirtualBox Guest Additions for Linux are a set of device drivers and system applications which may be installed in the guest operating system.

The following Linux distributions are officially supported:

- Oracle Linux as of version 5, including UEK kernels
- Fedora as of Fedora Core 4
- Redhat Enterprise Linux as of version 3
- SUSE and openSUSE Linux as of version 9
- Ubuntu as of version 5.10

Many other distributions are known to work with the Guest Additions.

The version of the Linux kernel supplied by default in SUSE and openSUSE 10.2, Ubuntu 6.10 (all versions) and Ubuntu 6.06 (server edition) contains a bug which can cause it to crash during startup when it is run in a virtual machine. The Guest Additions work in those distributions.

Note that some Linux distributions already come with all or part of the Oracle VM VirtualBox Guest Additions. You may choose to keep the distribution's version of the Guest Additions but

these are often not up to date and limited in functionality, so we recommend replacing them with the Guest Additions that come with Oracle VM VirtualBox. The Oracle VM VirtualBox Linux Guest Additions installer tries to detect an existing installation and replace them but depending on how the distribution integrates the Guest Additions, this may require some manual interaction. It is highly recommended to take a snapshot of the virtual machine before replacing preinstalled Guest Additions.

#### 4.2.2.1 Installing the Linux Guest Additions

The Oracle VM VirtualBox Guest Additions for Linux are provided on the same virtual CD-ROM file as the Guest Additions for Windows. See chapter 4.2.1.1, *Installing the Windows Guest Additions*, page 64. They also come with an installation program that guides you through the setup process. However, due to the significant differences between Linux distributions, installation may be slightly more complex when compared to Windows.

Installation generally involves the following steps:

1. Before installing the Guest Additions, you prepare your guest system for building external kernel modules. This works as described in chapter 2.3.2, *The Oracle VM VirtualBox Driver Modules*, page 33, except that this step must be performed in your Linux *guest* instead of on a Linux host system.

   If you suspect that something has gone wrong, check that your guest is set up correctly and run the following command as root:

   ```
   rcvboxadd setup
   ```

2. Insert the `VBoxGuestAdditions.iso` CD file into your Linux guest's virtual CD-ROM drive, as described for a Windows guest in chapter 4.2.1.1, *Installing the Windows Guest Additions*, page 64.

3. Change to the directory where your CD-ROM drive is mounted and run the following command as root:

   ```
   sh ./VBoxLinuxAdditions.run
   ```

#### 4.2.2.2 Graphics and Mouse Integration

In Linux and Oracle Solaris guests, Oracle VM VirtualBox graphics and mouse integration goes through the X Window System. Oracle VM VirtualBox can use the X.Org variant of the system, or XFree86 version 4.3 which is identical to the first X.Org release. During the installation process, the X.Org display server will be set up to use the graphics and mouse drivers which come with the Guest Additions.

After installing the Guest Additions into a fresh installation of a supported Linux distribution or Oracle Solaris system, many unsupported systems will work correctly too, the guest's graphics mode will change to fit the size of the Oracle VM VirtualBox window on the host when it is resized. You can also ask the guest system to switch to a particular resolution by sending a video mode hint using the `VBoxManage` tool.

Multiple guest monitors are supported in guests using the X.Org server version 1.3, which is part of release 7.3 of the X Window System version 11, or a later version. The layout of the guest screens can be adjusted as needed using the tools which come with the guest operating system.

If you want to understand more about the details of how the X.Org drivers are set up, in particular if you wish to use them in a setting which our installer does not handle correctly, see chapter 9.3.2, *Guest Graphics and Mouse Driver Setup in Depth*, page 211.

#### 4.2.2.3 Updating the Linux Guest Additions

The Guest Additions can simply be updated by going through the installation procedure again with an updated CD-ROM image. This will replace the drivers with updated versions. You should reboot after updating the Guest Additions.

#### 4.2.2.4 Uninstalling the Linux Guest Additions

If you have a version of the Guest Additions installed on your virtual machine and wish to remove it without installing new ones, you can do so by inserting the Guest Additions CD image into the virtual CD-ROM drive as described above. Then run the installer for the current Guest Additions with the `uninstall` parameter from the path that the CD image is mounted on in the guest, as follows:

```
sh ./VBoxLinuxAdditions.run uninstall
```

While this will normally work without issues, you may need to do some manual cleanup of the guest in some cases, especially of the XFree86Config or xorg.conf file. In particular, if the Additions version installed or the guest operating system were very old, or if you made your own changes to the Guest Additions setup after you installed them.

You can uninstall the Additions as follows:

```
/opt/VBoxGuestAdditions-<version>/uninstall.sh
```

Replace `/opt/VBoxGuestAdditions-`*`version`* with the correct Guest Additions installation directory.

### 4.2.3 Guest Additions for Oracle Solaris

Like the Windows Guest Additions, the Oracle VM VirtualBox Guest Additions for Oracle Solaris take the form of a set of device drivers and system applications which may be installed in the guest operating system.

The following Oracle Solaris distributions are officially supported:

- Oracle Solaris 11, including Oracle Solaris 11 Express
- Oracle Solaris 10 4/08 and later

Other distributions may work if they are based on comparable software releases.

#### 4.2.3.1 Installing the Oracle Solaris Guest Additions

The Oracle VM VirtualBox Guest Additions for Oracle Solaris are provided on the same ISO CD-ROM as the Additions for Windows and Linux. They come with an installation program that guides you through the setup process.

Installation involves the following steps:

1. Mount the `VBoxGuestAdditions.iso` file as your Oracle Solaris guest's virtual CD-ROM drive, exactly the same way as described for a Windows guest in chapter 4.2.1.1, *Installing the Windows Guest Additions*, page 64.

   If the CD-ROM drive on the guest does not get mounted, as seen with some versions of Oracle Solaris 10, run the following command as root:

   ```
   svcadm restart volfs
   ```

2. Change to the directory where your CD-ROM drive is mounted and run the following command as root:

```
pkgadd -G -d ./VBoxSolarisAdditions.pkg
```

3. Choose **1** and confirm installation of the Guest Additions package. After the installation is complete, log out and log in to X server on your guest, to activate the X11 Guest Additions.

#### 4.2.3.2 Uninstalling the Oracle Solaris Guest Additions

The Oracle Solaris Guest Additions can be safely removed by removing the package from the guest. Open a root terminal session and run the following command:

```
pkgrm SUNWvboxguest
```

#### 4.2.3.3 Updating the Oracle Solaris Guest Additions

The Guest Additions should be updated by first uninstalling the existing Guest Additions and then installing the new ones. Attempting to install new Guest Additions without removing the existing ones is not possible.

### 4.2.4 Guest Additions for OS/2

Oracle VM VirtualBox also ships with a set of drivers that improve running OS/2 in a virtual machine. Due to restrictions of OS/2 itself, this variant of the Guest Additions has a limited feature set. See chapter 14, *Known Limitations*, page 301 for details.

The OS/2 Guest Additions are provided on the same ISO CD-ROM as those for the other platforms. Mount the ISO in OS/2 as described previously. The OS/2 Guest Additions are located in the directory `\OS2`.

We do not provide an automatic installer at this time. See the `readme.txt` file in the CD-ROM directory, which describes how to install the OS/2 Guest Additions manually.

## 4.3 Shared Folders

With the *shared folders* feature of Oracle VM VirtualBox, you can access files of your host system from within the guest system. This is similar to how you would use network shares in Windows networks, except that shared folders do not require networking, only the Guest Additions. Shared folders are supported with Windows 2000 or later, Linux, and Oracle Solaris guests. Oracle VM VirtualBox release 6.0 includes experimental support for Mac OS X and OS/2 guests.

Shared folders physically reside on the *host* and are then shared with the guest, which uses a special file system driver in the Guest Additions to talk to the host. For Windows guests, shared folders are implemented as a pseudo-network redirector. For Linux and Oracle Solaris guests, the Guest Additions provide a virtual file system.

To share a host folder with a virtual machine in Oracle VM VirtualBox, you must specify the path of the folder and choose a *share name* that the guest can use to access the shared folder. This happens on the host. In the guest you can then use the share name to connect to it and access files.

There are several ways in which shared folders can be set up for a virtual machine:

- In the window of a running VM, you select **Shared Folders** from the **Devices** menu, or click on the folder icon on the status bar in the bottom right corner.
- If a VM is not currently running, you can configure shared folders in the virtual machine's **Settings** dialog.
- From the command line, you can create shared folders using `VBoxManage`, as follows:

```
VBoxManage sharedfolder add "VM name" --name "sharename" --hostpath "C:\test"
```

See chapter 8.33, *VBoxManage sharedfolder add/remove*, page 175.

There are two types of shares:

- Permanent shares, that are saved with the VM settings.
- Transient shares, that are added at runtime and disappear when the VM is powered off. These can be created using a checkbox in the VirtualBox Manager, or by using the `--transient` option of the `VBoxManage sharedfolder add` command.

Shared folders can either be read-write or read-only. This means that the guest is either allowed to both read and write, or just read files on the host. By default, shared folders are read-write. Read-only folders can be created using a checkbox in the VirtualBox Manager, or with the `--readonly` option of the `VBoxManage sharedfolder add` command.

Oracle VM VirtualBox shared folders also support symbolic links, also called *symlinks*, under the following conditions:

- The host operating system must support symlinks. For example, a Mac OS X, Linux, or Oracle Solaris host is required.
- Currently only Linux and Oracle Solaris Guest Additions support symlinks.
- For security reasons the guest OS is not allowed to create symlinks by default. If you trust the guest OS to not abuse the functionality, you can enable creation of symlinks for a shared folder as follows:

```
VBoxManage setextradata "VM name" VBoxInternal2/SharedFoldersEnableSymlinksCreate/<sharename> 1
```

### 4.3.1 Manual Mounting

You can mount the shared folder from inside a VM, in the same way as you would mount an ordinary network share:

- In a Windows guest, shared folders are browseable and therefore visible in Windows Explorer. To attach the host's shared folder to your Windows guest, open Windows Explorer and look for the folder in **My Networking Place**s, **Entire Network**, **Oracle VM VirtualBox Shared Folders**. By right-clicking on a shared folder and selecting **Map Network Drive** from the menu that pops up, you can assign a drive letter to that shared folder.

  Alternatively, on the Windows command line, use the following command:

  ```
  net use x: \\vboxsvr\sharename
  ```

  While `vboxsvr` is a fixed name, note that `vboxsrv` would also work, replace *`x:`* with the drive letter that you want to use for the share, and *`sharename`* with the share name specified with `VBoxManage`.

- In a Linux guest, use the following command:

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

  To mount a shared folder during boot, add the following entry to `/etc/fstab`:

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

- In a Oracle Solaris guest, use the following command:

```
mount -F vboxfs [-o OPTIONS] sharename mountpoint
```

Replace *sharename*, use a lowercase string, with the share name specified with `VBoxManage` or the GUI. Replace *mountpoint* with the path where you want the share to be mounted on the guest, such as `/mnt/share`. The usual mount rules apply. For example, create this directory first if it does not exist yet.

Here is an example of mounting the shared folder for the user jack on Oracle Solaris:

```
$ id
uid=5000(jack) gid=1(other)
$ mkdir /export/home/jack/mount
$ pfexec mount -F vboxfs -o uid=5000,gid=1 jackshare /export/home/jack/mount
$ cd ~/mount
$ ls
sharedfile1.mp3 sharedfile2.txt
$
```

Beyond the standard options supplied by the `mount` command, the following are available:

```
iocharset CHARSET
```

This option sets the character set used for I/O operations. Note that on Linux guests, if the `iocharset` option is not specified, then the Guest Additions driver will attempt to use the character set specified by the CONFIG_NLS_DEFAULT kernel option. If this option is not set either, then UTF-8 is used.

```
convertcp CHARSET
```

This option specifies the character set used for the shared folder name. This is UTF-8 by default.

The generic mount options, documented in the `mount` manual page, apply also. Especially useful are the options `uid`, `gid` and `mode`, as they can allow access by normal users in read/write mode, depending on the settings, even if root has mounted the filesystem.

- In an OS/2 guest, use the `VBoxControl` command to manage shared folders. For example:

```
VBoxControl sharedfolder use D: MyShareName
VBoxControl sharedfolder unuse D:
VBoxControl sharedfolder list
```

As with Windows guests, shared folders can also be accessed via UNC using `\\VBoxSF\`, `\\VBoxSvr\` or `\\VBoxSrv\` as the server name and the shared folder name as *sharename*.

### 4.3.2 Automatic Mounting

Oracle VM VirtualBox provides the option to mount shared folders automatically. When automatic mounting is enabled for a shared folder, the Guest Additions service will mount it for you automatically. For Windows or OS/2, a preferred drive letter can also be specified. For Linux or Oracle Solaris, a mount point directory can also be specified.

If a drive letter or mount point is not specified, or is in use already, an alternative location is found by the Guest Additions service. The service searches for an alternative location depending on the guest OS, as follows:

- **Windows and OS/2 guests.** Search for a free drive letter, starting at `Z:`. If all drive letters are assigned, the folder is not mounted.

- **Linux and Oracle Solaris guests.** Folders are mounted under the `/media` directory. The folder name is normalized (no spaces, slashes or colons) and is prefixed with `sf_`.

  For example, if you have a shared folder called `myfiles`, it will appear as `/media/sf_myfiles` in the guest.

  The guest properties `/VirtualBox/GuestAdd/SharedFolders/MountDir` and the more generic `/VirtualBox/GuestAdd/SharedFolders/MountPrefix` can be used to override the automatic mount directory and prefix. See chapter 4.7, *Guest Properties*, page 76.

Access to an automatically mounted shared folder is granted to everyone in a Windows guest, including the guest user. For Linux and Oracle Solaris guests, access is restricted to members of the group `vboxsf` and the `root` user.

## 4.4 Drag and Drop

Oracle VM VirtualBox enables you to drag and drop content from the host to the guest, and vice versa. For this to work the latest Guest Additions must be installed on the guest.

Drag and drop transparently allows copying or opening files, directories, and even certain clipboard formats from one end to the other. For example, from the host to the guest or from the guest to the host. You then can perform drag and drop operations between the host and a VM, as it would be a native drag and drop operation on the host OS.

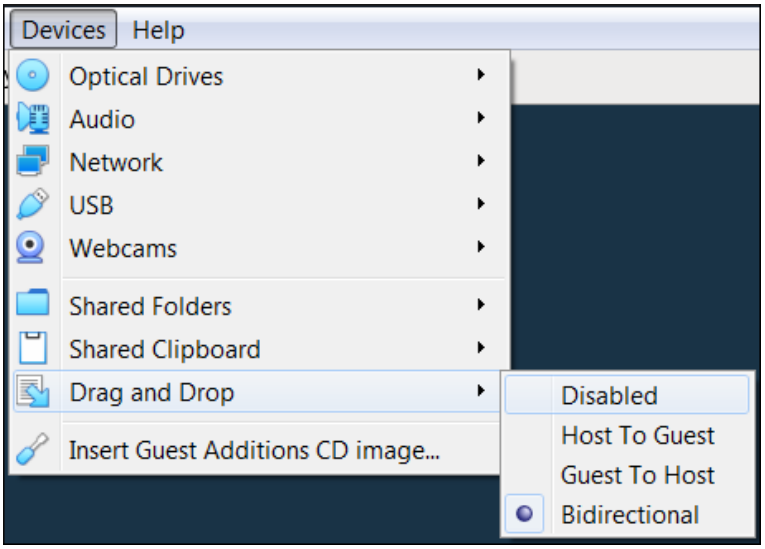
At the moment drag and drop is implemented for Windows-based and X-Windows-based systems, both on the host and guest side. As X-Windows supports many different drag and drop protocols only the most common one, XDND, is supported for now. Applications using other protocols, such as Motif or OffiX, will not be recognized by Oracle VM VirtualBox.

In the context of using drag and drop, the origin of the data is called the *source*. That is, where the actual data comes from and is specified. The *target* specifies where the data from the source should go to. Transferring data from the source to the target can be done in various ways, such as copying, moving, or linking.

> **Note:** At the moment only copying of data is supported. Moving or linking is not yet implemented.

When transferring data from the host to the guest OS, the host in this case is the source, whereas the guest OS is the target. However, when transferring data from the guest OS to the host, the guest OS this time became the source and the host is the target.

For security reasons drag and drop can be configured at runtime on a per-VM basis either using the **Drag and Drop** menu item in the **Devices** menu of the virtual machine, as shown below, or the `VBoxManage` command.




The following drag and drop modes are available:

- **Disabled.** Disables the drag and drop feature entirely. This is the default when creating a new VM.
- **Host To Guest.** Enables drag and drop operations from the host to the guest only.
- **Guest To Host.** Enables drag and drop operations from the guest to the host only.
- **Bidirectional.** Enables drag and drop operations in both directions: from the host to the guest, and from the guest to the host.

**Note:** Drag and drop support depends on the frontend being used. At the moment, only the VirtualBox Manager frontend provides this functionality.

To use the `VBoxManage` command to control the current drag and drop mode, see chapter 8, *VBoxManage*, page 120. The `modifyvm` and `controlvm` commands enable setting of a VM's current drag and drop mode from the command line.

### 4.4.1 Supported Formats

As Oracle VM VirtualBox can run on a variety of host operating systems and also supports a wide range of guests, certain data formats must be translated after transfer. This is so that the target operating system, which receiving the data, is able to handle them in an appropriate manner.

**Note:** When dragging files no data conversion is done in any way. For example, when transferring a file from a Linux guest to a Windows host the Linux-specific line endings are not converted to Windows line endings.

The following formats are handled by the Oracle VM VirtualBox drag and drop service:

- **Plain text:** From applications such as text editors, internet browsers and terminal windows.
- **Files:** From file managers such as Windows Explorer, Nautilus, and Finder.
- **Directories:** For directories, the same formats apply as for files.

### 4.4.2 Known Limitations

The following limitations are known for drag and drop:

On Windows hosts, dragging and dropping content between UAC-elevated (User Account Control) programs and non-UAC-elevated programs is not allowed. If you start Oracle VM VirtualBox with Administrator privileges then drag and drop will not work with Windows Explorer, which runs with regular user privileges by default.

## 4.5 Hardware-Accelerated Graphics

### 4.5.1 Hardware 3D Acceleration (OpenGL and Direct3D 8/9)

The Oracle VM VirtualBox Guest Additions contain experimental hardware 3D support for Windows, Linux, and Oracle Solaris guests.

With this feature, if an application inside your virtual machine uses 3D features through the OpenGL or Direct3D 8/9 programming interfaces, instead of emulating them in software, which would be slow, Oracle VM VirtualBox will attempt to use your host's 3D hardware. This works for all supported host platforms, provided that your host operating system can make use of your accelerated 3D hardware in the first place.

The 3D acceleration feature currently has the following preconditions:

- It is only available for certain Windows, Linux, and Oracle Solaris guests. In particular:
  - 3D acceleration with Windows guests requires Windows 2000, Windows XP, Vista, or Windows 7. Apart from on Windows 2000 guests, both OpenGL and Direct3D 8/9 are supported on an experimental basis.
  - OpenGL on Linux requires kernel 2.6.27 or later, as well as X.org server version 1.5 or later. Ubuntu 10.10 and Fedora 14 have been tested and confirmed as working.
  - OpenGL on Oracle Solaris guests requires X.org server version 1.5 or later.
- The Guest Additions must be installed.

> **Note:** For the basic Direct3D acceleration to work in a Windows Guest, Oracle VM VirtualBox needs to replace Windows system files in the virtual machine. As a result, the Guest Additions installation program offers Direct3D acceleration as an option that must be explicitly enabled. Also, you must install the Guest Additions in Safe Mode. This does *not* apply to the WDDM Direct3D video driver available for Windows Vista and later. See chapter 14, *Known Limitations*, page 301 for details.

- Because 3D support is still experimental at this time, it is disabled by default and must be *manually enabled* in the VM settings. See chapter 3.6, *Display Settings*, page 50.

> **Note:** Untrusted guest systems should not be allowed to use the 3D acceleration features of Oracle VM VirtualBox, just as untrusted host software should not be allowed to use 3D acceleration. Drivers for 3D hardware are generally too complex to be made properly secure and any software which is allowed to access them may be able to compromise the operating system running them. In addition, enabling 3D acceleration gives the guest direct access to a large body of additional program code in the Oracle VM VirtualBox host process which it might conceivably be able to use to crash the virtual machine.

To enable Aero theme support, the Oracle VM VirtualBox WDDM video driver must be installed, which is available with the Guest Additions installation. The WDDM driver is not installed by default for Vista and Windows 7 guest and must be *manually selected* in the Guest Additions installer by clicking **No** in the **Would You Like to Install Basic Direct3D Support** dialog displayed when the Direct3D feature is selected.

The Aero theme is not enabled by default. To enable it, do the following:

- **Windows Vista guests:** Right-click on the desktop and select **Personalize**, then select **Windows Color and Appearance** in the **Personalization** window. In the **Appearance Settings** dialog, select **Windows Aero** and click **OK**.
- **Windows 7 guests:** Right-click on the desktop and select **Personalize**. Select any Aero theme in the **Personalization** window.

Technically, Oracle VM VirtualBox implements this by installing an additional hardware 3D driver inside your guest when the Guest Additions are installed. This driver acts as a hardware 3D driver and reports to the guest operating system that the virtual hardware is capable of 3D hardware acceleration. When an application in the guest then requests hardware acceleration through the OpenGL or Direct3D programming interfaces, these are sent to the host through a special communication tunnel implemented by Oracle VM VirtualBox, and then the *host* performs the requested 3D operation using the host's programming interfaces.

### 4.5.2 Hardware 2D Video Acceleration for Windows Guests

The Oracle VM VirtualBox Guest Additions contain experimental hardware 2D video acceleration support for Windows guests.

With this feature, if an application such as a video player inside your Windows VM uses 2D video overlays to play a movie clip, then Oracle VM VirtualBox will attempt to use your host's video acceleration hardware instead of performing overlay stretching and color conversion in software, which would be slow. This currently works for Windows, Linux and Mac OS X host platforms, provided that your host operating system can make use of 2D video acceleration in the first place.

Hardware 2D video acceleration currently has the following preconditions:

- Only available for Windows guests, running Windows XP or later.
- Guest Additions must be installed.
- Because 2D support is still experimental at this time, it is disabled by default and must be *manually enabled* in the VM settings. See chapter 3.6, *Display Settings*, page 50.

Technically, Oracle VM VirtualBox implements this by exposing video overlay DirectDraw capabilities in the Guest Additions video driver. The driver sends all overlay commands to the host through a special communication tunnel implemented by Oracle VM VirtualBox. On the host side, OpenGL is then used to implement color space transformation and scaling

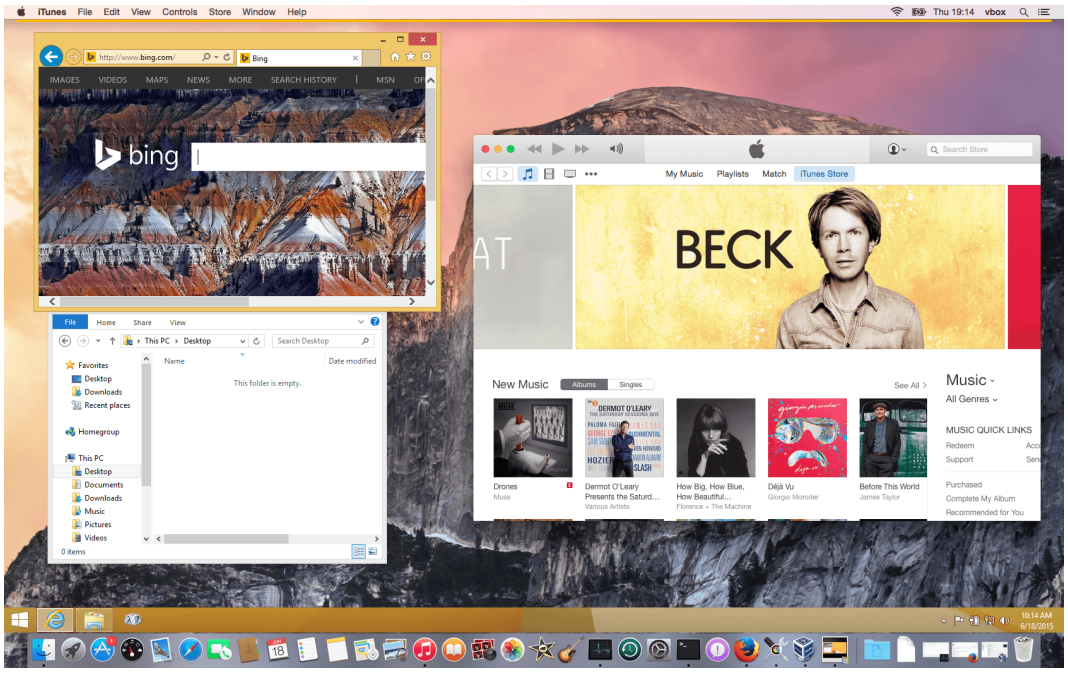
## 4.6 Seamless Windows

With the *seamless windows* feature of Oracle VM VirtualBox, you can have the windows that are displayed within a virtual machine appear side by side next to the windows of your host. This feature is supported for the following guest operating systems, provided that the Guest Additions are installed:

- Windows guests. Support was added in Oracle VM VirtualBox 1.5.

- Supported Linux or Oracle Solaris guests running the X Window System. Support was added with Oracle VM VirtualBox 1.6.

After seamless windows are enabled, Oracle VM VirtualBox suppresses the display of the desktop background of your guest, allowing you to run the windows of your guest operating system seamlessly next to the windows of your host.




To enable seamless mode, after starting the virtual machine, press the **Host key + L**. The Host key is normally the right control key. This will enlarge the size of the VM's display to the size of your host screen and mask out the guest operating system's background. To disable seamless windows and go back to the normal VM display, press the Host key + L again.

## 4.7 Guest Properties

Oracle VM VirtualBox enables requests of some properties from a running guest, provided that the Oracle VM VirtualBox Guest Additions are installed and the VM is running. This provides the following advantages:

- A number of predefined VM characteristics are automatically maintained by Oracle VM VirtualBox and can be retrieved on the host. For example, to monitor VM performance and statistics.

- Arbitrary string data can be exchanged between guest and host. This works in both directions.

To accomplish this, Oracle VM VirtualBox establishes a private communication channel between the Oracle VM VirtualBox Guest Additions and the host, and software on both sides can use this channel to exchange string data for arbitrary purposes. Guest properties are simply string keys to which a value is attached. They can be set, or written to, by either the host and the guest. They can also be read from both sides.

In addition to establishing the general mechanism of reading and writing values, a set of predefined guest properties is automatically maintained by the Oracle VM VirtualBox Guest Additions

to allow for retrieving interesting guest data such as the guest's exact operating system and service pack level, the installed version of the Guest Additions, users that are currently logged into the guest OS, network statistics and more. These predefined properties are all prefixed with `/VirtualBox/` and organized into a hierarchical tree of keys.

Some of this runtime information is shown when you select **Session Information Dialog** from a virtual machine's **Machine** menu.

A more flexible way to use this channel is with the `VBoxManage guestproperty` command. See chapter 8.34, *VBoxManage guestproperty*, page 176. For example, to have *all* the available guest properties for a given running VM listed with their respective values, use this command:

```
$ VBoxManage guestproperty enumerate "Windows Vista III"
VirtualBox Command Line Management Interface Version <version-number>
(C) 2005-2018 Oracle Corporation
All rights reserved.

Name: /VirtualBox/GuestInfo/OS/Product, value: Windows Vista Business Edition,
    timestamp: 1229098278843087000, flags:
Name: /VirtualBox/GuestInfo/OS/Release, value: 6.0.6001,
    timestamp: 1229098278950553000, flags:
Name: /VirtualBox/GuestInfo/OS/ServicePack, value: 1,
    timestamp: 1229098279122627000, flags:
Name: /VirtualBox/GuestAdd/InstallDir,
    value: C:/Program Files/Oracle/VirtualBox
    Guest Additions, timestamp: 1229098279269739000, flags:
Name: /VirtualBox/GuestAdd/Revision, value: 40720,
    timestamp: 1229098279345664000, flags:
Name: /VirtualBox/GuestAdd/Version, value: <version-number>,
    timestamp: 1229098279479515000, flags:
Name: /VirtualBox/GuestAdd/Components/VBoxControl.exe, value: <version-number>r40720,
    timestamp: 1229098279651731000, flags:
Name: /VirtualBox/GuestAdd/Components/VBoxHook.dll, value: <version-number>r40720,
    timestamp: 1229098279804835000, flags:
Name: /VirtualBox/GuestAdd/Components/VBoxDisp.dll, value: <version-number>r40720,
    timestamp: 1229098279880611000, flags:
Name: /VirtualBox/GuestAdd/Components/VBoxMRXNP.dll, value: <version-number>r40720,
    timestamp: 1229098279882618000, flags:
Name: /VirtualBox/GuestAdd/Components/VBoxService.exe, value: <version-number>r40720,
    timestamp: 1229098279883195000, flags:
Name: /VirtualBox/GuestAdd/Components/VBoxTray.exe, value: <version-number>r40720,
    timestamp: 1229098279885027000, flags:
Name: /VirtualBox/GuestAdd/Components/VBoxGuest.sys, value: <version-number>r40720,
    timestamp: 1229098279886838000, flags:
Name: /VirtualBox/GuestAdd/Components/VBoxMouse.sys, value: <version-number>r40720,
    timestamp: 1229098279890600000, flags:
Name: /VirtualBox/GuestAdd/Components/VBoxSF.sys, value: <version-number>r40720,
    timestamp: 1229098279893056000, flags:
Name: /VirtualBox/GuestAdd/Components/VBoxVideo.sys, value: <version-number>r40720,
    timestamp: 1229098279895767000, flags:
Name: /VirtualBox/GuestInfo/OS/LoggedInUsers, value: 1,
    timestamp: 1229099826317660000, flags:
Name: /VirtualBox/GuestInfo/OS/NoLoggedInUsers, value: false,
    timestamp: 1229098455580553000, flags:
Name: /VirtualBox/GuestInfo/Net/Count, value: 1,
    timestamp: 1229099826299785000, flags:
Name: /VirtualBox/HostInfo/GUI/LanguageID, value: C,
    timestamp: 1229098151272771000, flags:
Name: /VirtualBox/GuestInfo/Net/0/V4/IP, value: 192.168.2.102,
    timestamp: 1229099826300088000, flags:
Name: /VirtualBox/GuestInfo/Net/0/V4/Broadcast, value: 255.255.255.255,
    timestamp: 1229099826300220000, flags:
Name: /VirtualBox/GuestInfo/Net/0/V4/Netmask, value: 255.255.255.0,
    timestamp: 1229099826300350000, flags:
Name: /VirtualBox/GuestInfo/Net/0/Status, value: Up,
    timestamp: 1229099826300524000, flags:
Name: /VirtualBox/GuestInfo/OS/LoggedInUsersList, value: username,
    timestamp: 1229099826317386000, flags:
```

To query the value of a single property, use the `get` subcommand as follows:

```
$ VBoxManage guestproperty get "Windows Vista III" "/VirtualBox/GuestInfo/OS/Product"
VirtualBox Command Line Management Interface Version <version-number>
(C) 2005-2018 Oracle Corporation
All rights reserved.

Value: Windows Vista Business Edition
```

To add or change guest properties from the guest, use the tool `VBoxControl`. This tool is included in the Guest Additions of Oracle VM VirtualBox 2.2 or later. When started from a Linux guest, this tool requires root privileges for security reasons:

```
$ sudo VBoxControl guestproperty enumerate
VirtualBox Guest Additions Command Line Management Interface Version <version-number>
(C) 2005-2018 Oracle Corporation
All rights reserved.

Name: /VirtualBox/GuestInfo/OS/Release, value: 2.6.28-18-generic,
    timestamp: 1265813265835667000, flags: <NULL>
Name: /VirtualBox/GuestInfo/OS/Version, value: #59-Ubuntu SMP Thu Jan 28 01:23:03 UTC 2010,
    timestamp: 1265813265836305000, flags: <NULL>
      ...
```

For more complex needs, you can use the Oracle VM VirtualBox programming interfaces. See chapter 11, *Oracle VM VirtualBox Programming Interfaces*, page 276.

### 4.7.1 Using Guest Properties to Wait on VM Events

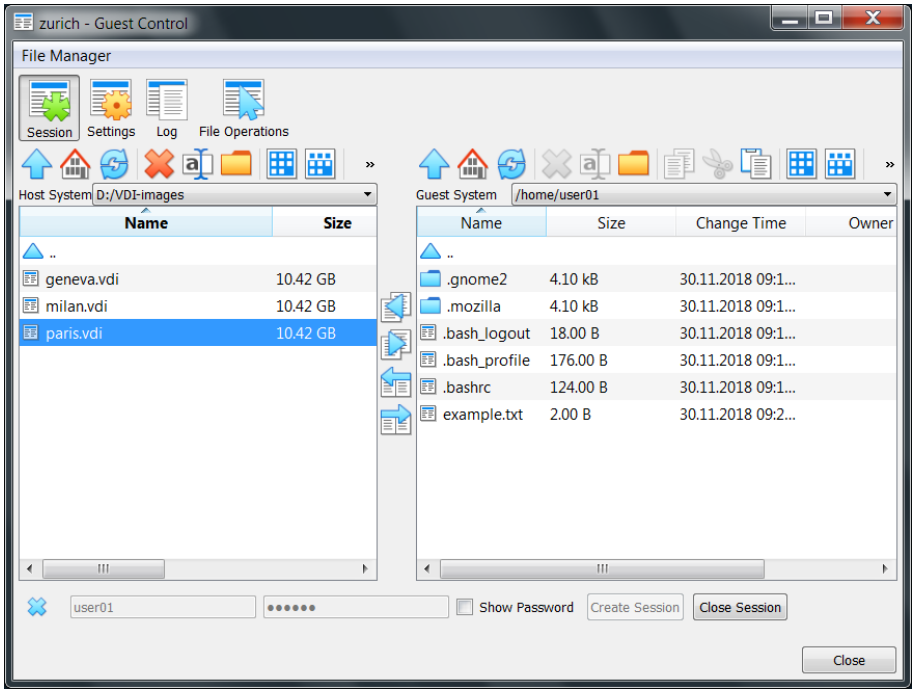
The properties `/VirtualBox/HostInfo/VBoxVer`, `/VirtualBox/HostInfo/VBoxVerExt` or `/VirtualBox/HostInfo/VBoxRev` can be waited on to detect that the VM state was restored from saved state or snapshot:

```
$ VBoxControl guestproperty wait /VirtualBox/HostInfo/VBoxVer
```

Similarly the `/VirtualBox/HostInfo/ResumeCounter` can be used to detect that a VM was resumed from the paused state or saved state.

## 4.8 Guest Control File Manager

The Guest Control File Manager is a feature of the Guest Additions that enables easy copying and moving of files between a guest and the host system. Other file management operations provide support to create new folders and to rename or delete files.




The Guest Control File Manager works by mounting the host file system. Guest users must authenticate and create a guest session before they can transfer files.

### 4.8.1 Using the Guest Control File Manager

The following steps describe how to use the Guest Control File Manager.

1. Open the Guest Control File Manager.

   In the guest VM, select **Machine**, **File Manager**.

   The left pane shows the files on the host system.

2. Create a guest session.

   At the bottom of the Guest Control File Manager, enter authentication credentials for a user on the guest system.

   Click **Create Session**.

   The contents of the guest VM file system appears in the right pane of the Guest Control File Manager.

3. Transfer files between the guest and the host system by using the move and copy file transfer icons.

   You can copy and move files from a guest to the host system or from the host system to the guest.

4. Close the Guest Control File Manager.

   Click **Close** to end the guest session.

## 4.9 Guest Control of Applications

The Guest Additions enable starting of applications inside a VM from the host system.

For this to work, the application needs to be installed inside the guest. No additional software needs to be installed on the host. Additionally, text mode output to stdout and stderr can be shown on the host for further processing. There are options to specify user credentials and a timeout value, in milliseconds, to limit the time the application is able to run.

This feature can be used to automate deployment of software within the guest.

The Guest Additions for Windows allow for automatic updating. This applies for already installed Guest Additions version 4.0 or later. Also, copying files from host to the guest as well as remotely creating guest directories is available.

To use these features, use the Oracle VM VirtualBox command line. See chapter 8.35, *VBoxManage guestcontrol*, page 177.

## 4.10 Memory Overcommitment

In server environments with many VMs, the Guest Additions can be used to share physical host memory between several VMs. This reduces the total amount of memory in use by the VMs. If memory usage is the limiting factor and CPU resources are still available, this can help with running more VMs on each host.

### 4.10.1 Memory Ballooning

The Guest Additions can change the amount of host memory that a VM uses, while the machine is running. Because of how this is implemented, this feature is called *memory ballooning*.

**Note:**

- Oracle VM VirtualBox supports memory ballooning only on 64-bit hosts. It is not supported on Mac OS X hosts.
- Memory ballooning does not work with large pages enabled. To turn off large pages support for a VM, run `VBoxManage modifyvm <VM name> --largepages off`

Normally, to change the amount of memory allocated to a virtual machine, you have to shut down the virtual machine entirely and modify its settings. With memory ballooning, memory that was allocated for a virtual machine can be given to another virtual machine without having to shut the machine down.

When memory ballooning is requested, the Oracle VM VirtualBox Guest Additions, which run inside the guest, allocate physical memory from the guest operating system on the kernel level and lock this memory down in the guest. This ensures that the guest will not use that memory any longer. No guest applications can allocate it, and the guest kernel will not use it either. Oracle VM VirtualBox can then reuse this memory and give it to another virtual machine.

The memory made available through the ballooning mechanism is only available for reuse by Oracle VM VirtualBox. It is *not* returned as free memory to the host. Requesting balloon memory from a running guest will therefore not increase the amount of free, unallocated memory on the host. Effectively, memory ballooning is therefore a memory overcommitment mechanism for multiple virtual machines while they are running. This can be useful to temporarily start another machine, or in more complicated environments, for sophisticated memory management of many virtual machines that may be running in parallel depending on how memory is used by the guests.

At this time, memory ballooning is only supported through `VBoxManage`. Use the following command to increase or decrease the size of the memory balloon within a running virtual machine that has Guest Additions installed:

```
VBoxManage controlvm "VM name" guestmemoryballoon n
```

where *VM name* is the name or UUID of the virtual machine in question and *n* is the amount of memory to allocate from the guest in megabytes. See chapter 8.14, *VBoxManage controlvm*, page 152.

You can also set a default balloon that will automatically be requested from the VM every time after it has started up with the following command:

```
VBoxManage modifyvm "VM name" --guestmemoryballoon n
```

By default, no balloon memory is allocated. This is a VM setting, like other `modifyvm` settings, and therefore can only be set while the machine is shut down. See chapter 8.8, *VBoxManage modifyvm*, page 135.

### 4.10.2 Page Fusion

Whereas memory ballooning simply reduces the amount of RAM that is available to a VM, Page Fusion works differently. It avoids memory duplication between several similar running VMs.

In a server environment running several similar VMs on the same host, lots of memory pages are identical. For example, if the VMs are using identical operating systems. Oracle VM VirtualBox's Page Fusion technology can efficiently identify these identical memory pages and share them between multiple VMs.

> **Note:** Oracle VM VirtualBox supports Page Fusion only on 64-bit hosts, and it is not supported on Mac OS X hosts. Page Fusion currently works only with Windows 2000 and later guests.

The more similar the VMs on a given host are, the more efficiently Page Fusion can reduce the amount of host memory that is in use. It therefore works best if all VMs on a host run identical operating systems, such as Windows XP Service Pack 2. Instead of having a complete copy of each operating system in each VM, Page Fusion identifies the identical memory pages in use by these operating systems and eliminates the duplicates, sharing host memory between several machines. This is called *deduplication*. If a VM tries to modify a page that has been shared with other VMs, a new page is allocated again for that VM with a copy of the shared page. This is called *copy on write*. All this is fully transparent to the virtual machine.

You may be familiar with this kind of memory overcommitment from other hypervisor products, which call this feature *page sharing* or *same page merging*. However, Page Fusion differs significantly from those other solutions, whose approaches have several drawbacks:

- Traditional hypervisors scan *all* guest memory and compute checksums, also called hashes, for every single memory page. Then, they look for pages with identical hashes and compare the entire content of those pages. If two pages produce the same hash, it is very likely that the pages are identical in content. This process can take rather long, especially if the system is not idling. As a result, the additional memory only becomes available after a significant amount of time, such as hours or sometimes days. Even worse, this kind of page sharing algorithm generally consumes significant CPU resources and increases the virtualization overhead by 10 to 20%.

  Page Fusion in Oracle VM VirtualBox uses logic in the Oracle VM VirtualBox Guest Additions to quickly identify memory cells that are most likely identical across VMs. It can therefore achieve most of the possible savings of page sharing almost immediately and with almost no overhead.

- Page Fusion is also much less likely to be confused by identical memory that it will eliminate, just to learn seconds later that the memory will now change and having to perform a highly expensive and often service-disrupting reallocation.

At this time, Page Fusion can only be controlled with `VBoxManage`, and only while a VM is shut down. To enable Page Fusion for a VM, use the following command:

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

You can observe Page Fusion operation using some metrics. `RAM/VMM/Shared` shows the total amount of fused pages, whereas the per-VM metric `Guest/RAM/Usage/Shared` will return the amount of fused memory for a given VM. See chapter 8.36, *VBoxManage metrics*, page 188 for information on how to query metrics.

**Note:** Enabling Page Fusion might indirectly increase the chances for malicious guests to successfully attack other VMs running on the same host. See chapter 13.3.4, *Potentially Insecure Operations*, page 298.

# 5 Virtual Storage

As the virtual machine will most probably expect to see a hard disk built into its virtual computer, Oracle VM VirtualBox must be able to present real storage to the guest as a virtual hard disk. There are presently three methods by which to achieve this:

- Oracle VM VirtualBox can use large image files on a real hard disk and present them to a guest as a virtual hard disk. This is the most common method, described in chapter 5.2, *Disk Image Files (VDI, VMDK, VHD, HDD)*, page 86.
- iSCSI storage servers can be attached to Oracle VM VirtualBox. This is described in chapter 5.10, *iSCSI Servers*, page 95.
- You can allow a virtual machine to access one of your host disks directly. This is an advanced feature, described in chapter 9.8.1, *Using a Raw Host Hard Disk From a Guest*, page 217.

Each such virtual storage device, such as an image file, iSCSI target, or physical hard disk, needs to be connected to the virtual hard disk controller that Oracle VM VirtualBox presents to a virtual machine. This is explained in the next section.

## 5.1 Hard Disk Controllers: IDE, SATA (AHCI), SCSI, SAS, USB MSD, NVMe

In a real PC, hard disks and CD/DVD drives are connected to a device called hard disk controller which drives hard disk operation and data transfers. Oracle VM VirtualBox can emulate the five most common types of hard disk controllers typically found in today's PCs: IDE, SATA (AHCI), SCSI, SAS, USB-based, and NVMe mass storage devices.

- **IDE (ATA)** controllers are a backwards compatible yet very advanced extension of the disk controller in the IBM PC/AT (1984). Initially, this interface worked only with hard disks, but was later extended to also support CD-ROM drives and other types of removable media. In physical PCs, this standard uses flat ribbon parallel cables with 40 or 80 wires. Each such cable can connect two devices to a controller, which have traditionally been called *master* and *slave*. Typical PCs had two connectors for such cables. As a result, support for up to four IDE devices was most common.

  In Oracle VM VirtualBox, each virtual machine may have one IDE controller enabled, which gives you up to four virtual storage devices that you can attach to the machine. By default, one of these virtual storage devices, the secondary master, is preconfigured to be the virtual machine's virtual CD/DVD drive. However, you can change the default setting.

  Even if your guest OS has no support for SCSI or SATA devices, it should always be able to see an IDE controller.

  You can also select which exact type of IDE controller hardware Oracle VM VirtualBox should present to the virtual machine: PIIX3, PIIX4, or ICH6. This makes no difference in terms of performance, but if you import a virtual machine from another virtualization product, the OS in that machine may expect a particular controller type and crash if it is not found.

After you have created a new virtual machine with the **New Virtual Machine** wizard of the graphical user interface, you will typically see one IDE controller in the machine's **Storage** settings. The virtual CD/DVD drive will be attached to one of the four ports of this controller.

- **Serial ATA (SATA)** is a newer standard introduced in 2003. Compared to IDE, it supports both much higher speeds and more devices per controller. Also, with physical hardware, devices can be added and removed while the system is running. The standard interface for SATA controllers is called Advanced Host Controller Interface (AHCI).

  Like a real SATA controller, Oracle VM VirtualBox's virtual SATA controller operates faster and also consumes fewer CPU resources than the virtual IDE controller. Also, this enables you to connect up to 30 virtual hard disks to one machine instead of just three, when compared to the Oracle VM VirtualBox IDE controller with a DVD drive attached.

  For this reason, depending on the selected guest OS, Oracle VM VirtualBox uses SATA as the default for newly created virtual machines. One virtual SATA controller is created by default, and the default disk that is created with a new VM is attached to this controller.

  **Warning:** The entire SATA controller and the virtual disks attached to it, including those in IDE compatibility mode, will not be seen by OSes that do not have device support for AHCI. In particular, *there is no support for AHCI in Windows before Windows Vista*. So Windows XP, even SP3, will not see such disks unless you install additional drivers. It is possible to switch from IDE to SATA after installation by installing the SATA drivers and changing the controller type in the VM **Settings** dialog.
  Oracle VM VirtualBox recommends the Intel Matrix Storage drivers, which can be downloaded from `http://downloadcenter.intel.com/Product_Filter.aspx?ProductID=2101`.

  To add a SATA controller to a machine for which it has not been enabled by default, either because it was created by an earlier version of Oracle VM VirtualBox, or because SATA is not supported by default by the selected guest OS, do the following. Go to the **Storage** page of the machine's **Settings** dialog, click **Add Controller** under the Storage Tree box and then select **Add SATA Controller**. The new controller appears as a separate PCI device in the virtual machine, and you can add virtual disks to it.

  To change the IDE compatibility mode settings for the SATA controller, see chapter 8.20, *VBoxManage storagectl*, page 164.

- **SCSI** is another established industry standard, standing for Small Computer System Interface. SCSI was standardized as early as 1986 as a generic interface for data transfer between all kinds of devices, including storage devices. Today SCSI is still used for connecting hard disks and tape devices, but it has mostly been displaced in commodity hardware. It is still in common use in high-performance workstations and servers.

  Primarily for compatibility with other virtualization software, Oracle VM VirtualBox optionally supports LSI Logic and BusLogic SCSI controllers, to each of which up to 15 virtual hard disks can be attached.

  To enable a SCSI controller, on the **Storage** page of a virtual machine's **Settings** dialog, click **Add Controller** under the Storage Tree box and then select **Add SCSI Controller**. The new controller appears as a separate PCI device in the virtual machine.

  **Warning:** As with the other controller types, a SCSI controller will only be seen by OSes with device support for it. Windows 2003 and later ships with drivers for the LSI Logic controller, while Windows NT 4.0 and Windows 2000 ships with drivers for the BusLogic controller. Windows XP ships with drivers for neither.

- **Serial Attached SCSI (SAS)** is another bus standard which uses the SCSI command set. As opposed to SCSI, however, with physical devices, serial cables are used instead of parallel ones, which simplifies physical device connections. In some ways, therefore, SAS is to SCSI what SATA is to IDE: it enables more reliable and faster connections.

  To support high-end guests which require SAS controllers, Oracle VM VirtualBox emulates a LSI Logic SAS controller, which can be enabled much the same way as a SCSI controller. At this time, up to eight devices can be connected to the SAS controller.

  **Warning:** As with SATA, the SAS controller will only be seen by OSes with device support for it. In particular, *there is no support for SAS in Windows before Windows Vista*. So Windows XP, even SP3, will not see such disks unless you install additional drivers.

- The **USB mass storage device class** is a standard to connect external storage devices like hard disks or flash drives to a host through USB. All major OSes support these devices for a long time and ship generic drivers making third-party drivers superfluous. In particular, legacy OSes without support for SATA controllers may benefit from USB mass storage devices.

  The virtual USB storage controller offered by Oracle VM VirtualBox works differently to the other storage controller types. While most storage controllers appear as a single PCI device to the guest with multiple disks attached to it, the USB-based storage controller does not appear as virtual storage controller. Each disk attached to the controller appears as a dedicated USB device to the guest.

  **Warning:** Booting from drives attached using USB is only supported when EFI is used as the BIOS lacks USB support.

- **Non volatile memory express (NVMe)** is a standard which emerged in 2011 for connecting non volatile memory (NVM) directly over PCI express to lift the bandwidth limitation of the previously used SATA protocol for SSDs. Unlike other standards the command set is very simple to achieve maximum throughput and is not compatible with ATA or SCSI. OSes need to support NVMe devices to make use of them. For example, Windows 8.1 added native NVMe support. For Windows 7, native support was added with an update.

  The NVMe controller is part of the extension pack.

  **Warning:** Booting from drives attached using NVMe is only supported when EFI is used as the BIOS lacks the appropriate driver.

In summary, Oracle VM VirtualBox gives you the following categories of virtual storage slots:

- Four slots attached to the traditional IDE controller, which are always present. One of these is typically a virtual CD/DVD drive.
- 30 slots attached to the SATA controller, if enabled and supported by the guest OS.
- 15 slots attached to the SCSI controller, if enabled and supported by the guest OS.
- Eight slots attached to the SAS controller, if enabled and supported by the guest OS.

- Eight slots attached to the virtual USB controller, if enabled and supported by the guest OS.
- Up to 255 slots attached to the NVMe controller, if enabled and supported by the guest OS.

Given this large choice of storage controllers, you may not know which one to choose. In general, you should avoid IDE unless it is the only controller supported by your guest. Whether you use SATA, SCSI, or SAS does not make any real difference. The variety of controllers is only supplied by Oracle VM VirtualBox for compatibility with existing hardware and other hypervisors.

## 5.2 Disk Image Files (VDI, VMDK, VHD, HDD)

Disk image files reside on the host system and are seen by the guest systems as hard disks of a certain geometry. When a guest OS reads from or writes to a hard disk, Oracle VM VirtualBox redirects the request to the image file.

Like a physical disk, a virtual disk has a size, or capacity, which must be specified when the image file is created. As opposed to a physical disk however, Oracle VM VirtualBox enables you to expand an image file after creation, even if it has data already. See chapter 8.24, *VBoxManage modifymedium*, page 167.

Oracle VM VirtualBox supports the following types of disk image files:

- **VDI.** Normally, Oracle VM VirtualBox uses its own container format for guest hard disks. This is called a Virtual Disk Image (VDI) file. This format is used when you create a new virtual machine with a new disk.
- **VMDK.** Oracle VM VirtualBox also fully supports the popular and open VMDK container format that is used by many other virtualization products, such as VMware.
- **VHD.** Oracle VM VirtualBox also fully supports the VHD format used by Microsoft.
- **HDD.** Image files of Parallels version 2 (HDD format) are also supported.

  Due to lack of documentation of the format, newer versions such as 3 and 4 are not supported. You can however convert such image files to version 2 format using tools provided by Parallels.

Irrespective of the disk capacity and format, as mentioned in chapter 1.8, *Creating Your First Virtual Machine*, page 8, there are two options for creating a disk image: fixed-size or dynamically allocated.

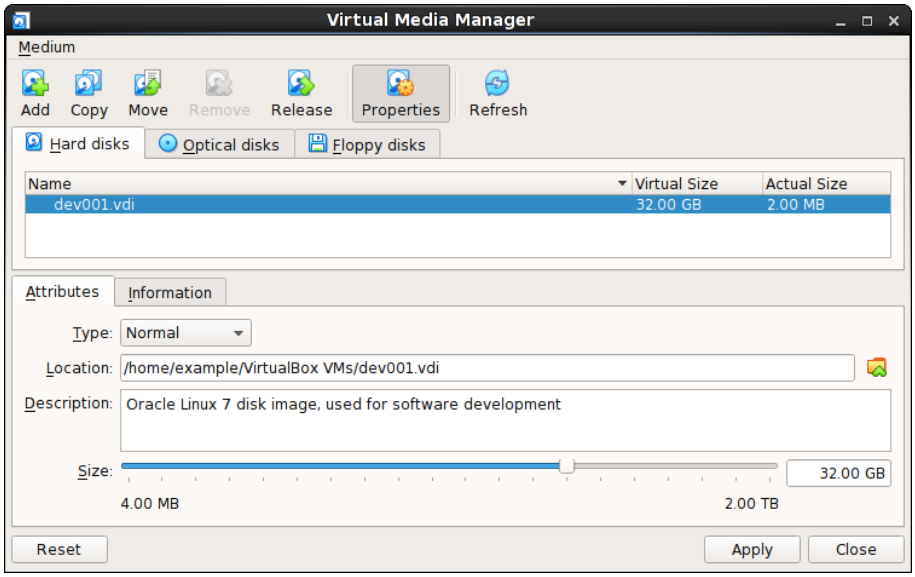
- **Fixed-size.** If you create a fixed-size image, an image file will be created on your host system which has roughly the same size as the virtual disk's capacity. So, for a 10 GB disk, you will have a 10 GB file. Note that the creation of a fixed-size image can take a long time depending on the size of the image and the write performance of your hard disk.
- **Dynamically allocated.** For more flexible storage management, use a dynamically allocated image. This will initially be very small and not occupy any space for unused virtual disk sectors, but will grow every time a disk sector is written to for the first time, until the drive reaches the maximum capacity chosen when the drive was created. While this format takes less space initially, the fact that Oracle VM VirtualBox needs to expand the image file consumes additional computing resources, so until the disk file size has stabilized, write operations may be slower than with fixed size disks. However, after a time the rate of growth will slow and the average penalty for write operations will be negligible.

## 5.3 The Virtual Media Manager

Oracle VM VirtualBox keeps track of all the hard disk, CD/DVD-ROM, and floppy disk images which are in use by virtual machines. These are often referred to as *known media* and come from two sources:

- All media currently attached to virtual machines.
- Registered media, for compatibility with Oracle VM VirtualBox versions older than version 4.0. For details about how media registration has changed with version 4.0, see chapter 10.1, *Where Oracle VM VirtualBox Stores its Files*, page 265.

The known media can be viewed and changed using the **Virtual Media Manager**, which you can access from the **File** menu in the VirtualBox Manager window.




The known media are conveniently grouped in separate tabs for the supported formats. These formats are:

- Hard disk images, either in Oracle VM VirtualBox's own Virtual Disk Image (VDI) format, or in the third-party formats listed in chapter 5.2, *Disk Image Files (VDI, VMDK, VHD, HDD)*, page 86.
- CD/DVD images in standard ISO format.
- Floppy images in standard RAW format.

For each image, the Virtual Media Manager shows you the full path of the image file and other information, such as the virtual machine the image is currently attached to.

The Virtual Media Manager enables you to do the following:

- **Add** an image to the registry.
- **Copy** a virtual hard disk to create another one.

  You can specify one of the following target types: VDI, VHD, or VMDK.

- **Move** an image that is currently in the registry to another location.

  A file dialog prompts you for the new image file location.

  When you use the Virtual Media Manager to move a disk image, Oracle VM VirtualBox updates all related configuration files automatically.

  **Note:** Always use the Virtual Media Manager or the `VBoxManage modifymedium` command to move a disk image.
  If you use a file management feature of the host OS to move a disk image to a new location, run the `VBoxManage modifymedium --setlocation` command to configure the new path of the disk image on the host file system. This command updates the Oracle VM VirtualBox configuration automatically.

- **Remove** an image from the registry. You can optionally delete the image file when removing the image.
- **Release** an image to detach it from a VM. This action only applies if the image is currently attached to a VM as a virtual hard disk.
- View and edit the **Properties** of a disk image.

  Available properties include the following:

  - **Type:** Specifies the snapshot behavior of the disk. See chapter 5.4, *Special Image Write Modes*, page 89.
  - **Location:** Specifies the location of the disk image file on the host system. You can use a file dialog to browse for the disk image location.
  - **Description:** Specifies a short description of the disk image.
  - **Size:** Specifies the size of the disk image. You can use the slider to increase or decrease the disk image size.
  - **Information:** Specifies detailed information about the disk image.
- **Refresh** the property values of the selected disk image.

To perform these actions, highlight the medium in the Virtual Media Manager and then do one of the following:

- Click an icon in the Virtual Media Manager task bar.
- Right-click the medium and select an option.

Use the **Storage** page in a VM's **Settings** dialog to create a new disk image. By default, disk images are stored in the VM's folder.

You can copy hard disk image files to other host systems and import them in to VMs from the host system. However, certain guest OSes, such as Windows 2000 and Windows XP, require that you configure the new VM in a similar way to the old one.

**Note:** Do not simply make copies of virtual disk images. If you import such a second copy into a VM, Oracle VM VirtualBox issues an error because Oracle VM VirtualBox assigns a universally unique identifier (UUID) to each disk image to ensure that it is only used one time. See chapter 5.6, *Cloning Disk Images*, page 92. Also, if you want to copy a VM to another system, use the Oracle VM VirtualBox import and export features. See chapter 1.15, *Importing and Exporting Virtual Machines*, page 21.

## 5.4 Special Image Write Modes

For each virtual disk image supported by Oracle VM VirtualBox, you can determine separately how it should be affected by write operations from a virtual machine and snapshot operations. This applies to all of the aforementioned image formats (VDI, VMDK, VHD, or HDD) and irrespective of whether an image is fixed-size or dynamically allocated.

By default, images are in *normal* mode. To mark an existing image with one of the non-standard modes listed below, use `VBoxManage modifyhd`. See chapter 8.24, *VBoxManage modifymedium*, page 167. Alternatively, use `VBoxManage` to attach the image to a VM and use the `--mtype` argument. See chapter 8.19, *VBoxManage storageattach*, page 160.

The available virtual disk image modes are as follows:

- **Normal images** have no restrictions on how guests can read from and write to the disk. This is the default image mode.

  When you take a snapshot of your virtual machine as described in chapter 1.11, *Snapshots*, page 17, the state of a normal hard disk is recorded together with the snapshot, and when reverting to the snapshot, its state will be fully reset.

  The image file itself is not reset. Instead, when a snapshot is taken, Oracle VM VirtualBox "freezes" the image file and no longer writes to it. For the write operations from the VM, a second, *differencing* image file is created which receives only the changes to the original image. See chapter 5.5, *Differencing Images*, page 90.

  While you can attach the same normal image to more than one virtual machine, only one of these virtual machines attached to the same image file can be executed simultaneously, as otherwise there would be conflicts if several machines write to the same image file.

- **Write-through hard disks** are completely unaffected by snapshots. Their state is *not* saved when a snapshot is taken, and not restored when a snapshot is restored.

- **Shareable hard disks** are a variant of write-through hard disks. In principle they behave exactly the same. Their state is *not* saved when a snapshot is taken, and not restored when a snapshot is restored. The difference only shows if you attach such disks to several VMs. Shareable disks may be attached to several VMs which may run concurrently. This makes them suitable for use by cluster filesystems between VMs and similar applications which are explicitly prepared to access a disk concurrently. Only fixed size images can be used in this way, and dynamically allocated images are rejected.

  **Warning:** This is an expert feature, and misuse can lead to data loss, as regular filesystems are not prepared to handle simultaneous changes by several parties.

- **Immutable images** only remember write accesses temporarily while the virtual machine is running. All changes are lost when the virtual machine is powered on the next time. As a result, as opposed to Normal images, the same immutable image can be used with several virtual machines without restrictions.

  Creating an immutable image makes little sense since it would be initially empty and lose its contents with every machine restart. You would have a disk that is always unformatted when the machine starts up. Instead, you can first create a normal image and then later mark it as immutable when you decide that the contents are useful.

  If you take a snapshot of a machine with immutable images, then on every machine power-up, those images are reset to the state of the last (current) snapshot, instead of the state of the original immutable image.

**Note:** As a special exception, immutable images are *not* reset if they are attached to a machine in a saved state or whose last snapshot was taken while the machine was running. This is called an *online snapshot*. As a result, if the machine's current snapshot is an online snapshot, its immutable images behave exactly like the a normal image. To reenable the automatic resetting of such images, delete the current snapshot of the machine.

Oracle VM VirtualBox never writes to an immutable image directly at all. All write operations from the machine are directed to a differencing image. The next time the VM is powered on, the differencing image is reset so that every time the VM starts, its immutable images have exactly the same content.

The differencing image is only reset when the machine is powered on from within Oracle VM VirtualBox, not when you reboot by requesting a reboot from within the machine. This is also why immutable images behave as described above when snapshots are also present, which use differencing images as well.

If the automatic discarding of the differencing image on VM startup does not fit your needs, you can turn it off using the `autoreset` parameter of `VBoxManage modifyhd`. See chapter 8.24, *VBoxManage modifymedium*, page 167.

- **Multiattach mode images** can be attached to more than one virtual machine at the same time, even if these machines are running simultaneously. For each virtual machine to which such an image is attached, a differencing image is created. As a result, data that is written to such a virtual disk by one machine is not seen by the other machines to which the image is attached. Each machine creates its own write history of the multiattach image.

  Technically, a multiattach image behaves identically to an immutable image except the differencing image is not reset every time the machine starts.

  This mode is useful for sharing files which are almost never written, for instance picture galleries, where every guest changes only a small amount of data and the majority of the disk content remains unchanged. The modified blocks are stored in differencing images which remain relatively small and the shared content is stored only once at the host.

- **Read-only images** are used automatically for CD/DVD images, since CDs/DVDs can never be written to.

The following scenario illustrates the differences between the various image modes, with respect to snapshots.

Assume you have installed your guest OS in your VM, and you have taken a snapshot. Later, your VM is infected with a virus and you would like to go back to the snapshot. With a normal hard disk image, you simply restore the snapshot, and the earlier state of your hard disk image will be restored as well and your virus infection will be undone. With an immutable hard disk, all it takes is to shut down and power on your VM, and the virus infection will be discarded. With a write-through image however, you cannot easily undo the virus infection by means of virtualization, but will have to disinfect your virtual machine like a real computer.

You might find write-through images useful if you want to preserve critical data irrespective of snapshots. As you can attach more than one image to a VM, you may want to have one immutable image for the OS and one write-through image for your data files.

## 5.5 Differencing Images

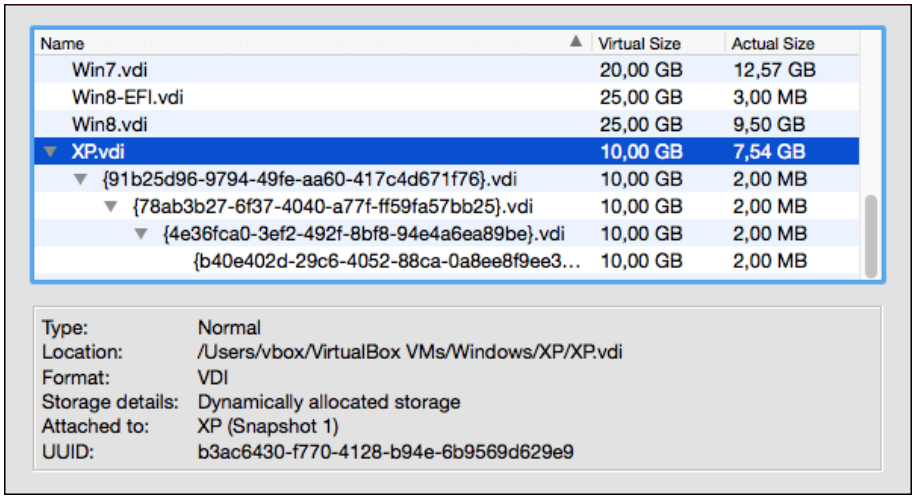
The previous section mentioned differencing images and how they are used with snapshots, immutable images, and multiple disk attachments. This section describes in more detail how differencing images work.

A differencing image is a special disk image that only holds the differences to another image. A differencing image by itself is useless, it must always refer to another image. The differencing image is then typically referred to as a *child*, which holds the differences to its *parent*.

When a differencing image is active, it receives all write operations from the virtual machine instead of its parent. The differencing image only contains the sectors of the virtual hard disk that have changed since the differencing image was created. When the machine reads a sector from such a virtual hard disk, it looks into the differencing image first. If the sector is present, it is returned from there. If not, Oracle VM VirtualBox looks into the parent. In other words, the parent becomes *read-only*. It is never written to again, but it is read from if a sector has not changed.

Differencing images can be chained. If another differencing image is created for a virtual disk that already has a differencing image, then it becomes a *grandchild* of the original parent. The first differencing image then becomes read-only as well, and write operations only go to the second-level differencing image. When reading from the virtual disk, Oracle VM VirtualBox needs to look into the second differencing image first, then into the first if the sector was not found, and then into the original image.

There can be an unlimited number of differencing images, and each image can have more than one child. As a result, the differencing images can form a complex tree with parents, siblings, and children, depending on how complex your machine configuration is. Write operations always go to the one *active* differencing image that is attached to the machine, and for read operations, Oracle VM VirtualBox may need to look up all the parents in the chain until the sector in question is found. You can view such a tree in the Virtual Media Manager.




In all of these situations, from the point of view of the virtual machine, the virtual hard disk behaves like any other disk. While the virtual machine is running, there is a slight run-time I/O overhead because Oracle VM VirtualBox might need to look up sectors several times. This is not noticeable however since the tables with sector information are always kept in memory and can be looked up quickly.

Differencing images are used in the following situations:

- **Snapshots.** When you create a snapshot, as explained in the previous section, Oracle VM VirtualBox "freezes" the images attached to the virtual machine and creates differencing images for each image that is not in "write-through" mode. From the point of view of the virtual machine, the virtual disks continue to operate before, but all write operations go into the differencing images. Each time you create another snapshot, for each hard disk attachment, another differencing image is created and attached, forming a chain or tree.

  In the above screenshot, you see that the original disk image is now attached to a snapshot, representing the state of the disk when the snapshot was taken.

If you *restore* a snapshot, and want to go back to the exact machine state that was stored in the snapshot, the following happens:

- Oracle VM VirtualBox copies the virtual machine settings that were copied into the snapshot back to the virtual machine. As a result, if you have made changes to the machine configuration since taking the snapshot, they are undone.
- If the snapshot was taken while the machine was running, it contains a saved machine state, and that state is restored as well. After restoring the snapshot, the machine will then be in Saved state and resume execution from there when it is next started. Otherwise the machine will be in Powered Off state and do a full boot.
- For each disk image attached to the machine, the differencing image holding all the write operations since the current snapshot was taken is thrown away, and the original parent image is made active again. If you restored the root snapshot, then this will be the root disk image for each attachment. Otherwise, some other differencing image descended from it. This effectively restores the old machine state.

If you later *delete* a snapshot in order to free disk space, for each disk attachment, one of the differencing images becomes obsolete. In this case, the differencing image of the disk attachment cannot simply be deleted. Instead, Oracle VM VirtualBox needs to look at each sector of the differencing image and needs to copy it back into its parent. This is called "merging" images and can be a potentially lengthy process, depending on how large the differencing image is. It can also temporarily need a considerable amount of extra disk space, before the differencing image obsoleted by the merge operation is deleted.

- **Immutable images.** When an image is switched to immutable mode, a differencing image is created as well. As with snapshots, the parent image then becomes read-only, and the differencing image receives all the write operations. Every time the virtual machine is started, all the immutable images which are attached to it have their respective differencing image thrown away, effectively resetting the virtual machine's virtual disk with every restart.

## 5.6 Cloning Disk Images

You can duplicate hard disk image files on the same host to quickly produce a second virtual machine with the same OS setup. However, you should *only* make copies of virtual disk images using the utility supplied with Oracle VM VirtualBox. See chapter 8.25, *VBoxManage clonemedium*, page 168. This is because Oracle VM VirtualBox assigns a UUID to each disk image, which is also stored inside the image, and Oracle VM VirtualBox will refuse to work with two images that use the same number. If you do accidentally try to reimport a disk image which you copied normally, you can make a second copy using the `VBoxManage clonevm` command and import that instead.

Note that newer Linux distributions identify the boot hard disk from the ID of the drive. The ID Oracle VM VirtualBox reports for a drive is determined from the UUID of the virtual disk image. So if you clone a disk image and try to boot the copied image the guest might not be able to determine its own boot disk as the UUID changed. In this case you have to adapt the disk ID in your boot loader script, for example `/boot/grub/menu.lst`. The disk ID looks like the following:

```
scsi-SATA_VBOX_HARDDISK_VB5cfdb1e2-c251e503
```

The ID for the copied image can be determined as follows:

```
hdparm -i /dev/sda
```

## 5.7 Host Input/Output Caching

Oracle VM VirtualBox can optionally disable the I/O caching that the host OS would otherwise perform on disk image files.

Traditionally, Oracle VM VirtualBox has opened disk image files as normal files, which results in them being cached by the host OS like any other file. The main advantage of this is speed: when the guest OS writes to disk and the host OS cache uses delayed writing, the write operation can be reported as completed to the guest OS quickly while the host OS can perform the operation asynchronously. Also, when you start a VM a second time and have enough memory available for the OS to use for caching, large parts of the virtual disk may be in system memory, and the VM can access the data much faster.

Note that this applies only to image files. Buffering does not occur for virtual disks residing on remote iSCSI storage, which is the more common scenario in enterprise-class setups. See chapter 5.10, *iSCSI Servers*, page 95.

While buffering is a useful default setting for virtualizing a few machines on a desktop computer, there are some disadvantages to this approach:

- Delayed writing through the host OS cache is less secure. When the guest OS writes data, it considers the data written even though it has not yet arrived on a physical disk. If for some reason the write does not happen, such as power failure or host crash, the likelihood of data loss increases.
- Disk image files tend to be very large. Caching them can therefore quickly use up the entire host OS cache. Depending on the efficiency of the host OS caching, this may slow down the host immensely, especially if several VMs run at the same time. For example, on Linux hosts, host caching may result in Linux delaying all writes until the host cache is nearly full and then writing out all these changes at once, possibly stalling VM execution for minutes. This can result in I/O errors in the guest as I/O requests time out there.
- Physical memory is often wasted as guest OSes typically have their own I/O caches, which may result in the data being cached twice, in both the guest and the host caches, for little effect.

If you decide to disable host I/O caching for the above reasons, Oracle VM VirtualBox uses its own small cache to buffer writes, but no read caching since this is typically already performed by the guest OS. In addition, Oracle VM VirtualBox fully supports asynchronous I/O for its virtual SATA, SCSI, and SAS controllers through multiple I/O threads.

Since asynchronous I/O is not supported by IDE controllers, for performance reasons, you may want to leave host caching enabled for your VM's virtual IDE controllers.

For this reason, Oracle VM VirtualBox enables you to configure whether the host I/O cache is used for each I/O controller separately. Either select the **Use Host I/O Cache** check box in the **Storage** settings for a given virtual storage controller, or use the following `VBoxManage` command to disable the host I/O cache for a virtual storage controller:

```
VBoxManage storagectl "VM name" --name <controllername> --hostiocache off
```

See chapter 8.20, *VBoxManage storagectl*, page 164.

For the above reasons, Oracle VM VirtualBox now uses SATA controllers by default for new virtual machines.

## 5.8 Limiting Bandwidth for Disk Images

Oracle VM VirtualBox supports limiting of 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 using `VBoxManage`. The example below creates a bandwidth group named Limit, sets the limit to 20 MB per second, 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" --storagectl "SATA" --port 0 --device 0 --type hdd
                                   --medium disk1.vdi --bandwidthgroup Limit
VBoxManage storageattach "VM name" --storagectl "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 example above the bandwidth of both images combined can never exceed 20 MBps. However, if one disk does not 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 example below changes the limit for the group created in the example above to 10 MBps:

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

## 5.9 CD/DVD Support

Virtual CD/DVD drives by default support only reading. The medium configuration is changeable at runtime. You can select between the following options to provide the medium data:

- **Host Drive** defines that the guest can read from the medium in the host drive.
- **Image file** gives the guest read-only access to the data in the image. This is typically an ISO file.
- **Empty** means a drive without an inserted medium.

Changing between the above, or changing a medium in the host drive that is accessed by a machine, or changing an image file will signal a medium change to the guest OS. The guest OS can then react to the change, for example by starting an installation program.

Medium changes can be prevented by the guest, and Oracle VM VirtualBox reflects that by locking the host drive if appropriate. You can force a medium removal in such situations by using the Oracle VM VirtualBox GUI or the `VBoxManage` command line tool. Effectively this is the equivalent of the emergency eject which many CD/DVD drives provide, with all associated side effects. The guest OS can issue error messages, just like on real hardware, and guest applications may misbehave. Use this with caution.

> **Note:** The identification string of the drive provided to the guest, displayed by configuration tools such as the Windows Device Manager, is always VBOX CD-ROM, irrespective of the current configuration of the virtual drive. This is to prevent hardware detection from being triggered in the guest OS every time the configuration is changed.

The standard CD/DVD emulation enables reading of standard data CD and DVD formats only. As an experimental feature, for additional capabilities, it is possible to give the guest direct access to the CD/DVD host drive by enabling *passthrough* mode. Depending on the host hardware, this may potentially enable the following things to work:

- CD/DVD writing from within the guest, if the host DVD drive is a CD/DVD writer
- Playing audio CDs
- Playing encrypted DVDs

There is a **Passthrough** check box in the GUI dialog for configuring the media attached to a storage controller, or you can use the `--passthrough` option with `VBoxManage storageattach`. See chapter 8.19, *VBoxManage storageattach*, page 160.

Even if passthrough is enabled, unsafe commands, such as updating the drive firmware, will be blocked. Video CD formats are never supported, not even in passthrough mode, and cannot be played from a virtual machine.

On Oracle Solaris hosts, passthrough requires running Oracle VM VirtualBox with real root permissions due to security measures enforced by the host.

## 5.10 iSCSI Servers

iSCSI stands for "Internet SCSI" and is a standard that supports use of the SCSI protocol over Internet (TCP/IP) connections. Especially with the advent of Gigabit Ethernet, it has become affordable to attach iSCSI storage servers simply as remote hard disks to a computer network. In iSCSI terminology, the server providing storage resources is called an *iSCSI target*, while the client connecting to the server and accessing its resources is called an *iSCSI initiator*.

Oracle VM VirtualBox can transparently present iSCSI remote storage to a virtual machine as a virtual hard disk. The guest OS will not see any difference between a virtual disk image (VDI file) and an iSCSI target. To achieve this, Oracle VM VirtualBox has an integrated iSCSI initiator.

Oracle VM VirtualBox's iSCSI support has been developed according to the iSCSI standard and should work with all standard-conforming iSCSI targets. To use an iSCSI target with Oracle VM VirtualBox, you must use the command line. See chapter 8.19, *VBoxManage storageattach*, page 160.

## 5.11 vboximg-mount: A Utility for FUSE Mounting a Virtual Disk Image

`vboximg-mount` is a command line utility for Mac OS X hosts that provides raw access to an Oracle VM VirtualBox virtual disk image on the host system. Use this utility to mount, view, and optionally modify the disk image contents.

The utility is based on Filesystem in Userspace (FUSE) technology and uses the VirtualBox runtime engine. Ensure that Oracle VM VirtualBox is running on the host system.

> **Note:** When using `vboximg-mount`, ensure that the following conditions apply:
>
> - The disk image is not being used by any other systems, such as by guest VMs.
> - No VMs are running on the host system.

Raw access using FUSE is preferred over direct loopback mounting of virtual disk images, because it is snapshot aware. It can selectively merge disk differencing images in an exposed virtual hard disk, providing historical or up-to-date representations of the virtual disk contents.

`vboximg-mount` enables you to view information about registered VMs, their attached disk media, and any snapshots. Also, you can view partition information for a disk image.

Use the `--help` option to view information about the `vboximg-mount` command usage.

When `vboximg-mount` mounts an Oracle VM VirtualBox disk image, it creates a one level deep file system at a mount point that you specify. The file system includes a device node that represents the synthesized disk image as a readable or readable-writeable bytestream. This bytestream can be mounted either by using the host OS or by using other FUSE-based file systems.

### 5.11.1 Viewing Detailed Information About a Virtual Disk Image

The following examples show how to use the `vboximg-mount` command to view information about virtual disk images.

The following command outputs detailed information about all registered VMs and associated snapshots:

```
$ vboximg-mount --list --verbose

    ------------------------------------------------------
    VM Name:   "macOS High Sierra 10.13"
    UUID:      3887d96d-831c-4187-a55a-567c504ff0e1
    Location:  /Volumes/work/vm_guests/macOS High Sierra 10.13/macOS High Sierra 10.13.vbox
       ----------------------
       HDD base:   "macOS High Sierra 10.13.vdi"
       UUID:       f9ea7173-6869-4aa9-b487-68023a655980
       Location:   /Volumes/work/vm_guests/macOS High Sierra 10.13/macOS High Sierra 10.13.vdi

         Diff 1:
              UUID:       98c2bac9-cf37-443d-a935-4e879b70166d
              Location:   /Volumes/work/vm_guests/macOS High Sierra 10.13/
              Snapshots/{98c2bac9-cf37-443d-a935-4e879b70166d}.vdi
         Diff 2:
              UUID:       f401f381-7377-40b3-948e-3c61241b1a42
              Location:   /Volumes/work/vm_guests/macOS High Sierra 10.13/
              Snapshots/{f401f381-7377-40b3-948e-3c61241b1a42}.vdi
       ----------------------
       HDD base:   "simple_fixed_disk.vdi"
       UUID:       ffba4d7e-1277-489d-8173-22ca7660773d
       Location:   /Volumes/work/vm_guests/macOS High Sierra 10.13/simple_fixed_disk.vdi

         Diff 1:
              UUID:       aecab681-0d2d-468b-8682-93f79dc97a48
              Location:   /Volumes/work/vm_guests/macOS High Sierra 10.13/
              Snapshots/{aecab681-0d2d-468b-8682-93f79dc97a48}.vdi
         Diff 2:
              UUID:       70d6b34d-8422-47fa-8521-3b6929a1971c
              Location:   /Volumes/work/vm_guests/macOS High Sierra 10.13/
              Snapshots/{70d6b34d-8422-47fa-8521-3b6929a1971c}.vdi
      ------------------------------------------------------
      VM Name:   "debian"
      UUID:      5365ab5f-470d-44c0-9863-dad532ee5905
      Location:  /Volumes/work/vm_guests/debian/debian.vbox
         ----------------------
         HDD base:   "debian.vdi"
         UUID:       96d2e92e-0d4e-46ab-a0f1-008fdbf997e7
         Location:   /Volumes/work/vm_guests/debian/ol7.vdi

            Diff 1:
                UUID:       f9cc866a-9166-42e9-a503-bbfe9b7312e8
                Location:   /Volumes/work/vm_guests/debian/Snapshots/
                {f9cc866a-9166-42e9-a503-bbfe9b7312e8}.vdi
```

The following command outputs partition information about the specified disk image:

```
$ vboximg-mount --image=f9ea7173-6869-4aa9-b487-68023a655980 --list

    Virtual disk image:

       Path: /Volumes/work/vm_guests/macOS High Sierra 10.13/macOS High Sierra 10.13.vdi
       UUID: f9ea7173-6869-4aa9-b487-68023a655980

      #     Start  Sectors     Size       Offset  Type
      1        40  409599    199.9M        20480  EFI System
      2    409640  67453071   32.1G    209735680  Hierarchical File System Plus (HFS+)
      3  67862712  1269535   107.8M  34745708544  Apple Boot (Recovery HD)
```

### 5.11.2 Mounting a Virtual Disk Image

The following steps show how to use the `vboximg-mount` command to mount a partition of a virtual disk image on the host OS.

1. Create a mount point on the host OS. For example:

   ```
   $ mkdir macos_sysdisk
   ```

2. Show partition information about the virtual disk image.

   ```
   $ vboximg-mount --image=<uuid> --list
   ```

   where *uuid* is the UUID of the disk image.

3. Use `vboximg-mount` to perform a FUSE mount of a partition on the virtual disk image. For example:

   ```
   $ vboximg-mount --image=<uuid> -p 2 macos_sysdisk
   ```

   where *uuid* is the UUID for the disk image.

   In this example, partition 2 is mounted on the `macos_sysdisk` mount point. The mount includes all snapshots for the disk image.

4. Use the host OS to mount the `vhdd` device node. The FUSE-mounted device node represents the virtual disk image.

   ```
   $ ls macos_sysdisk
      macOS High Sierra 10.13.vdi  vhdd
   $ sudo mount macos_sysdisk/vhdd /mnt
   ```

# 6 Virtual Networking

As mentioned in chapter 3.9, *Network Settings*, page 54, Oracle VM VirtualBox provides up to eight virtual PCI Ethernet cards for each virtual machine. For each such card, you can individually select the following:

- The hardware that will be virtualized.
- The virtualization mode that the virtual card operates in, with respect to your physical networking hardware on the host.

Four of the network cards can be configured in the **Network** section of the **Settings** dialog in the graphical user interface of Oracle VM VirtualBox. You can configure all eight network cards on the command line using `VBoxManage modifyvm`. See chapter 8.8, *VBoxManage modifyvm*, page 135.

This chapter explains the various networking settings in more detail.

## 6.1 Virtual Networking Hardware

For each card, you can individually select what kind of *hardware* will be presented to the virtual machine. Oracle VM VirtualBox can virtualize the following types of networking hardware:

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

The PCNet FAST III is the default because it is supported by nearly all operating systems, as well as by the GNU GRUB boot manager. As an exception, the Intel PRO/1000 family adapters are chosen for some guest operating system types that no longer ship with drivers for the PCNet card, such as Windows Vista.

The Intel PRO/1000 MT Desktop type works with Windows Vista and later versions. The T Server variant of the Intel PRO/1000 card is recognized by Windows XP guests without additional driver installation. The MT Server variant facilitates OVF imports from other platforms.

The Paravirtualized network adapter (virtio-net) is special. If you select this adapter, then Oracle VM VirtualBox does *not* virtualize common networking hardware that is supported by common guest operating systems. Instead, Oracle VM VirtualBox expects a special software interface for virtualized environments to be provided by the guest, thus avoiding the complexity of emulating networking hardware and improving network performance. Oracle VM VirtualBox provides support for the industry-standard *virtio* networking drivers, which are part of the open source KVM project.

The virtio networking drivers are available for the following guest operating systems:

- Linux kernels version 2.6.25 or later can be configured to provide virtio support. Some distributions have also back-ported virtio to older kernels.
- For Windows 2000, XP, and Vista, virtio drivers can be downloaded and installed from the KVM project web page:

  `http://www.linux-kvm.org/page/WindowsGuestDrivers`.

Oracle VM VirtualBox also has limited support for *jumbo frames*. These are networking packets with more than 1500 bytes of data, provided that you use the Intel card virtualization and bridged networking. Jumbo frames are not supported with the AMD networking devices. In those cases, jumbo packets will silently be dropped for both the transmit and the receive direction. Guest operating systems trying to use this feature will observe this as a packet loss, which may lead to unexpected application behavior in the guest. This does not cause problems with guest operating systems in their default configuration, as jumbo frames need to be explicitly enabled.

## 6.2 Introduction to Networking Modes

Each of the networking adapters can be separately configured to operate in one of the following modes:

- **Not attached.** In this mode, Oracle VM VirtualBox reports to the guest that a network card is present, but that there is no connection. This is as if no Ethernet cable was plugged into the card. Using this mode, it is possible to "pull" the virtual Ethernet cable and disrupt the connection, which can be useful to inform a guest operating system that no network connection is available and enforce a reconfiguration.
- **Network Address Translation (NAT).** If all you want is to browse the Web, download files, and view email inside the guest, then this default mode should be sufficient for you, and you can skip the rest of this section. Please note that there are certain limitations when using Windows file sharing. See chapter 6.3.3, *NAT Limitations*, page 102.
- **NAT Network.** A NAT network is a type of internal network that allows outbound connections. See chapter 6.4, *Network Address Translation Service*, page 102.
- **Bridged networking.** This is for more advanced networking needs, such as network simulations and running servers in a guest. When enabled, Oracle VM VirtualBox connects to one of your installed network cards and exchanges network packets directly, circumventing your host operating system's network stack.
- **Internal networking.** This can be used to create a different kind of software-based network which is visible to selected virtual machines, but not to applications running on the host or to the outside world.
- **Host-only networking.** This can be used to create a network containing the host and a set of virtual machines, without the need for the host's physical network interface. Instead, a virtual network interface, similar to a loopback interface, is created on the host, providing connectivity among virtual machines and the host.
- **Generic networking.** Rarely used modes which share the same generic network interface, by allowing the user to select a driver which can be included with Oracle VM VirtualBox or be distributed in an extension pack.

  The following sub-modes are available:

  – **UDP Tunnel:** Used to interconnect virtual machines running on different hosts directly, easily, and transparently, over an existing network infrastructure.

– **VDE (Virtual Distributed Ethernet) networking:** Used to connect to a Virtual Distributed Ethernet switch on a Linux or a FreeBSD host. At the moment this option requires compilation of Oracle VM VirtualBox from sources, as the Oracle packages do not include it.

The following table provides an overview of the most important networking modes.

| Mode | VM→Host | VM←Host | VM1↔VM2 | VM→Net/LAN | VM←Net/LAN |
|---|---|---|---|---|---|
| Host-only | + | + | + | – | – |
| Internal | – | – | + | – | – |
| Bridged | + | + | + | + | + |
| NAT | + | Port forward | – | + | Port forward |
| NATservice | + | Port forward | + | + | Port forward |

The following sections describe the available network modes in more detail.

## 6.3 Network Address Translation (NAT)

Network Address Translation (NAT) is the simplest way of accessing an external network from a virtual machine. Usually, it does not require any configuration on the host network and guest system. For this reason, it is the default networking mode in Oracle VM VirtualBox.

A virtual machine with NAT enabled acts much like a real computer that connects to the Internet through a router. The router, in this case, is the Oracle VM VirtualBox networking engine, which maps traffic from and to the virtual machine transparently. In Oracle VM VirtualBox this router is placed between each virtual machine and the host. This separation maximizes security since by default virtual machines cannot talk to each other.

The disadvantage of NAT mode is that, much like a private network behind a router, the virtual machine is invisible and unreachable from the outside internet. You cannot run a server this way unless you set up port forwarding. See chapter 6.3.1, *Configuring Port Forwarding with NAT*, page 100.

The network frames sent out by the guest operating system are received by Oracle VM VirtualBox's NAT engine, which extracts the TCP/IP data and resends it using the host operating system. To an application on the host, or to another computer on the same network as the host, it looks like the data was sent by the Oracle VM VirtualBox application on the host, using an IP address belonging to the host. Oracle VM VirtualBox listens for replies to the packages sent, and repacks and resends them to the guest machine on its private network.

The virtual machine receives its network address and configuration on the private network from a DHCP server integrated into Oracle VM VirtualBox. The IP address thus assigned to the virtual machine is usually on a completely different network than the host. As more than one card of a virtual machine can be set up to use NAT, the first card is connected to the private network 10.0.2.0, the second card to the network 10.0.3.0 and so on. If you need to change the guest-assigned IP range, see chapter 9.10, *Fine Tuning the Oracle VM VirtualBox NAT Engine*, page 221.

### 6.3.1 Configuring Port Forwarding with NAT

As the virtual machine is connected to a private network internal to Oracle VM VirtualBox and invisible to the host, network services on the guest are not accessible to the host machine or to

other computers on the same network. However, like a physical router, Oracle VM VirtualBox can make selected services available to the world outside the guest through *port forwarding*. This means that Oracle VM VirtualBox listens to certain ports on the host and resends all packets which arrive there to the guest, on the same or a different port.

To an application on the host or other physical or virtual machines 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. However, you still gain the advantages of running the service in a virtual machine. For example, services on the host machine or on other virtual machines cannot be compromised or crashed by a vulnerability or a bug in the service, and the service can run in a different operating system than the host system.

To configure port forwarding you can use the graphical **Port Forwarding** editor which can be found in the **Network Settings** dialog for network adaptors configured to use NAT. Here, you can map host ports to guest ports to allow network traffic to be routed to a specific port in the guest.

Alternatively, the command line tool `VBoxManage` can be used. See chapter 8.8, *VBoxManage modifyvm*, page 135.

You will need to know which ports on the guest the service uses and to decide which ports to use on the host. You may want to use the same ports on the guest and on the host. You can use any ports on the host which are not already in use by a service. For example, to set up incoming NAT connections to an `ssh` server in the guest, use the following command:

```
VBoxManage modifyvm "VM name" --natpf1 "guestssh,tcp,,2222,,22"
```

In the above example, all TCP traffic arriving on port 2222 on any host interface will be forwarded to port 22 in the guest. The protocol name `tcp` is a mandatory attribute defining which protocol should be used for forwarding, `udp` could also be used. The name `guestssh` is purely descriptive and will be auto-generated if omitted. The number after `--natpf` denotes the network card, as with other `VBoxManage` commands.

To remove this forwarding rule, use the following command:

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

If for some reason the guest uses a static assigned IP address not leased from the built-in DHCP server, it is required to specify the guest IP when registering the forwarding rule, as follows:

```
VBoxManage modifyvm "VM name" --natpf1 "guestssh,tcp,,2222,10.0.2.19,22"
```

This example is identical to the previous one, except that the NAT engine is being told that the guest can be found at the 10.0.2.19 address.

To forward *all* incoming traffic from a specific host interface to the guest, specify the IP of that host interface as follows:

```
VBoxManage modifyvm "VM name" --natpf1 "guestssh,tcp,127.0.0.1,2222,,22"
```

This example forwards all TCP traffic arriving on the localhost interface at 127.0.0.1 through port 2222 to port 22 in the guest.

It is possible to configure incoming NAT connections while the VM is running, see chapter 8.14, *VBoxManage controlvm*, page 152.

### 6.3.2 PXE Booting with NAT

PXE booting is now supported in NAT mode. The NAT DHCP server provides a boot file name of the form `vmname.pxe` if the directory `TFTP` exists in the directory where the user's `VirtualBox.xml` file is kept. It is the responsibility of the user to provide `vmname.pxe`.

### 6.3.3 NAT Limitations

There are some limitations of NAT mode which users should be aware of, as follows:

- **ICMP protocol limitations.** Some frequently used network debugging tools, such as `ping` or tracerouting, rely on the ICMP protocol for sending and receiving messages. While ICMP support has been improved with Oracle VM VirtualBox 2.1, meaning `ping` should now work, some other tools may not work reliably.
- **Receiving of UDP broadcasts.** The guest does not reliably receive UDP broadcasts. In order to save resources, it only listens for a certain amount of time after the guest has sent UDP data on a particular port. As a consequence, NetBios name resolution based on broadcasts does not always work, but WINS always works. As a workaround, you can use the numeric IP of the desired server in the `\\server\share` notation.
- **Some protocols are not supported.** Protocols other than TCP and UDP are not supported. GRE is not supported. This means some VPN products, such as PPTP from Microsoft, cannot be used. There are other VPN products which use only TCP and UDP.
- **Forwarding host ports below 1024.** On UNIX-based hosts, such as Linux, Oracle Solaris, and Mac OS X, it is not possible to bind to ports below 1024 from applications that are not run by `root`. As a result, if you try to configure such a port forwarding, the VM will refuse to start.

These limitations normally do not affect standard network use. But the presence of NAT has also subtle effects that may interfere with protocols that are normally working. One example is NFS, where the server is often configured to refuse connections from non-privileged ports, which are those ports not below 1024.

## 6.4 Network Address Translation Service

The Network Address Translation (NAT) service works in a similar way to a home router, grouping the systems using it into a network and preventing systems outside of this network from directly accessing systems inside it, but letting systems inside communicate with each other and with systems outside using TCP and UDP over IPv4 and IPv6.

A NAT service is attached to an internal network. Virtual machines which are to make use of it should be attached to that internal network. The name of internal network is chosen when the NAT service is created and the internal network will be created if it does not already exist. The following is an example command to create a NAT network:

```
VBoxManage natnetwork add --netname natnet1 --network "192.168.15.0/24" --enable
```

Here, natnet1 is the name of the internal network to be used and 192.168.15.0/24 is the network address and mask of the NAT service interface. By default in this static configuration the gateway will be assigned the address 192.168.15.1, the address following the interface address, though this is subject to change. To attach a DHCP server to the internal network, modify the example command as follows:

```
VBoxManage natnetwork add --netname natnet1 --network "192.168.15.0/24" --enable --dhcp on
```

To add a DHCP server to an existing network, use the following command:

```
VBoxManage natnetwork modify --netname natnet1 --dhcp on
```

To disable the DHCP server, use the following command:

```
VBoxManage natnetwork modify --netname natnet1 --dhcp off
```

A DHCP server provides a list of registered nameservers, but does not map servers from the 127/8 network.

To start the NAT service, use the following command:

```
VBoxManage natnetwork start --netname natnet1
```

If the network has a DHCP server attached then it will start together with the NAT network service.

To stop the NAT network service, together with any DHCP server:

```
VBoxManage natnetwork stop --netname natnet1
```

To delete the NAT network service:

```
VBoxManage natnetwork remove --netname natnet1
```

This command does not remove the DHCP server if one is enabled on the internal network.

Port-forwarding is supported, using the `--port-forward-4` switch for IPv4 and `--port-forward-6` for IPv6. For example:

```
VBoxManage natnetwork modify \
  --netname natnet1 --port-forward-4 "ssh:tcp:[]:1022:[192.168.15.5]:22"
```

This adds a port-forwarding rule from the host's TCP 1022 port to the port 22 on the guest with IP address 192.168.15.5. Host port, guest port and guest IP are mandatory. To delete the rule, use the following command:

```
VBoxManage natnetwork modify --netname natnet1 --port-forward-4 delete ssh
```

It is possible to bind a NAT service to specified interface. For example:

```
VBoxManage setextradata global "NAT/win-nat-test-0/SourceIp4" 192.168.1.185
```

To see the list of registered NAT networks, use the following command:

```
VBoxManage list natnetworks
```

## 6.5 Bridged Networking

With bridged networking, Oracle VM VirtualBox uses a device driver on your *host* system that filters data from your physical network adapter. This driver is therefore called a *net filter* driver. This enables Oracle VM VirtualBox to intercept data from the physical network and inject data into it, effectively creating a new network interface in software. When a guest is using such a new software interface, it looks to the host system as though the guest were physically connected to the interface using a network cable. The host can send data to the guest through that interface and receive data from it. This means that you can set up routing or bridging between the guest and the rest of your network.

> **Note:** Even though TAP interfaces are no longer necessary on Linux for bridged networking, you *can* still use TAP interfaces for certain advanced setups, since you can connect a VM to any host interface.

To enable bridged networking, open the **Settings** dialog of a virtual machine, go to the **Network** page and select **Bridged Network** in the drop-down list for the **Attached To** field. Select a host interface from the list at the bottom of the page, which contains the physical network interfaces of your systems. On a typical MacBook, for example, this will allow you to select between en1: AirPort, which is the wireless interface, and en0: Ethernet, which represents the interface with a network cable.

**Note:** Bridging to a wireless interface is done differently from bridging to a wired interface, because most wireless adapters do not support promiscuous mode. All traffic has to use the MAC address of the host's wireless adapter, and therefore Oracle VM VirtualBox needs to replace the source MAC address in the Ethernet header of an outgoing packet to make sure the reply will be sent to the host interface. When Oracle VM VirtualBox sees an incoming packet with a destination IP address that belongs to one of the virtual machine adapters it replaces the destination MAC address in the Ethernet header with the VM adapter's MAC address and passes it on. Oracle VM VirtualBox examines ARP and DHCP packets in order to learn the IP addresses of virtual machines.

Depending on your host operating system, the following limitations apply:

- **Mac OS X hosts.** Functionality is limited when using AirPort, the Mac's wireless networking system, for bridged networking. Currently, Oracle VM VirtualBox supports only IPv4 and IPv6 over AirPort. For other protocols, such as IPX, you must choose a wired interface.
- **Linux hosts.** Functionality is limited when using wireless interfaces for bridged networking. Currently, Oracle VM VirtualBox supports only IPv4 and IPv6 over wireless. For other protocols, such as IPX, you must choose a wired interface.

  Also, setting the MTU to less than 1500 bytes on wired interfaces provided by the sky2 driver on the Marvell Yukon II EC Ultra Ethernet NIC is known to cause packet losses under certain conditions.

  Some adapters strip VLAN tags in hardware. This does not allow you to use VLAN trunking between VM and the external network with pre-2.6.27 Linux kernels, or with host operating systems other than Linux.
- **Oracle Solaris hosts.** There is no support for using wireless interfaces. Filtering guest traffic using IPFilter is also not completely supported due to technical restrictions of the Oracle Solaris networking subsystem. These issues may be addressed in later releases of Oracle Solaris 11.

  On Oracle Solaris 11 hosts build 159 and above, it is possible to use Oracle Solaris Crossbow Virtual Network Interfaces (VNICs) directly with Oracle VM VirtualBox without any additional configuration other than each VNIC must be exclusive for every guest network interface.

  When using VLAN interfaces with Oracle VM VirtualBox, they must be named according to the PPA-hack naming scheme, such as e1000g513001. Otherwise, the guest may receive packets in an unexpected format.

## 6.6 Internal Networking

Internal Networking is similar to bridged networking in that the VM can directly communicate with the outside world. However, the outside world is limited to other VMs on the same host which connect to the same internal network.

Even though technically, everything that can be done using internal networking can also be done using bridged networking, there are security advantages with internal networking. In bridged networking mode, all traffic goes through a physical interface of the host system. It is therefore possible to attach a packet sniffer such as Wireshark to the host interface and log all traffic that goes over it. If, for any reason, you prefer two or more VMs on the same machine to communicate privately, hiding their data from both the host system and the user, bridged networking therefore is not an option.

Internal networks are created automatically as needed. There is no central configuration. Every internal network is identified simply by its name. Once there is more than one active

virtual network card with the same internal network ID, the Oracle VM VirtualBox support driver will automatically *wire* the cards and act as a network switch. The Oracle VM VirtualBox support driver implements a complete Ethernet switch and supports both broadcast/multicast frames and promiscuous mode.

In order to attach a VM's network card to an internal network, set its networking mode to Internal Networking. There are two ways to accomplish this:

- Use the VM's **Settings** dialog in the Oracle VM VirtualBox graphical user interface. In the **Networking** category of the settings dialog, select **Internal Networking** from the drop-down list of networking modes. Select the name of an existing internal network from the drop-down list below, or enter a new name into the **Name** field.
- Use the command line, for example:

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

  Optionally, you can specify a network name with the command:

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

  If you do not specify a network name, the network card will be attached to the network `intnet` by default.

Unless you configure the virtual network cards in the guest operating systems that are participating in the internal network to use static IP addresses, you may want to use the DHCP server that is built into Oracle VM VirtualBox to manage IP addresses for the internal network. See chapter 8.39, *VBoxManage dhcpserver*, page 193.

As a security measure, by default, the Linux implementation of internal networking only allows VMs running under the same user ID to establish an internal network. However, it is possible to create a shared internal networking interface, accessible by users with different user IDs.

## 6.7 Host-Only Networking

Host-only networking is another networking mode that was added with version 2.2 of Oracle VM VirtualBox. It can be thought of as a hybrid between the bridged and internal networking modes. As with bridged networking, the virtual machines can talk to each other and the host as if they were connected through a physical Ethernet switch. As with internal networking, a physical networking interface need not be present, and the virtual machines cannot talk to the world outside the host since they are not connected to a physical networking interface.

When host-only networking is used, Oracle VM VirtualBox creates a new software interface on the host which then appears next to your existing network interfaces. In other words, whereas with bridged networking an existing physical interface is used to attach virtual machines to, with host-only networking a new *loopback* interface is created on the host. And whereas with internal networking, the traffic between the virtual machines cannot be seen, the traffic on the loopback interface on the host can be intercepted.

Host-only networking is particularly useful for preconfigured virtual appliances, where multiple virtual machines are shipped together and designed to cooperate. For example, one virtual machine may contain a web server and a second one a database, and since they are intended to talk to each other, the appliance can instruct Oracle VM VirtualBox to set up a host-only network for the two. A second, bridged, network would then connect the web server to the outside world to serve data to, but the outside world cannot connect to the database.

To change a virtual machine's virtual network interface to Host Only mode, do either of the following:

- Go to the **Network** page in the virtual machine's **Settings** dialog and select **Host-Only Networking**.
- On the command line, enter `VBoxManage modifyvm "VM name" --nic<x> hostonly`. See chapter 8.8, *VBoxManage modifyvm*, page 135.

Before you can attach a VM to a host-only network you have to create at least one host-only interface. You can use the GUI for this. Choose **File**, **Preferences**, **Network**, **Host-Only Network**, **(+)Add Host-Only Network**.

Alternatively, you can use the command line:

```
VBoxManage hostonlyif create
```

See chapter 8.38, *VBoxManage hostonlyif*, page 192.

For host-only networking, as with internal networking, you may find the DHCP server useful that is built into Oracle VM VirtualBox. This can be enabled to then manage the IP addresses in the host-only network since otherwise you would need to configure all IP addresses statically.

- In the Oracle VM VirtualBox graphical user interface, you can configure all these items in the global settings by choosing **File**, **Preferences**, **Network**. This lists all host-only networks which are presently in use. Click on the network name and then on **Edit**. You can then modify the adapter and DHCP settings.
- Alternatively, you can use `VBoxManage dhcpserver` on the command line. See chapter 8.39, *VBoxManage dhcpserver*, page 193.

**Note:** On Linux and Mac OS X hosts the number of host-only interfaces is limited to 128. There is no such limit for Oracle Solaris and Windows hosts.

## 6.8 UDP Tunnel Networking

This networking mode enables you to interconnect virtual machines running on different hosts.

Technically this is done by encapsulating Ethernet frames sent or received by the guest network card into UDP/IP datagrams, and sending them over any network available to the host.

UDP Tunnel mode has the following parameters:

- **Source UDP port:** The port on which the host listens. Datagrams arriving on this port from any source address will be forwarded to the receiving part of the guest network card.
- **Destination address:** IP address of the target host of the transmitted data.
- **Destination UDP port:** Port number to which the transmitted data is sent.

When interconnecting two virtual machines on two different hosts, their IP addresses must be swapped. On a single host, source and destination UDP ports must be swapped.

In the following example, host 1 uses the IP address 10.0.0.1 and host 2 uses IP address 10.0.0.2. To configure using the command-line:

```
VBoxManage modifyvm "VM 01 on host 1" --nic<x> generic
VBoxManage modifyvm "VM 01 on host 1" --nicgenericdrv<x> UDPTunnel
VBoxManage modifyvm "VM 01 on host 1" --nicproperty<x> dest=10.0.0.2
VBoxManage modifyvm "VM 01 on host 1" --nicproperty<x> sport=10001
VBoxManage modifyvm "VM 01 on host 1" --nicproperty<x> dport=10002
```

```
VBoxManage modifyvm "VM 02 on host 2" --nic<y> generic
VBoxManage modifyvm "VM 02 on host 2" --nicgenericdrv<y> UDPTunnel
VBoxManage modifyvm "VM 02 on host 2" --nicproperty<y> dest=10.0.0.1
VBoxManage modifyvm "VM 02 on host 2" --nicproperty<y> sport=10002
VBoxManage modifyvm "VM 02 on host 2" --nicproperty<y> dport=10001
```

Of course, you can always interconnect two virtual machines on the same host, by setting the destination address parameter to 127.0.0.1 on both. It will act similarly to an internal network in this case. However, the host can see the network traffic which it could not in the normal internal network case.

**Note:** On UNIX-based hosts, such as Linux, Oracle Solaris, and Mac OS X, it is not possible to bind to ports below 1024 from applications that are not run by `root`. As a result, if you try to configure such a source UDP port, the VM will refuse to start.

## 6.9 VDE Networking

Virtual Distributed Ethernet (VDE) is a flexible, virtual network infrastructure system, spanning across multiple hosts in a secure way. It enables L2/L3 switching, including spanning-tree protocol, VLANs, and WAN emulation. It is an optional part of Oracle VM VirtualBox which is only included in the source code.

VDE is a project developed by Renzo Davoli, Associate Professor at the University of Bologna, Italy.

The basic building blocks of the infrastructure are VDE switches, VDE plugs, and VDE wires which interconnect the switches.

The Oracle VM VirtualBox VDE driver has a single parameter: VDE network. This is the name of the VDE network switch socket to which the VM will be connected.

The following basic example shows how to connect a virtual machine to a VDE switch.

1. Create a VDE switch:

   ```
   vde_switch -s /tmp/switch1
   ```

2. Configure VMs using the command-line:

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

   ```
   VBoxManage modifyvm "VM name" --nicgenericdrv<x> VDE
   ```

   To connect to an automatically allocated switch port:

   ```
   VBoxManage modifyvm "VM name" --nicproperty<x> network=/tmp/switch1
   ```

   To connect to a specific switch port *n*:

   ```
   VBoxManage modifyvm "VM name" --nicproperty<x> network=/tmp/switch1[<n>]
   ```

   This command can be useful for VLANs.

3. (Optional) Map between a VDE switch port and a VLAN.

   Using the switch command line:

   ```
   vde$ vlan/create <VLAN>
   ```

```
vde$ port/setvlan <port> <VLAN>
```

VDE is available on Linux and FreeBSD hosts only. It is only available if the VDE software and the VDE plugin library from the VirtualSquare project are installed on the host system.

> **Note:** For Linux hosts, the shared library libvdeplug.so must be available in the search path for shared libraries.

For more information on setting up VDE networks, please see the documentation accompanying the software. See also http://wiki.virtualsquare.org/wiki/index.php/VDE_Basic_Networking.

## 6.10 Limiting Bandwidth for Network Input/Output

Oracle VM VirtualBox supports limiting of the maximum bandwidth used for network transmission. Several network adapters of one VM may share limits through bandwidth groups. It is possible to have more than one such limit.

> **Note:** Oracle VM VirtualBox shapes VM traffic only in the transmit direction, delaying the packets being sent by virtual machines. It does not limit the traffic being received by virtual machines.

Limits are configured through `VBoxManage`. The following example creates a bandwidth group named Limit, sets the limit to 20 Mbps and assigns the group to the first and second adapters of the VM:

```
VBoxManage bandwidthctl "VM name" add Limit --type network --limit 20m
VBoxManage modifyvm "VM name" --nicbandwidthgroup1 Limit
VBoxManage modifyvm "VM name" --nicbandwidthgroup2 Limit
```

All adapters in a group share the bandwidth limit, meaning that in the example above the bandwidth of both adapters combined can never exceed 20 Mbps. However, if one adapter does not 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 previous example to 100 Kbps:

```
VBoxManage bandwidthctl "VM name" set Limit --limit 100k
```

To completely disable shaping for the first adapter of VM use the following command:

```
VBoxManage modifyvm "VM name" --nicbandwidthgroup1 none
```

It is also possible to disable shaping for all adapters assigned to a bandwidth group while VM is running, by specifying the zero limit for the group. For example, for the bandwidth group named Limit:

```
VBoxManage bandwidthctl "VM name" set Limit --limit 0
```

## 6.11 Improving Network Performance

Oracle VM VirtualBox provides a variety of virtual network adapters that can be attached to the host's network in a number of ways. Depending on which types of adapters and attachments are used the network performance will be different. Performance-wise the virtio network adapter is preferable over Intel PRO/1000 emulated adapters, which are preferred over the PCNet family of adapters. Both virtio and Intel PRO/1000 adapters enjoy the benefit of segmentation and checksum offloading. Segmentation offloading is essential for high performance as it allows for less context switches, dramatically increasing the sizes of packets that cross the VM/host boundary.

> **Note:** Neither virtio nor Intel PRO/1000 drivers for Windows XP support segmentation offloading. Therefore Windows XP guests never reach the same transmission rates as other guest types. Refer to MS Knowledge base article 842264 for additional information.

Three attachment types: Internal, Bridged, and Host-Only, have nearly identical performance. The Internal type is a little bit faster and uses less CPU cycles as the packets never reach the host's network stack. The NAT attachment type is the slowest and most secure of all attachment types, as it provides network address translation. The generic driver attachment is special and cannot be considered as an alternative to other attachment types.

The number of CPUs assigned to VM does not improve network performance and in some cases may hurt it due to increased concurrency in the guest.

Here is a short summary of things to check in order to improve network performance:

- Whenever possible use the virtio network adapter. Otherwise, use one of the Intel PRO/1000 adapters.
- Use a Bridged attachment instead of NAT.
- Make sure segmentation offloading is enabled in the guest OS. Usually it will be enabled by default. You can check and modify offloading settings using the `ethtool` command on Linux guests.
- Perform a full detailed analysis of network traffic on the VM's network adaptor using a third party tool such as Wireshark. To do this, a promiscuous mode policy needs to be used on the VM's network adaptor. Use of this mode is only possible on the following network types: NAT Network, Bridged Adapter, Internal Network, and Host-Only Adapter.

  To setup a promiscuous mode policy, either select from the drop down list located in the **Network Settings** dialog for the network adaptor or use the command line tool `VBoxManage`. See chapter 8.8, *VBoxManage modifyvm*, page 135.

  Promiscuous mode policies are as follows:

  - `deny`, which hides any traffic not intended for the VM's network adaptor. This is the default setting.
  - `allow-vms`, which hides all host traffic from the VM's network adaptor, but allows it to see traffic from and to other VMs.
  - `allow-all`, which removes all restrictions. The VM's network adaptor sees all traffic.

# 7 Remote Virtual Machines

## 7.1 Remote Display (VRDP Support)

Oracle VM VirtualBox can display virtual machines remotely, meaning that a virtual machine can execute on one computer even though the machine will be displayed on a second computer, and the machine will be controlled from there as well, as if the virtual machine was running on that second computer.

For maximum flexibility, Oracle VM VirtualBox implements remote machine display through a generic extension interface called the VirtualBox Remote Desktop Extension (VRDE). The base open source Oracle VM VirtualBox package only provides this interface, while implementations can be supplied by third parties with Oracle VM VirtualBox extension packages, which must be installed separately from the base package. See chapter 1.6, *Installing Oracle VM VirtualBox and Extension Packs*, page 6.

Oracle provides support for the VirtualBox Remote Display Protocol (VRDP) in such an Oracle VM VirtualBox extension package. When this package is installed, Oracle VM VirtualBox versions 4.0 and later support VRDP the same way as binary, non-open source, versions of Oracle VM VirtualBox before 4.0 did.

VRDP is a backwards-compatible extension to Microsoft's Remote Desktop Protocol (RDP). As a result, you can use any standard RDP client to control the remote VM.

Even when the extension is installed, the VRDP server is disabled by default. It can easily be enabled on a per-VM basis either in the VirtualBox Manager in the **Display** settings, see chapter 3.6, *Display Settings*, page 50, or with the `VBoxManage` command, as follows:

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

By default, the VRDP server uses TCP port `3389`. You will need to change the default port if you run more than one VRDP server, since the port can only be used by one server at a time. You might also need to change it on Windows hosts since the default port might already be used by the RDP server that is built into Windows itself. Ports 5000 through 5050 are typically not used and might be a good choice.

The port can be changed either in the **Display** settings of the graphical user interface or with the `--vrdeport` option of the `VBoxManage modifyvm` command. You can specify a comma-separated list of ports or ranges of ports. Use a dash between two port numbers to specify a range. The VRDP server will bind to *one* of the available ports from the specified list. For example, `VBoxManage modifyvm "VM name" --vrdeport 5000,5010-5012` will configure the server to bind to one of the ports 5000, 5010, 5011, or 5012. See chapter 8.8.5, *Remote Machine Settings*, page 143.

The actual port used by a running VM can be either queried with the `VBoxManage showvminfo` command or seen in the GUI on the **Runtime** tab of the **Session Information** dialog, which is accessible from the **Machine** menu of the VM window.

Support for IPv6 has been implemented in Oracle VM VirtualBox 4.3. If the host OS supports IPv6 the VRDP server will automatically listen for IPv6 connections in addition to IPv4.

### 7.1.1 Common Third-Party RDP Viewers

Since VRDP is backwards-compatible to RDP, you can use any standard RDP viewer to connect to such a remote virtual machine. For this to work, you must specify the IP address of your *host*

system, not of the virtual machine, as the server address to connect to. You must also specify the port number that the VRDP server is using.

The following examples are for the most common RDP viewers:

- On Windows, you can use the Microsoft Terminal Services Connector, `mstsc.exe`, that is included with Windows. Press the Windows key + R, to display the **Run** dialog. Enter `mstsc` to start the program. You can also find the program in **Start**, **All Programs**, **Accessories**, **Remote Desktop Connection**. If you use the **Run** dialog, you can enter options directly. For example:

  ```
  mstsc 1.2.3.4:3389
  ```

  Replace `1.2.3.4` with the host IP address, and `3389` with a different port, if necessary.

  **Note:**

  - IPv6 addresses must be enclosed in square brackets to specify a port. For example: `mstsc [fe80::1:2:3:4]:3389`
  - When connecting to localhost in order to test the connection, the addresses `localhost` and `127.0.0.1` might not work using `mstsc.exe`. Instead, the address `127.0.0.2[:3389]` has to be used.

- On other systems, you can use the standard open source `rdesktop` program. This ships with most Linux distributions, but Oracle VM VirtualBox also comes with a modified variant of `rdesktop` for remote USB support. See chapter 7.1.4, *Remote USB*, page 114.

  With `rdesktop`, use a command line such as the following:

  ```
  rdesktop -a 16 -N 1.2.3.4:3389
  ```

  Replace `1.2.3.4` with the host IP address, and `3389` with a different port, if necessary. The `-a 16` option requests a color depth of 16 bits per pixel, which we recommend. For best performance, after installation of the guest operating system, you should set its display color depth to the same value. The `-N` option enables use of the NumPad keys.

- You can use the Remmina remote desktop client with VRDP. This application is included with some Linux distributions, such as Debian and Ubuntu.

- If you run the KDE desktop, you can use `krdc`, the KDE RDP viewer. A typical command line is as follows:

  ```
  krdc rdp://1.2.3.4:3389
  ```

  Replace `1.2.3.4` with the host IP address, and `3389` with a different port, if necessary. The "rdp://" prefix is required with krdc to switch it into RDP mode.

- With Sun Ray thin clients you can use `uttsc`, which is part of the Sun Ray Windows Connector package. See the Sun Ray documentation for details.

### 7.1.2 VBoxHeadless, the Remote Desktop Server

While any VM started from the VirtualBox Manager is capable of running virtual machines remotely, it is not convenient to have to run the full-fledged GUI if you never want to have VMs displayed locally in the first place. In particular, if you are running server hardware whose only purpose is to host VMs, and all your VMs are supposed to run remotely over VRDP, then it is pointless to have a graphical user interface on the server at all. This is especially true for Linux or Oracle Solaris hosts, as the VirtualBox Manager comes with dependencies on the Qt and SDL libraries. This is inconvenient if you would rather not have the X Window system on your server at all.

Oracle VM VirtualBox therefore comes with a front-end called `VBoxHeadless`, which produces no visible output on the host at all, but still can deliver VRDP data. This front-end has no dependencies on the X Window system on Linux and Oracle Solaris hosts.

> **Note:** Before Oracle VM VirtualBox 1.6, the headless server was called `VBoxVRDP`. For the sake of backwards compatibility, the Oracle VM VirtualBox installation still installs an executable with that name as well.

To start a virtual machine with `VBoxHeadless`, you have the following options:

- Use the `VBoxManage` command, as follows:

  ```
  VBoxManage startvm "VM name" --type headless
  ```

  The `--type` option causes Oracle VM VirtualBox to use `VBoxHeadless` as the front-end to the internal virtualization engine, instead of the Qt front-end.

- Use the `VBoxHeadless` command, as follows:

  ```
  VBoxHeadless --startvm <uuid|name>
  ```

  This way of starting the VM helps troubleshooting problems reported by `VBoxManage startvm`, because you can sometimes see more detailed error messages, especially for early failures before the VM execution is started. In normal situations `VBoxManage startvm` is preferred, since it runs the VM directly as a background process which has to be done explicitly when directly starting with `VBoxHeadless`.

- Start `VBoxHeadless` from the VirtualBox Manager GUI, by pressing the Shift key when starting a virtual machine or by selecting **Headless Start** from the **Machine** menu.

When you use the `VBoxHeadless` command to start a VM, the VRDP server will be enabled according to the VM configuration. You can override the VM's setting using `--vrde` command line parameter. To enable the VRDP server, start the VM as follows:

```
VBoxHeadless --startvm <uuid|name> --vrde on
```

To disable the VRDP server:

```
VBoxHeadless --startvm <uuid|name> --vrde off
```

To have the VRDP server enabled depending on the VM configuration, as for other front-ends:

```
VBoxHeadless --startvm <uuid|name> --vrde config
```

This command is the same as the following:

```
VBoxHeadless --startvm <uuid|name>
```

If you start the VM with `VBoxManage startvm` then the configuration settings of the VM are always used.

### 7.1.3 Step by Step: Creating a Virtual Machine on a Headless Server

The following instructions describe how to create a virtual machine on a headless server over a network connection. This example creates a virtual machine, establishes an RDP connection and installs a guest operating system. All of these tasks are done without having to touch the headless server. You need the following prerequisites:

- Oracle VM VirtualBox on a server machine with a supported host operating system. The Oracle VM VirtualBox Extension Pack for the VRDP server must be installed, see chapter 7.1, *Remote Display (VRDP Support)*, page 110. The procedures assume a Linux server is used.
- An ISO file accessible from the server, containing the installation data for the guest operating system to install. Windows XP is used in the example.
- A terminal connection to that host through which you can access a command line, such as `ssh`.
- An RDP viewer on the remote client. See chapter 7.1.1, *Common Third-Party RDP Viewers*, page 110 for examples.

Note that on the server machine, since we will only use the headless server, Qt and the X Window system are not required.

1. On the headless server, create a new virtual machine. For example:

   ```
   VBoxManage createvm --name "Windows XP" --ostype WindowsXP --register
   ```

   If you do not specify `--register`, you will have to manually use the `registervm` command later.

   You do not need to specify `--ostype`, but doing so selects some sensible default values for certain VM parameters. For example, the RAM size and the type of the virtual network device. To get a complete list of supported operating systems you can use the following command:

   ```
   VBoxManage list ostypes
   ```

2. Make sure the settings for the VM are appropriate for the guest operating system that we will install. For example:

   ```
   VBoxManage modifyvm "Windows XP" --memory 256 --acpi on --boot1 dvd --nic1 nat
   ```

3. Create a virtual hard disk for the VM. For example, to create a 10 GB virtual hard disk:

   ```
   VBoxManage createhd --filename "WinXP.vdi" --size 10000
   ```

4. Add an IDE Controller to the new VM. For example:

   ```
   VBoxManage storagectl "Windows XP" --name "IDE Controller"
     --add ide --controller PIIX4
   ```

5. Set the VDI file you created as the first virtual hard disk of the new VM. For example:

   ```
   VBoxManage storageattach "Windows XP" --storagectl "IDE Controller"
    --port 0 --device 0 --type hdd --medium "WinXP.vdi"
   ```

6. Attach the ISO file that contains the operating system installation that you want to install later to the virtual machine. This is done so that the VM can boot from it.

```
VBoxManage storageattach "Windows XP" --storagectl "IDE Controller"
 --port 0 --device 1 --type dvddrive --medium /full/path/to/iso.iso
```

7. Enable the VirtualBox Remote Desktop Extension, the VRDP server, as follows:

```
VBoxManage modifyvm "Windows XP" --vrde on
```

8. Start the virtual machine using the `VBoxHeadless` command:

```
VBoxHeadless --startvm "Windows XP"
```

   If the configuration steps worked, you should see a copyright notice. If you are returned to the command line, then something did not work correctly.

9. On the client machine, start the RDP viewer and connect to the server. See chapter 7.1.1, *Common Third-Party RDP Viewers*, page 110 for details of how to use various common RDP viewers.

   The installation routine of your guest operating system should be displayed in the RDP viewer.

### 7.1.4 Remote USB

As a special feature additional to the VRDP support, Oracle VM VirtualBox also supports remote USB devices over the wire. That is, an Oracle VM VirtualBox guest that runs on one computer can access the USB devices of the remote computer on which the VRDP data is being displayed the same way as USB devices that are connected to the actual host. This enables running of virtual machines on an Oracle VM VirtualBox host that acts as a server, where a client can connect from elsewhere that needs only a network adapter and a display capable of running an RDP viewer. When USB devices are plugged into the client, the remote Oracle VM VirtualBox server can access them.

For these remote USB devices, the same filter rules apply as for other USB devices. See chapter 3.11.1, *USB Settings*, page 56. All you have to do is specify Remote, or Any, when setting up these rules.

Accessing remote USB devices is only possible if the RDP client supports this extension. On Linux and Oracle Solaris hosts, the Oracle VM VirtualBox installation provides a suitable VRDP client called `rdesktop-vrdp`. Recent versions of `uttsc`, a client tailored for the use with Sun Ray thin clients, also support accessing remote USB devices. RDP clients for other platforms will be provided in future Oracle VM VirtualBox versions.

To make a remote USB device available to a VM, `rdesktop-vrdp` should be started as follows:

```
rdesktop-vrdp -r usb -a 16 -N my.host.address
```

See chapter 12.8.7, *USB Not Working*, page 293 for further details on how to properly set up the permissions for USB devices. Furthermore it is advisable to disable automatic loading of any host driver on the remote host which might work on USB devices to ensure that the devices are accessible by the RDP client. If the setup was properly done on the remote host, plug and unplug events are visible in the VBox.log file of the VM.

### 7.1.5 RDP Authentication

For each virtual machine that is remotely accessible using RDP, you can individually determine if and how client connections are authenticated. For this, use the `VBoxManage modifyvm` command with the `--vrdeauthtype` option. See chapter 8.8, *VBoxManage modifyvm*, page 135. The following methods of authentication are available:

- The **null** method means that there is no authentication at all. Any client can connect to the VRDP server and thus the virtual machine. This is very insecure and only to be recommended for private networks.
- The **external** method provides external authentication through a special authentication library. Oracle VM VirtualBox ships with two special authentication libraries:
  1. The default authentication library, `VBoxAuth`, authenticates against user credentials of the hosts. Depending on the host platform, this means the following:
     - On Linux hosts, `VBoxAuth.so` authenticates users against the host's PAM system.
     - On Windows hosts, `VBoxAuth.dll` authenticates users against the host's WinLogon system.
     - On Mac OS X hosts, `VBoxAuth.dylib` authenticates users against the host's directory service.

     In other words, the external method by default performs authentication with the user accounts that exist on the host system. Any user with valid authentication credentials is accepted. For example, the username does not have to correspond to the user running the VM.
  2. An additional library called `VBoxAuthSimple` performs authentication against credentials configured in the "extradata" section of a virtual machine's XML settings file. This is probably the simplest way to get authentication that does not depend on a running and supported guest. The following steps are required:
     a) Enable `VBoxAuthSimple` with the following command:

     ```
     VBoxManage setproperty vrdeauthlibrary "VBoxAuthSimple"
     ```

     b) To enable the library for a particular VM, you must switch authentication to external, as follows:

     ```
     VBoxManage modifyvm "VM name" --vrdeauthtype external
     ```

     Replace `<vm>` with the VM name or UUID.

     c) You then need to configure users and passwords by writing items into the machine's extradata. Since the XML machine settings file, into whose `extradata` section the password needs to be written, is a plain text file, Oracle VM VirtualBox uses hashes to encrypt passwords. The following command must be used:

     ```
     VBoxManage setextradata "VM name" "VBoxAuthSimple/users/<user>" <hash>
     ```

     Replace `<vm>` with the VM name or UUID, `<user>` with the user name who should be allowed to log in and `<hash>` with the encrypted password. As an example, to obtain the hash value for the password `secret`, you can use the following command:

     ```
     VBoxManage internalcommands passwordhash "secret"
     ```

     This command will generate output similar to the following:

     ```
     2bb80d537b1da3e38bd30361aa855686bde0eacd7162fef6a25fe97bf527a25b
     ```

     You then use `VBoxManage setextradata` to store this value in the machine's `extradata` section.

     As a combined example, to set the password for the user `john` and the machine `My VM` to `secret`, use this command:

```
VBoxManage setextradata "My VM" "VBoxAuthSimple/users/john"
    2bb80d537b1da3e38bd30361aa855686bde0eacd7162fef6a25fe97bf527a25b
```

- The **guest** authentication method performs authentication with a special component that comes with the Guest Additions. As a result, authentication is not performed on the host, but with the guest user accounts.

  This method is currently still in testing and not yet supported.

In addition to the methods described above, you can replace the default external authentication module with any other module. For this, Oracle VM VirtualBox provides a well-defined interface that enables you to write your own authentication module. This is described in detail in the Oracle VM VirtualBox Software Development Kit (SDK) reference. See chapter 11, *Oracle VM VirtualBox Programming Interfaces*, page 276.

### 7.1.6 RDP Encryption

RDP features data stream encryption, which is based on the RC4 symmetric cipher, with keys up to 128-bit. The RC4 keys are replaced at regular intervals, every 4096 packets.

RDP provides the following different authentication methods:

- **RDP4** authentication was used historically. With RDP4, the RDP client does not perform any checks in order to verify the identity of the server it connects to. Since user credentials can be obtained using a man in the middle (MITM) attack, RDP4 authentication is insecure and should generally not be used.
- **RDP5.1** authentication employs a server certificate for which the client possesses the public key. This way it is guaranteed that the server possess the corresponding private key. However, as this hard-coded private key became public some years ago, RDP5.1 authentication is also insecure.
- **RDP5.2** authentication uses Enhanced RDP Security, which means that an external security protocol is used to secure the connection. RDP4 and RDP5.1 use Standard RDP Security. The VRDP server supports Enhanced RDP Security with TLS protocol and, as a part of TLS handshake, sends the server certificate to the client.

  The `Security/Method` VRDE property sets the desired security method, which is used for a connection. Valid values are as follows:

  - **Negotiate.** Both Enhanced (TLS) and Standard RDP Security connections are allowed. The security method is negotiated with the client. This is the default setting.
  - **RDP.** Only Standard RDP Security is accepted.
  - **TLS.** Only Enhanced RDP Security is accepted. The client must support TLS.

    The OpenSSL library version determines which versions of TLS are supported. The Oracle VM VirtualBox clients include at least Version 1.1.0 of the OpenSSL library. This library supports TLS versions 1.0, 1.1, and 1.2. Some clients might include newer versions of the OpenSSL library and thus support additional TLS versions.

  For example, the following command enables a client to use either Standard or Enhanced RDP Security connection:

  ```
  vboxmanage modifyvm "VM name" --vrdeproperty "Security/Method=negotiate"
  ```

  If the `Security/Method` property is set to either Negotiate or TLS, the TLS protocol will be automatically used by the server, if the client supports TLS. However, in order to use TLS the server must possess the Server Certificate, the Server Private Key and the Certificate Authority (CA) Certificate. The following example shows how to generate a server certificate.

1. Create a CA self signed certificate.

   ```
   openssl req -new -x509 -days 365 -extensions v3_ca \
     -keyout ca_key_private.pem -out ca_cert.pem
   ```

2. Generate a server private key and a request for signing.

   ```
   openssl genrsa -out server_key_private.pem
   openssl req -new -key server_key_private.pem -out server_req.pem
   ```

3. Generate the server certificate.

   ```
   openssl x509 -req -days 365 -in server_req.pem \
     -CA ca_cert.pem -CAkey ca_key_private.pem -set_serial 01 -out server_cert.pem
   ```

The server must be configured to access the required files. For example:

```
vboxmanage modifyvm "VM name" \
  --vrdeproperty "Security/CACertificate=path/ca_cert.pem"

vboxmanage modifyvm "VM name" \
  --vrdeproperty "Security/ServerCertificate=path/server_cert.pem"

vboxmanage modifyvm "VM name" \
  --vrdeproperty "Security/ServerPrivateKey=path/server_key_private.pem"
```

As the client that connects to the server determines what type of encryption will be used, with `rdesktop`, the Linux RDP viewer, use the `-4` or `-5` options.

### 7.1.7 Multiple Connections to the VRDP Server

The VRDP server of Oracle VM VirtualBox supports multiple simultaneous connections to the same running VM from different clients. All connected clients see the same screen output and share a mouse pointer and keyboard focus. This is similar to several people using the same computer at the same time, taking turns at the keyboard.

The following command enables multiple connection mode:

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

### 7.1.8 Multiple Remote Monitors

To access two or more remote VM displays you have to enable the VRDP multiconnection mode. See chapter 7.1.7, *Multiple Connections to the VRDP Server*, page 117.

The RDP client can select the virtual monitor number to connect to using the `domain` login parameter (`-d`). If the parameter ends with `@` followed by a number, Oracle VM VirtualBox interprets this number as the screen index. The primary guest screen is selected with `@1`, the first secondary screen is `@2`, and so on.

The Microsoft RDP6 client does not let you specify a separate domain name. Instead, enter `domain\username` in the **Username** field. For example, `@2\name`. `name` must be supplied, and must be the name used to log in if the VRDP server is set up to require credentials. If it is not, you may use any text as the username.

### 7.1.9 VRDP Video Redirection

The VRDP server can redirect video streams from the guest to the RDP client. Video frames are compressed using the JPEG algorithm allowing a higher compression ratio than standard RDP bitmap compression methods. It is possible to increase the compression ratio by lowering the video quality.

The VRDP server automatically detects video streams in a guest as frequently updated rectangular areas. As a result, this method works with any guest operating system without having to install additional software in the guest. In particular, the Guest Additions are not required.

On the client side, however, currently only the Windows 7 Remote Desktop Connection client supports this feature. If a client does not support video redirection, the VRDP server falls back to regular bitmap updates.

The following command enables video redirection:

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

The quality of the video is defined as a value from 10 to 100 percent, representing a JPEG compression level, where lower numbers mean lower quality but higher compression. The quality can be changed using the following command:

```
VBoxManage modifyvm "VM name" --vrdevideochannelquality 75
```

### 7.1.10 VRDP Customization

With Oracle VM VirtualBox it is possible to disable display output, mouse and keyboard input, audio, remote USB, or clipboard individually in the VRDP server.

The following commands change the corresponding server settings:

```
VBoxManage modifyvm "VM name" --vrdeproperty Client/DisableDisplay=1
VBoxManage modifyvm "VM name" --vrdeproperty Client/DisableInput=1
VBoxManage modifyvm "VM name" --vrdeproperty Client/DisableUSB=1
VBoxManage modifyvm "VM name" --vrdeproperty Client/DisableAudio=1
VBoxManage modifyvm "VM name" --vrdeproperty Client/DisableClipboard=1
VBoxManage modifyvm "VM name" --vrdeproperty Client/DisableUpstreamAudio=1
```

To reenable a feature, use a similar command without the trailing 1. For example:

```
VBoxManage modifyvm "VM name" --vrdeproperty Client/DisableDisplay=
```

## 7.2 Teleporting

Oracle VM VirtualBox supports *teleporting*. Teleporting is moving a virtual machine over a network from one Oracle VM VirtualBox host to another, while the virtual machine is running. This works regardless of the host operating system that is running on the hosts. You can teleport virtual machines between Oracle Solaris and Mac OS X hosts, for example.

Teleporting requires that a machine be currently running on one host, which is called the *source*. The host to which the virtual machine will be teleported is called the *target*. The machine on the target is then configured to wait for the source to contact the target. The machine's running state will then be transferred from the source to the target with minimal downtime.

Teleporting happens over any TCP/IP network. The source and the target only need to agree on a TCP/IP port which is specified in the teleporting settings.

At this time, there are a few prerequisites for this to work, as follows:

- On the target host, you must configure a virtual machine in Oracle VM VirtualBox with exactly the same hardware settings as the machine on the source that you want to teleport. This does not apply to settings which are merely descriptive, such as the VM name, but obviously for teleporting to work, the target machine must have the same amount of memory and other hardware settings. Otherwise teleporting will fail with an error message.

- The two virtual machines on the source and the target must share the same storage, hard disks as well as floppy disks and CD/DVD images. This means that they either use the same iSCSI targets or that the storage resides somewhere on the network and both hosts have access to it using NFS or SMB/CIFS.

  This also means that neither the source nor the target machine can have any snapshots.

To configure teleporting, perform the following steps:

1. On the *target* host, configure the virtual machine to wait for a teleport request to arrive when it is started, instead of actually attempting to start the machine. This is done with the following `VBoxManage` command:

   ```
   VBoxManage modifyvm <targetvmname> --teleporter on --teleporterport <port>
   ```

   where `<targetvmname>` is the name of the virtual machine on the target host and `<port>` is a TCP/IP port number to be used on both the source and the target hosts. For example, use 6000. See chapter 8.8.6, *Teleporting Settings*, page 146.

2. Start the VM on the target host. Instead of running, the VM shows a progress dialog, indicating that it is waiting for a teleport request to arrive.

3. Start the VM on the *source* host as usual. When it is running and you want it to be teleported, issue the following command on the source host:

   ```
   VBoxManage controlvm <sourcevmname> teleport --host <targethost> --port <port>
   ```

   where `<sourcevmname>` is the name of the virtual machine on the source host, which is the machine that is currently running. `<targethost>` is the host or IP name of the target host on which the machine is waiting for the teleport request, and `<port>` must be the same number as specified in the command on the target host. See chapter 8.14, *VBoxManage controlvm*, page 152.

For testing, you can also teleport machines on the same host. In that case, use localhost as the hostname on both the source and the target host.

**Note:** In rare cases, if the CPUs of the source and the target are very different, teleporting can fail with an error message, or the target may hang. This may happen especially if the VM is running application software that is highly optimized to run on a particular CPU without correctly checking that certain CPU features are actually present. Oracle VM VirtualBox filters what CPU capabilities are presented to the guest operating system. Advanced users can attempt to restrict these virtual CPU capabilities with the `VBoxManage modifyvm --cpuid` command. See chapter 8.8.6, *Teleporting Settings*, page 146.

# 8 VBoxManage

## 8.1 Introduction

As briefly mentioned in chapter 1.17, *Alternative Front-Ends*, page 28, `VBoxManage` is the command-line interface to Oracle VM VirtualBox. With it, you can completely control Oracle VM VirtualBox from the command line of your host operating system. `VBoxManage` supports all the features that the graphical user interface gives you access to, but it supports a lot more than that. It exposes all the features of the virtualization engine, even those that cannot be accessed from the GUI.

You will need to use the command line if you want to do the following:

- Use a different user interface than the main GUI such as the VBoxHeadless server.
- Control some of the more advanced and experimental configuration settings for a VM.

There are two main things to keep in mind when using `VBoxManage`. First, `VBoxManage` must always be used with a specific subcommand, such as `list` or `createvm` or `startvm`. All the subcommands that `VBoxManage` supports are described in detail in chapter 8, *VBoxManage*, page 120.

Second, most of these subcommands require that you specify a particular virtual machine after the subcommand. There are two ways you can do this:

- You can specify the VM name, as it is shown in the Oracle VM VirtualBox GUI. Note that if that name contains spaces, then you must enclose the entire name in double quotes. This is always required with command line arguments that contain spaces. For example:

```
VBoxManage startvm "Windows XP"
```

- You can specify the UUID, which is the internal unique identifier that Oracle VM VirtualBox uses to refer to the virtual machine. Assuming that the VM called “Windows XP” has the UUID shown below, the following command has the same effect as the previous example:

```
VBoxManage startvm 670e746d-abea-4ba6-ad02-2a3b043810a5
```

You can enter `VBoxManage list vms` to have all currently registered VMs listed with all their settings, including their respective names and UUIDs.

Some typical examples of how to control Oracle VM VirtualBox from the command line are listed below:

- To create a new virtual machine from the command line and immediately register it with Oracle VM VirtualBox, use `VBoxManage createvm` with the `--register` option, as follows:

```
$ VBoxManage createvm --name "SUSE 10.2" --register
VirtualBox Command Line Management Interface Version <version-number>
(C) 2005-2018 Oracle Corporation
All rights reserved.

Virtual machine 'SUSE 10.2' is created.
UUID: c89fc351-8ec6-4f02-a048-57f4d25288e5
Settings file: '/home/username/.config/VirtualBox/Machines/SUSE 10.2/SUSE 10.2.xml'
```

As can be seen from the above output, a new virtual machine has been created with a new UUID and a new XML settings file.

For more details, see chapter 8.7, *VBoxManage createvm*, page 134.

- To show the configuration of a particular VM, use `VBoxManage showvminfo`. See chapter 8.5, *VBoxManage showvminfo*, page 133 for details and an example.
- To change settings while a VM is powered off, use `VBoxManage modifyvm`. For example:

```
VBoxManage modifyvm "Windows XP" --memory 512
```

  See also chapter 8.8, *VBoxManage modifyvm*, page 135.

- To change the storage configuration, such as to add a storage controller and then a virtual disk, use `VBoxManage storagectl` and `VBoxManage storageattach`. See chapter 8.20, *VBoxManage storagectl*, page 164 and chapter 8.19, *VBoxManage storageattach*, page 160.
- To control VM operation, use one of the following:
  - To start a VM that is currently powered off, use `VBoxManage startvm`. See chapter 8.13, *VBoxManage startvm*, page 151.
  - To pause or save a VM that is currently running or change some of its settings, use `VBoxManage controlvm`. See chapter 8.14, *VBoxManage controlvm*, page 152.

## 8.2 Commands Overview

When running `VBoxManage` without parameters or when supplying an invalid command line, the following command syntax list is shown. Note that the output will be slightly different depending on the host platform. If in doubt, check the output of `VBoxManage` for the commands available on your particular host.

```
Usage:

  VBoxManage [<general option>] <command>


General Options:

  [-v|--version]            print version number and exit
  [-q|--nologo]             suppress the logo
  [--settingspw <pw>]       provide the settings password
  [--settingspwfile <file>] provide a file containing the settings password
  [@<response-file>]        load arguments from the given response file (bourne style)


Commands:

  list [--long|-l] [--sorted|-s]          vms|runningvms|ostypes|hostdvds|hostfloppies|
                            intnets|bridgedifs|hostonlyifs|natnets|dhcpservers|
                            hostinfo|hostcpuids|hddbackends|hdds|dvds|floppies|
                            usbhost|usbfilters|systemproperties|extpacks|
                            groups|webcams|screenshotformats|cloudproviders|
                            cloudprofiles

  showvminfo                <uuid|vmname> [--details]
                            [--machinereadable]
  showvminfo                <uuid|vmname> --log <idx>

  registervm                <filename>

  unregistervm              <uuid|vmname> [--delete]
```

```
createvm                 --name <name>
                         [--groups <group>, ...]
                         [--ostype <ostype>]
                         [--register]
                         [--basefolder <path>]
                         [--uuid <uuid>]
                         [--default]

modifyvm                 <uuid|vmname>
                         [--name <name>]
                         [--groups <group>, ...]
                         [--description <desc>]
                         [--ostype <ostype>]
                         [--iconfile <filename>]
                         [--memory <memorysize in MB>]
                         [--pagefusion on|off]
                         [--vram <vramsize in MB>]
                         [--acpi on|off]
                         [--pciattach 03:04.0]
                         [--pciattach 03:04.0@02:01.0]
                         [--pcidetach 03:04.0]
                         [--ioapic on|off]
                         [--hpet on|off]
                         [--triplefaultreset on|off]
                         [--apic on|off]
                         [--x2apic on|off]
                         [--paravirtprovider none|default|legacy|minimal|
                                             hyperv|kvm]
                         [--paravirtdebug <key=value> [,<key=value> ...]]
                         [--hwvirtex on|off]
                         [--nestedpaging on|off]
                         [--largepages on|off]
                         [--vtxvpid on|off]
                         [--vtxux on|off]
                         [--pae on|off]
                         [--longmode on|off]
                         [--ibpb-on-vm-exit on|off]
                         [--ibpb-on-vm-entry on|off]
                         [--spec-ctrl on|off]
                         [--l1d-flush-on-sched on|off]
                         [--l1d-flush-on-vm-entry on|off]
                         [--mds-clear-on-sched on|off]
                         [--mds-clear-on-vm-entry on|off]
                         [--nested-hw-virt on|off]
                         [--cpu-profile "host|Intel 80[86|286|386]"]
                         [--cpuid-portability-level <0..3>
                         [--cpuid-set <leaf[:subleaf]> <eax> <ebx> <ecx> <edx>]
                         [--cpuid-remove <leaf[:subleaf]>]
                         [--cpuidremoveall]
                         [--hardwareuuid <uuid>]
                         [--cpus <number>]
                         [--cpuhotplug on|off]
                         [--plugcpu <id>]
                         [--unplugcpu <id>]
                         [--cpuexecutioncap <1-100>]
                         [--rtcuseutc on|off]
                         [--graphicscontroller none|vboxvga|vmsvga|vboxsvga]
                         [--monitorcount <number>]
                         [--accelerate3d on|off]
                         [--accelerate2dvideo on|off]
                         [--firmware bios|efi|efi32|efi64]
                         [--chipset ich9|piix3]
                         [--bioslogofadein on|off]
                         [--bioslogofadeout on|off]
                         [--bioslogodisplaytime <msec>]
                         [--bioslogoimagepath <imagepath>]
```

```
[--biosbootmenu disabled|menuonly|messageandmenu]
[--biosapic disabled|apic|x2apic]
[--biossystemtimeoffset <msec>]
[--biospxedebug on|off]
[--boot<1-4> none|floppy|dvd|disk|net>]
[--nic<1-N> none|null|nat|bridged|intnet|hostonly|
           generic|natnetwork]
[--nictype<1-N> Am79C970A|Am79C973|
                82540EM|82543GC|82545EM|
                virtio]
[--cableconnected<1-N> on|off]
[--nictrace<1-N> on|off]
[--nictracefile<1-N> <filename>]
[--nicproperty<1-N> name=[value]]
[--nicspeed<1-N> <kbps>]
[--nicbootprio<1-N> <priority>]
[--nicpromisc<1-N> deny|allow-vms|allow-all]
[--nicbandwidthgroup<1-N> none|<name>]
[--bridgeadapter<1-N> none|<devicename>]
[--hostonlyadapter<1-N> none|<devicename>]
[--intnet<1-N> <network name>]
[--nat-network<1-N> <network name>]
[--nicgenericdrv<1-N> <driver>
[--natnet<1-N> <network>|default]
[--natsettings<1-N> [<mtu>],[<socksnd>],
                    [<sockrcv>],[<tcpsnd>],
                    [<tcprcv>]]
[--natpf<1-N> [<rulename>],tcp|udp,[<hostip>],
             <hostport>,[<guestip>],<guestport>]
[--natpf<1-N> delete <rulename>]
[--nattftpprefix<1-N> <prefix>]
[--nattftpfile<1-N> <file>]
[--nattftpserver<1-N> <ip>]
[--natbindip<1-N> <ip>
[--natdnspassdomain<1-N> on|off]
[--natdnsproxy<1-N> on|off]
[--natdnshostresolver<1-N> on|off]
[--nataliasmode<1-N> default|[log],[proxyonly],
                              [sameports]]
[--macaddress<1-N> auto|<mac>]
[--mouse ps2|usb|usbtablet|usbmultitouch]
[--keyboard ps2|usb
[--uart<1-N> off|<I/O base> <IRQ>]
[--uartmode<1-N> disconnected|
                 server <pipe>|
                 client <pipe>|
                 tcpserver <port>|
                 tcpclient <hostname:port>|
                 file <file>|
                 <devicename>]
[--uarttype<1-N> 16450|16550A|16750
[--lpt<1-N> off|<I/O base> <IRQ>]
[--lptmode<1-N> <devicename>]
[--guestmemoryballoon <balloonsize in MB>]
[--audio none|null|dsound|oss|alsa|pulse|
         oss|pulse|coreaudio]
[--audioin on|off]
[--audioout on|off]
[--audiocontroller ac97|hda|sb16]
[--audiocodec stac9700|ad1980|stac9221|sb16]
[--clipboard disabled|hosttoguest|guesttohost|
             bidirectional]
[--draganddrop disabled|hosttoguest|guesttohost|
             bidirectional]
[--vrde on|off]
[--vrdeextpack default|<name>
[--vrdeproperty <name=[value]>]
```

```
                        [--vrdeport <hostport>]
                        [--vrdeaddress <hostip>]
                        [--vrdeauthtype null|external|guest]
                        [--vrdeauthlibrary default|<name>
                        [--vrdemulticon on|off]
                        [--vrdereusecon on|off]
                        [--vrdevideochannel on|off]
                        [--vrdevideochannelquality <percent>]
                        [--usbohci on|off]
                        [--usbehci on|off]
                        [--usbxhci on|off]
                        [--usbrename <oldname> <newname>]
                        [--snapshotfolder default|<path>]
                        [--teleporter on|off]
                        [--teleporterport <port>]
                        [--teleporteraddress <address|empty>
                        [--teleporterpassword <password>]
                        [--teleporterpasswordfile <file>|stdin]
                        [--tracing-enabled on|off]
                        [--tracing-config <config-string>]
                        [--tracing-allow-vm-access on|off]
                        [--usbcardreader on|off]
                        [--autostart-enabled on|off]
                        [--autostart-delay <seconds>]
                        [--recording on|off]
                        [--recordingscreens all|<screen ID> [<screen ID> ...]]
                        [--recordingfile <filename>]
                        [--recordingvideores <width> <height>]
                        [--recordingvideorate <rate>]
                        [--recordingvideofps <fps>]
                        [--recordingmaxtime <s>]
                        [--recordingmaxsize <MB>]
                        [--recordingopts <key=value> [,<key=value> ...]]
                        [--defaultfrontend default|<name>]

clonevm                 <uuid|vmname>
                        [--snapshot <uuid>|<name>]
                        [--mode machine|machineandchildren|all]
                        [--options link|keepallmacs|keepnatmacs|
                                   keepdisknames|keephwuuids]
                        [--name <name>]
                        [--groups <group>, ...]
                        [--basefolder <basefolder>]
                        [--uuid <uuid>]
                        [--register]

movevm                  <uuid|vmname>
                        --type basic
                        [--folder <path>]

import                  <ovfname/ovaname>
                        [--dry-run|-n]
                        [--options keepallmacs|keepnatmacs|importtovdi]
                        [more options]
                        (run with -n to have options displayed
                         for a particular OVF)

export                  <machines> --output|-o <name>.<ovf/ova/tar.gz>
                        [--legacy09|--ovf09|--ovf10|--ovf20|--opc10]
                        [--manifest]
                        [--iso]
                        [--options manifest|iso|nomacs|nomacsbutnat]
                        [--vsys <number of virtual system>]
                                [--vmname <name>]
                                [--product <product name>]
                                [--producturl <product url>]
                                [--vendor <vendor name>]
```

```
                        [--vendorurl <vendor url>]
                        [--version <version info>]
                        [--description <description info>]
                        [--eula <license text>]
                        [--eulafile <filename>]
                    [--cloud <number of virtual system>]
                        [--vmname <name>]
                        [--cloudprofile <cloud profile name>]
                        [--cloudshape <shape>]
                        [--clouddomain <domain>]
                        [--clouddisksize <disk size in GB>]
                        [--cloudbucket <bucket name>]
                        [--cloudocivcn <OCI vcn id>]
                        [--cloudocisubnet <OCI subnet id>]
                        [--cloudkeepobject <true/false>]
                        [--cloudlaunchinstance <true/false>]
                        [--cloudpublicip <true/false>]

startvm                 <uuid|vmname>...
                        [--type gui|sdl|headless|separate]
                        [-E|--putenv <NAME>[=<VALUE>]]

controlvm               <uuid|vmname>
                        pause|resume|reset|poweroff|savestate|
                        acpipowerbutton|acpisleepbutton|
                        keyboardputscancode <hex> [<hex> ...]|
                        keyboardputstring <string1> [<string2> ...]|
                        keyboardputfile <filename>|
                        setlinkstate<1-N> on|off |
                        nic<1-N> null|nat|bridged|intnet|hostonly|generic|
                                 natnetwork [<devicename>] |
                        nictrace<1-N> on|off |
                        nictracefile<1-N> <filename> |
                        nicproperty<1-N> name=[value] |
                        nicpromisc<1-N> deny|allow-vms|allow-all |
                        natpf<1-N> [<rulename>],tcp|udp,[<hostip>],
                                    <hostport>,[<guestip>],<guestport> |
                        natpf<1-N> delete <rulename> |
                        guestmemoryballoon <balloonsize in MB> |
                        usbattach <uuid>|<address>
                                  [--capturefile <filename>] |
                        usbdetach <uuid>|<address> |
                        audioin on|off |
                        audioout on|off |
                        clipboard disabled|hosttoguest|guesttohost|
                                  bidirectional |
                        draganddrop disabled|hosttoguest|guesttohost|
                                  bidirectional |
                        vrde on|off |
                        vrdeport <port> |
                        vrdeproperty <name=[value]> |
                        vrdevideochannelquality <percent> |
                        setvideomodehint <xres> <yres> <bpp>
                                         [[<display>] [<enabled:yes|no> |
                                           [<xorigin> <yorigin>]]] |
                        setscreenlayout <display> on|primary <xorigin> <yorigin> <xres> <yres> <bpp> | off
                        screenshotpng <file> [display] |
                        recording on|off |
                        recording screens all|none|<screen>,[<screen>...] |
                        recording filename <file> |
                        recording videores <width>x<height> |
                        recording videorate <rate> |
                        recording videofps <fps> |
                        recording maxtime <s> |
                        recording maxfilesize <MB> |
                        setcredentials <username>
                                       --passwordfile <file> | <password>
```

```
                              <domain>
                              [--allowlocallogon <yes|no>] |
                         teleport --host <name> --port <port>
                                  [--maxdowntime <msec>]
                                  [--passwordfile <file> |
                                   --password <password>] |
                         plugcpu <id> |
                         unplugcpu <id> |
                         cpuexecutioncap <1-100>
                         webcam <attach [path [settings]]> | <detach [path]> | <list>
                         addencpassword <id>
                                  <password file>|-
                                  [--removeonsuspend <yes|no>]
                         removeencpassword <id>
                         removeallencpasswords
                         changeuartmode<1-N> disconnected|
                                             server <pipe>|
                                             client <pipe>|
                                             tcpserver <port>|
                                             tcpclient <hostname:port>|
                                             file <file>|
                                             <devicename>]

discardstate             <uuid|vmname>

adoptstate               <uuid|vmname> <state_file>

snapshot                 <uuid|vmname>
                         take <name> [--description <desc>] [--live]
                              [--uniquename Number,Timestamp,Space,Force] |
                         delete <uuid|snapname> |
                         restore <uuid|snapname> |
                         restorecurrent |
                         edit <uuid|snapname>|--current
                              [--name <name>]
                              [--description <desc>] |
                         list [--details|--machinereadable] |
                         showvminfo <uuid|snapname>

closemedium              [disk|dvd|floppy] <uuid|filename>
                         [--delete]

storageattach            <uuid|vmname>
                         --storagectl <name>
                         [--port <number>]
                         [--device <number>]
                         [--type dvddrive|hdd|fdd]
                         [--medium none|emptydrive|additions|
                                   <uuid|filename>|host:<drive>|iscsi]
                         [--mtype normal|writethrough|immutable|shareable|
                                  readonly|multiattach]
                         [--comment <text>]
                         [--setuuid <uuid>]
                         [--setparentuuid <uuid>]
                         [--passthrough on|off]
                         [--tempeject on|off]
                         [--nonrotational on|off]
                         [--discard on|off]
                         [--hotpluggable on|off]
                         [--bandwidthgroup <name>]
                         [--forceunmount]
                         [--server <name>|<ip>]
                         [--target <target>]
                         [--tport <port>]
                         [--lun <lun>]
                         [--encodedlun <lun>]
                         [--username <username>]
```

```
                        [--password <password>]
                        [--passwordfile <file>]
                        [--initiator <initiator>]
                        [--intnet]

storagectl              <uuid|vmname>
                        --name <name>
                        [--add ide|sata|scsi|floppy|sas|usb|pcie]
                        [--controller LSILogic|LSILogicSAS|BusLogic|
                                      IntelAHCI|PIIX3|PIIX4|ICH6|I82078|
                        [             USB|NVMe]
                        [--portcount <1-n>]
                        [--hostiocache on|off]
                        [--bootable on|off]
                        [--rename <name>]
                        [--remove]

bandwidthctl            <uuid|vmname>
                        add <name> --type disk|network
                            --limit <megabytes per second>[k|m|g|K|M|G] |
                        set <name>
                            --limit <megabytes per second>[k|m|g|K|M|G] |
                        remove <name> |
                        list [--machinereadable]
                        (limit units: k=kilobit, m=megabit, g=gigabit,
                                      K=kilobyte, M=megabyte, G=gigabyte)

showmediuminfo          [disk|dvd|floppy] <uuid|filename>

createmedium            [disk|dvd|floppy] --filename <filename>
                        [--size <megabytes>|--sizebyte <bytes>]
                        [--diffparent <uuid>|<filename>
                        [--format VDI|VMDK|VHD] (default: VDI)
                        [--variant Standard,Fixed,Split2G,Stream,ESX,
                                   Formatted]

modifymedium            [disk|dvd|floppy] <uuid|filename>
                        [--type normal|writethrough|immutable|shareable|
                                readonly|multiattach]
                        [--autoreset on|off]
                        [--property <name=[value]>]
                        [--compact]
                        [--resize <megabytes>|--resizebyte <bytes>]
                        [--move <path>]
                        [--setlocation <path>]
                        [--description <description string>]
clonemedium             [disk|dvd|floppy] <uuid|inputfile> <uuid|outputfile>
                        [--format VDI|VMDK|VHD|RAW|<other>]
                        [--variant Standard,Fixed,Split2G,Stream,ESX]
                        [--existing]

mediumproperty          [disk|dvd|floppy] set <uuid|filename>
                        <property> <value>

                        [disk|dvd|floppy] get <uuid|filename>
                        <property>

                        [disk|dvd|floppy] delete <uuid|filename>
                        <property>

encryptmedium           <uuid|filename>
                        [--newpassword <file>|-]
                        [--oldpassword <file>|-]
                        [--cipher <cipher identifier>]
                        [--newpasswordid <password identifier>]

checkmediumpwd          <uuid|filename>
```

```
                        <pwd file>|-

convertfromraw          <filename> <outputfile>
                        [--format VDI|VMDK|VHD]
                        [--variant Standard,Fixed,Split2G,Stream,ESX]
                        [--uuid <uuid>]
convertfromraw          stdin <outputfile> <bytes>
                        [--format VDI|VMDK|VHD]
                        [--variant Standard,Fixed,Split2G,Stream,ESX]
                        [--uuid <uuid>]

getextradata            global|<uuid|vmname>
                        <key>|[enumerate]

setextradata            global|<uuid|vmname>
                        <key>
                        [<value>] (no value deletes key)

setproperty             machinefolder default|<folder> |
                        hwvirtexclusive on|off |
                        vrdeauthlibrary default|<library> |
                        websrvauthlibrary default|null|<library> |
                        vrdeextpack null|<library> |
                        autostartdbpath null|<folder> |
                        loghistorycount <value>
                        defaultfrontend default|<name>
                        logginglevel <log setting>
                        proxymode system|noproxy|manual
                        proxyurl <url>

usbfilter               add <index,0-N>
                        --target <uuid|vmname>|global
                        --name <string>
                        --action ignore|hold (global filters only)
                        [--active yes|no] (yes)
                        [--vendorid <XXXX>] (null)
                        [--productid <XXXX>] (null)
                        [--revision <IIFF>] (null)
                        [--manufacturer <string>] (null)
                        [--product <string>] (null)
                        [--remote yes|no] (null, VM filters only)
                        [--serialnumber <string>] (null)
                        [--maskedinterfaces <XXXXXXXX>]

usbfilter               modify <index,0-N>
                        --target <uuid|vmname>|global
                        [--name <string>]
                        [--action ignore|hold] (global filters only)
                        [--active yes|no]
                        [--vendorid <XXXX>|""]
                        [--productid <XXXX>|""]
                        [--revision <IIFF>|""]
                        [--manufacturer <string>|""]
                        [--product <string>|""]
                        [--remote yes|no] (null, VM filters only)
                        [--serialnumber <string>|""]
                        [--maskedinterfaces <XXXXXXXX>]

usbfilter               remove <index,0-N>
                        --target <uuid|vmname>|global

sharedfolder            add <uuid|vmname>
                        --name <name> --hostpath <hostpath>
                        [--transient] [--readonly] [--automount]

sharedfolder            remove <uuid|vmname>
                        --name <name> [--transient]
```

```
guestproperty             get <uuid|vmname>
                          <property> [--verbose]

guestproperty             set <uuid|vmname>
                          <property> [<value> [--flags <flags>]]

guestproperty             delete|unset <uuid|vmname>
                          <property>

guestproperty             enumerate <uuid|vmname>
                          [--patterns <patterns>]

guestproperty             wait <uuid|vmname> <patterns>
                          [--timeout <msec>] [--fail-on-timeout]

guestcontrol              <uuid|vmname> [--verbose|-v] [--quiet|-q]
                            [--username <name>] [--domain <domain>]
                            [--passwordfile <file> | --password <password>]

                            run [common-options]
                            [--exe <path to executable>] [--timeout <msec>]
                            [-E|--putenv <NAME>[=<VALUE>]] [--unquoted-args]
                            [--ignore-operhaned-processes] [--profile]
                            [--no-wait-stdout|--wait-stdout]
                            [--no-wait-stderr|--wait-stderr]
                            [--dos2unix] [--unix2dos]
                            -- <program/arg0> [argument1] ... [argumentN]]

                            start [common-options]
                            [--exe <path to executable>] [--timeout <msec>]
                            [-E|--putenv <NAME>[=<VALUE>]] [--unquoted-args]
                            [--ignore-operhaned-processes] [--profile]
                            -- <program/arg0> [argument1] ... [argumentN]]

                            copyfrom [common-options]
                            [--follow] [-R|--recursive]
                            <guest-src0> [guest-src1 [...]] <host-dst>

                            copyfrom [common-options]
                            [--follow] [-R|--recursive]
                            [--target-directory <host-dst-dir>]
                            <guest-src0> [guest-src1 [...]]

                            copyto [common-options]
                            [--follow] [-R|--recursive]
                            <host-src0> [host-src1 [...]] <guest-dst>

                            copyto [common-options]
                            [--follow] [-R|--recursive]
                            [--target-directory <guest-dst>]
                            <host-src0> [host-src1 [...]]

                            mkdir|createdir[ectory] [common-options]
                            [--parents] [--mode <mode>]
                            <guest directory> [...]

                            rmdir|removedir[ectory] [common-options]
                            [-R|--recursive]
                            <guest directory> [...]

                            removefile|rm [common-options] [-f|--force]
                            <guest file> [...]

                            mv|move|ren[ame] [common-options]
                            <source> [source1 [...]] <dest>
```

```
                          mktemp|createtemp[orary] [common-options]
                          [--secure] [--mode <mode>] [--tmpdir <directory>]
                          <template>

                          stat [common-options]
                          <file> [...]

guestcontrol              <uuid|vmname> [--verbose|-v] [--quiet|-q]

                          list <all|sessions|processes|files> [common-opts]

                          closeprocess [common-options]
                          <   --session-id <ID>
                            | --session-name <name or pattern>
                          <PID1> [PID1 [...]]

                          closesession [common-options]
                          <  --all | --session-id <ID>
                            | --session-name <name or pattern> >

                          updatega|updateguestadditions|updateadditions
                          [--source <guest additions .ISO>]
                          [--wait-start] [common-options]
                          [-- [<argument1>] ... [<argumentN>]]

                          watch [common-options]

metrics                   list [*|host|<vmname> [<metric_list>]]
                                              (comma-separated)

metrics                   setup
                          [--period <seconds>] (default: 1)
                          [--samples <count>] (default: 1)
                          [--list]
                          [*|host|<vmname> [<metric_list>]]

metrics                   query [*|host|<vmname> [<metric_list>]]

metrics                   enable
                          [--list]
                          [*|host|<vmname> [<metric_list>]]

metrics                   disable
                          [--list]
                          [*|host|<vmname> [<metric_list>]]

metrics                   collect
                          [--period <seconds>] (default: 1)
                          [--samples <count>] (default: 1)
                          [--list]
                          [--detach]
                          [*|host|<vmname> [<metric_list>]]

natnetwork                add --netname <name>
                          --network <network>
                          [--enable|--disable]
                          [--dhcp on|off]
                          [--port-forward-4 <rule>]
                          [--loopback-4 <rule>]
                          [--ipv6 on|off]
                          [--port-forward-6 <rule>]
                          [--loopback-6 <rule>]

natnetwork                remove --netname <name>

natnetwork                modify --netname <name>
                          [--network <network>]
```

```
                            [--enable|--disable]
                            [--dhcp on|off]
                            [--port-forward-4 <rule>]
                            [--loopback-4 <rule>]
                            [--ipv6 on|off]
                            [--port-forward-6 <rule>]
                            [--loopback-6 <rule>]

  natnetwork                start --netname <name>

  natnetwork                stop --netname <name>

  natnetwork                list [<pattern>]

  hostonlyif                ipconfig <name>
                            [--dhcp |
                            --ip<ipv4> [--netmask<ipv4> (def: 255.255.255.0)] |
                            --ipv6<ipv6> [--netmasklengthv6<length> (def: 64)]]
                            create |
                            remove <name>

  dhcpserver                add|modify --netname <network_name> |
                                       --ifname <hostonly_if_name>
                            [--ip <ip_address>
                            --netmask <network_mask>
                            --lowerip <lower_ip>
                            --upperip <upper_ip>]
                            [--enable | --disable]
                            [--options [--vm <name> --nic <1-N>]
                             --id <number> [--value <string> | --remove]]
                             (multiple options allowed after --options)

  dhcpserver                remove --netname <network_name> |
                                   --ifname <hostonly_if_name>

  usbdevsource              add <source name>
                            --backend <backend>
                            --address <address>
  usbdevsource              remove <source name>

  VBoxManage mediumio <[--disk=uuid|filename] | [--dvd=uuid|filename] | [--floppy=uuid|filename]>
     [--password-file-|filename] formatfat [--quick]
  VBoxManage mediumio <[--disk=uuid|filename] | [--dvd=uuid|filename] | [--floppy=uuid|filename]>
     [--password-file-|filename] cat [--hex] [--offset=byte-offset] [--size=bytes] [--output=-|filename]
  VBoxManage mediumio <[--disk=uuid|filename] | [--dvd=uuid|filename] | [--floppy=uuid|filename]>
[--password-file-|filename] stream [--format=image-format] [--variant=image-variant] [--output=-|filename]

  VBoxManage debugvm <uuid|vmname> dumpvmcore [--filename=name]
  VBoxManage debugvm <uuid|vmname> info <item> [args...]
  VBoxManage debugvm <uuid|vmname> injectnmi
  VBoxManage debugvm <uuid|vmname> log [[--release] | [--debug]] [group-settings...]
  VBoxManage debugvm <uuid|vmname> logdest [[--release] | [--debug]] [destinations...]
  VBoxManage debugvm <uuid|vmname> logflags [[--release] | [--debug]] [flags...]
  VBoxManage debugvm <uuid|vmname> osdetect
  VBoxManage debugvm <uuid|vmname> osinfo
  VBoxManage debugvm <uuid|vmname> osdmesg [--lines=lines]
  VBoxManage debugvm <uuid|vmname> getregisters [--cpu=id] [reg-set.reg-name...]
  VBoxManage debugvm <uuid|vmname> setregisters [--cpu=id] [reg-set.reg-name=value...]
  VBoxManage debugvm <uuid|vmname> show [[--human-readable] | [--sh-export] | [--sh-eval] | [--cmd-set]]
     [settings-item...]
  VBoxManage debugvm <uuid|vmname> stack [--cpu=id]
  VBoxManage debugvm <uuid|vmname> statistics [--reset] [--descriptions] [--pattern=pattern]

  VBoxManage extpack install [--replace] <tarball>
  VBoxManage extpack uninstall [--force] <name>
  VBoxManage extpack cleanup
```

```
  VBoxManage unattended detect <--iso=install-iso> [--machine-readable]
  VBoxManage unattended install <uuid|vmname> <--iso=install-iso> [--user=login] [--password=password]
[--password-file=file] [--full-user-name=name] [--key=product-key] [--install-additions] [--no-install-additions]
[--additions-iso=add-iso] [--install-txs] [--no-install-txs] [--validation-kit-iso=testing-iso]
[--locale=ll_CC] [--country=CC] [--time-zone=tz] [--hostname=fqdn] [--package-selection-adjustment=keyword]
[--dry-run] [--auxiliary-base-path=path] [--image-index=number] [--script-template=file] [--post-install-template=fil
[--post-install-command=command] [--extra-install-kernel-parameters=params] [--language=lang]
[--start-vm=session-type]
```

Each time `VBoxManage` is invoked, only one command can be executed. However, a command might support several subcommands which then can be invoked in one single call. The following sections provide detailed reference information on the different commands.

## 8.3 General Options

- `-v|--version`: Show the version of this tool and exit.
- `--nologo`: Suppress the output of the logo information. This option is useful for scripts.
- `--settingspw`: Specifiy a settings password.
- `--settingspwfile`: Specify a file containing the settings password.

The settings password is used for certain settings which need to be stored in encrypted form for security reasons. At the moment, the only encrypted setting is the iSCSI initiator secret, see chapter 8.19, *VBoxManage storageattach*, page 160. As long as no settings password is specified, this information is stored in *plain text*. After using the `--settingspw|--settingspwfile` option once, it must be always used. Otherwise, the encrypted setting cannot be unencrypted.

## 8.4 VBoxManage list

The `list` command gives relevant information about your system and information about Oracle VM VirtualBox's current settings.

The following subcommands are available with `VBoxManage list`:

- `vms`: Lists all virtual machines currently registered with Oracle VM VirtualBox. By default this displays a compact list with each VM's name and UUID. If you also specify `--long` or `-l`, this will be a detailed list as with the `showvminfo` command, see chapter 8.5, *VBoxManage showvminfo*, page 133.
- `runningvms`: Lists all currently running virtual machines by their unique identifiers (UUIDs) in the same format as with `vms`.
- `ostypes`: Lists all guest operating systems presently known to Oracle VM VirtualBox, along with the identifiers used to refer to them with the `modifyvm` command.
- `hostdvds`, `hostfloppies`: Lists the DVD, floppy, bridged networking, and host-only networking interfaces on the host, along with the name used to access them from within Oracle VM VirtualBox.
- `intnets`: Displays information about the internal networks.
- `bridgedifs`, `hostonlyifs`, `natnets`, `dhcpservers`: Lists the bridged network interfaces, host-only network interfaces, NAT network interfaces, and DHCP servers currently available on the host. See chapter 6, *Virtual Networking*, page 98.

- `hostinfo`: Displays information about the host system, such as CPUs, memory size, and operating system version.
- `hostcpuids`: Lists the CPUID parameters for the host CPUs. This can be used for a more fine grained analyis of the host's virtualization capabilities.
- `hddbackends`: Lists all known virtual disk back-ends of Oracle VM VirtualBox. For each such format, such as VDI, VMDK, or RAW, this subcommand lists the back-end's capabilities and configuration.
- `hdds`, `dvds`, `floppies`: Shows information about virtual disk images currently in use by Oracle VM VirtualBox, including all their settings, the unique identifiers (UUIDs) associated with them by Oracle VM VirtualBox and all files associated with them. This is the command-line equivalent of the Virtual Media Manager. See chapter 5.3, *The Virtual Media Manager*, page 87.
- `usbhost`: Shows information about USB devices attached to the host, including information useful for constructing USB filters and whether they are currently in use by the host.
- `usbfilters`: Lists all global USB filters registered with Oracle VM VirtualBox and displays the filter parameters. Global USB filters are for devices which are accessible to all virtual machines.
- `systemproperties`: Displays some global Oracle VM VirtualBox settings, such as minimum and maximum guest RAM and virtual hard disk size, folder settings and the current authentication library in use.
- `extpacks`: Displays all Oracle VM VirtualBox extension packs that are currently installed. See chapter 1.6, *Installing Oracle VM VirtualBox and Extension Packs*, page 6 and chapter 8.43, *VBoxManage extpack*, page 201.
- `groups`: Displays details of the VM Groups. See chapter 1.10, *Using VM Groups*, page 16.
- `webcams`: Displays a list of webcams attached to the running VM. The output format is a list of absolute paths or aliases that were used for attaching the webcams to the VM using the webcam attach command.
- `screenshotformats`: Displays a list of available screenshot formats.
- `cloudproviders`: Displays a list of cloud providers that are supported by Oracle VM VirtualBox. Oracle Cloud Infrastructure is an example of a cloud provider.
- `cloudprofiles`: Displays a list of cloud profiles that have been configured.

  Cloud profiles are used when exporting VMs to a cloud service. See chapter 1.15.4, *Exporting an Appliance to Oracle Cloud Infrastructure*, page 24.

## 8.5 VBoxManage showvminfo

The `showvminfo` command shows information about a particular virtual machine. This is the same information as `VBoxManage list vms --long` would show for all virtual machines.

You will see information as shown in the following example.

```
$ VBoxManage showvminfo "Windows XP"
VirtualBox Command Line Management Interface Version <version-number>
(C) 2005-2018 Oracle Corporation
All rights reserved.

Name:           Windows XP
```

```
Guest OS:        Other/Unknown
UUID:            1bf3464d-57c6-4d49-92a9-a5cc3816b7e7
Config file:     /home/username/.config/VirtualBox/Machines/Windows XP/Windows XP.xml
Memory size:     512MB
VRAM size:       12MB
Number of CPUs:  2
Boot menu mode:  message and menu
Boot Device (1): DVD
Boot Device (2): HardDisk
Boot Device (3): Not Assigned
Boot Device (4): Not Assigned
ACPI:            on
IOAPIC:          on
...
```

Use the `--machinereadable` option to produce the same output, but in machine readable format with a property=value string on each line. For example:

```
...
groups="/"
ostype="Oracle (64-bit)"
UUID="457af700-bc0a-4258-aa3c-13b03da171f2"
...
```

## 8.6 VBoxManage registervm/unregistervm

The `registervm` command enables you to import a virtual machine definition in an XML file into Oracle VM VirtualBox. The machine must not conflict with one already registered in Oracle VM VirtualBox and it may not have any hard or removable disks attached. It is advisable to place the definition file in the machines folder before registering it.

> **Note:** When creating a new virtual machine with `VBoxManage createvm`, as shown in chapter 8.7, *VBoxManage createvm*, page 134, you can directly specify the `--register` option to avoid having to register it separately.

The `unregistervm` command unregisters a virtual machine. If `--delete` is also specified, the following files will also be deleted automatically:

- All hard disk image files, including differencing files, which are used by the machine and not shared with other machines.
- Saved state files that the machine created. One if the machine was in Saved state and one for each online snapshot.
- The machine XML file and its backups.
- The machine log files.
- The machine directory, if it is empty after having deleted all of the above files.

## 8.7 VBoxManage createvm

The `VBoxManage createvm` command creates a new XML virtual machine definition file.

You must specify the name of the VM by using `--name <name>`. This name is used by default as the file name of the settings file that has the `.xml` extension and the machine folder, which

is a subfolder of the `.config/VirtualBox/Machines` folder. Note that the machine folder path name varies based on the OS type and the Oracle VM VirtualBox version.

Ensure that the VM name conforms to the host OS's file name requirements. If you later rename the VM, the file and folder names will be updated to match the new name automatically.

The `--basefolder <path>` option specifies the machine folder path name. Note that the names of the file and the folder do not change if you rename the VM.

The `--group <group-ID>, ...` option assigns the VM to the specified groups. Note that group IDs always start with `/` so that they can be nested. By default, each VM is assigned membership to the `/` group.

The `--ostype <ostype>` option specifies the guest OS to run in the VM. Run the `VBoxManage list ostypes` command to see the available OS types.

The `--uuid <uuid>` option specifies the universal unique identifier (UUID) of the VM. The UUID must be unique within the namespace of the host or of its VM group memberships. By default, the `VBoxManage` command automatically generates the UUID.

The `--default` option applies a default hardware configuration for the specified guest OS. By default, the VM is created with minimal hardware.

The `--register` option registers the VM with your Oracle VM VirtualBox installation. By default, the `VBoxManage createvm` command creates only the XML configuration for the VM but does not registered the VM. If you do not register the VM at creation, you can run the `VBoxManage registervm` command after you create the VM.

## 8.8 VBoxManage modifyvm

This command changes the properties of a registered virtual machine which is not running. Most of the properties that this command makes available correspond to the VM settings that Oracle VM VirtualBox graphical user interface displays in each VM's **Settings** dialog. These are described in chapter 3, *Configuring Virtual Machines*, page 40. However, some of the more advanced settings are only available through the `VBoxManage` interface.

These commands require that the machine is powered off, neither running nor in a Saved state. Some machine settings can also be changed while a machine is running. Those settings will then have a corresponding subcommand with the `VBoxManage controlvm` subcommand. See chapter 8.14, *VBoxManage controlvm*, page 152.

### 8.8.1 General Settings

The following general settings are available through `VBoxManage modifyvm`:

- `--name <name>`: Changes the VM's name and can be used to rename the internal virtual machine files, as described in chapter 8.7, *VBoxManage createvm*, page 134.
- `--groups <group>, ...`: Changes the group membership of a VM. Groups always start with a `/` and can be nested. By default VMs are in group `/`.
- `--description <desc>`: Changes the VM's description, which is a way to record details about the VM in a way which is meaningful for the user. The GUI interprets HTML formatting, the command line allows arbitrary strings potentially containing multiple lines.
- `--ostype <ostype>`: Specifies what guest operating system is supposed to run in the VM. To learn about the various identifiers that can be used here, use `VBoxManage list ostypes`.
- `--iconfile <filename>`: Specifies the absolute path on the host file system for the Oracle VM VirtualBox icon to be displayed in the VM.

- `--memory <memorysize>`: Sets the amount of RAM, in MB, that the virtual machine should allocate for itself from the host. See chapter 1.8, *Creating Your First Virtual Machine*, page 8.
- `--pagefusion on|off`: Enables and disables the Page Fusion feature. Page Fusion is disabled by default. The Page Fusion feature minimises memory duplication between VMs with similar configurations running on the same host. See chapter 4.10.2, *Page Fusion*, page 81.
- `--vram <vramsize>`: Sets the amount of RAM that the virtual graphics card should have. See chapter 3.6, *Display Settings*, page 50.
- `--acpi on|off` and `--ioapic on|off`: Determines whether the VM has ACPI and I/O APIC support. See chapter 3.5.1, *Motherboard Tab*, page 47.
- `--pciattach <host PCI address [@ guest PCI bus address]>`: Attaches a specified PCI network controller on the host to a specified PCI bus on the guest. See chapter 9.5, *PCI Passthrough*, page 213.
- `--pcidetach <host PCI address>`: Detaches a specified PCI network controller on the host from the attached PCI bus on the guest. See chapter 9.5, *PCI Passthrough*, page 213.
- `--hardwareuuid <uuid>`: The UUID presented to the guest through memory tables (DMI/SMBIOS), hardware, and guest properties. By default this is the same as the VM UUID. This setting is useful when cloning a VM. Teleporting takes care of this automatically.
- `--cpus <cpucount>`: Sets the number of virtual CPUs for the virtual machine, see chapter 3.5.2, *Processor Tab*, page 48. If CPU hot-plugging is enabled, this then sets the *maximum* number of virtual CPUs that can be plugged into the virtual machines.
- `--cpuhotplug on|off`: Enables CPU hot-plugging. When enabled, virtual CPUs can be added to and removed from a virtual machine while it is running. See chapter 9.4, *CPU Hot-Plugging*, page 212.
- `--plugcpu|unplugcpu <id>`: If CPU hot-plugging is enabled, this setting adds or removes a virtual CPU on the virtual machine. `<id>` specifies the index of the virtual CPU to be added or removed and must be a number from 0 to the maximum number of CPUs configured with the `--cpus` option. CPU 0 can never be removed.
- `--cpuexecutioncap <1-100>`: Controls how much CPU time a virtual CPU can use. A value of 50 implies a single virtual CPU can use up to 50% of a single host CPU.
- `--pae on|off`: Enables and disables PAE. See chapter 3.5.2, *Processor Tab*, page 48.
- `--longmode on|off`: Enables and disables long mode. See chapter 3.5.2, *Processor Tab*, page 48.
- `--spec-ctrl on|off`: Enables and disables the exposure of speculation control interfaces to the guest, provided they are available on the host. Depending on the host CPU and workload, enabling speculation control may significantly reduce performance.
- `--cpu-profile <host|intel 80[86|286|386]>`: Enables specification of a profile for guest CPU emulation. Specify either one based on the host system CPU (host), or one from a number of older Intel Micro-architectures: 8086, 80286, 80386.
- `--hpet on|off`: Enables and disables a High Precision Event Timer (HPET) which can replace the legacy system timers. This is turned off by default. Note that Windows supports a HPET only from Vista onwards.

- `--hwvirtex on|off`: Enables and disables the use of hardware virtualization extensions, such as Intel VT-x or AMD-V, in the processor of your host system. See chapter 10.3, *Hardware vs. Software Virtualization*, page 270.

- `--triplefaultreset on|off`: Enables resetting of the guest instead of triggering a Guru Meditation. Some guests raise a triple fault to reset the CPU so sometimes this is desired behavior. Works only for non-SMP guests.

- `--apic on|off`: Enables and disables I/O APIC. With I/O APIC, operating systems can use more than 16 interrupt requests (IRQs) thus avoiding IRQ sharing for improved reliability. This setting is enabled by default. See chapter 3.5.1, *Motherboard Tab*, page 47.

- `--x2apic on|off`: Enables and disables CPU x2APIC support. CPU x2APIC support helps operating systems run more efficiently on high core count configurations, and optimizes interrupt distribution in virtualized environments. This setting is enabled by default. Disable this setting when using host or guest operating systems that are incompatible with x2APIC support.

- `--paravirtprovider none|default|legacy|minimal|hyperv|kvm`: Specifies which paravirtualization interface to provide to the guest operating system. Specifying `none` explicitly turns off exposing any paravirtualization interface. The option `default` selects an appropriate interface when starting the VM, depending on the guest OS type. This is the default option chosen when creating new VMs. The `legacy` option is used for VMs which were created with older Oracle VM VirtualBox versions and will pick a paravirtualization interface when starting the VM with Oracle VM VirtualBox 5.0 and newer. The `minimal` provider is mandatory for Mac OS X guests. `kvm` and `hyperv` are recommended for Linux and Windows guests respectively. These options are explained in chapter 10.4, *Paravirtualization Providers*, page 271.

- `--paravirtdebug <keyword=value> [,<keyword=value> ...]`: Specifies debugging options specific to the paravirtualization provider configured for this VM. See the provider specific options in chapter 9.31, *Paravirtualized Debugging*, page 253 for a list of supported keyword-value pairs for each provider.

- `--nestedpaging on|off`: If hardware virtualization is enabled, this additional setting enables or disables the use of the nested paging feature in the processor of your host system. See chapter 10.3, *Hardware vs. Software Virtualization*, page 270 and chapter 13.4.1, *CVE-2018-3646*, page 299.

- `--largepages on|off`: If hardware virtualization *and* nested paging are enabled, for Intel VT-x only, an additional performance improvement of up to 5% can be obtained by enabling this setting. This causes the hypervisor to use large pages to reduce TLB use and overhead.

- `--vtxvpid on|off`: If hardware virtualization is enabled, for Intel VT-x only, this additional setting enables or disables the use of the tagged TLB (VPID) feature in the processor of your host system. See chapter 10.3, *Hardware vs. Software Virtualization*, page 270.

- `--vtxux on|off`: If hardware virtualization is enabled, for Intel VT-x only, this setting enables or disables the use of the unrestricted guest mode feature for executing your guest.

- `--accelerate3d on|off`: If the Guest Additions are installed, this setting enables or disables hardware 3D acceleration. See chapter 4.5.1, *Hardware 3D Acceleration (OpenGL and Direct3D 8/9)*, page 74.

- `--accelerate2dvideo on|off`: If the Guest Additions are installed, this setting enables or disables 2D video acceleration. See chapter 4.5.2, *Hardware 2D Video Acceleration for Windows Guests*, page 75.

- `--chipset piix3|ich9`: By default, Oracle VM VirtualBox emulates an Intel PIIX3 chipset. Usually there is no reason to change the default setting unless this is required to relax some of its constraints. See chapter 3.5.1, *Motherboard Tab*, page 47.

- You can influence the BIOS logo that is displayed when a virtual machine starts up with a number of settings. By default, an Oracle VM VirtualBox logo is displayed.

  With `--bioslogofadein on|off` and `--bioslogofadeout on|off`, you can determine whether the logo should fade in and out, respectively.

  With `--bioslogodisplaytime <msec>` you can set how long the logo should be visible, in milliseconds.

  With `--bioslogoimagepath <imagepath>` you can replace the image that is shown with your own logo. The image must be an uncompressed 256 color BMP file without color space information (Windows 3.0 format). The image must not be bigger than 640 x 480.

- `--biosbootmenu disabled|menuonly|messageandmenu`: Specifies whether the BIOS enables the user to select a temporary boot device. The `menuonly` option suppresses the message, but the user can still press F12 to select a temporary boot device.

- `--biosapic x2apic|apic|disabled`: Specifies the firmware APIC level to be used. Options are: x2apic, apic or disabled (no apic or x2apic) respectively.

  Note that if x2apic is specified and x2APIC is unsupported by the VCPU, biosapic downgrades to apic, if supported. Otherwise biosapic downgrades to disabled. Similarly, if apic is specified, and APIC is unsupported, a downgrade to disabled results.

- `--biossystemtimeoffset <ms>`: Specifies a fixed time offset, in milliseconds, of the guest relative to the host time. If the offset is positive, the guest time runs ahead of the host time.

- `--biospxedebug on|off`: Enables additional debugging output when using the Intel PXE boot ROM. The output is written to the release log file. See chapter 12.1.2, *Collecting Debugging Information*, page 278.

- `--boot<1-4> none|floppy|dvd|disk|net`: Specifies the boot order for the virtual machine. There are four *slots*, which the VM will try to access from 1 to 4, and for each of which you can set a device that the VM should attempt to boot from.

- `--rtcuseutc on|off`: Sets the real-time clock (RTC) to operate in UTC time. See chapter 3.5.1, *Motherboard Tab*, page 47.

- `--graphicscontroller none|vboxvga|vmsvga|vboxsvga`: Specifies the use of a graphics controller, with an option to choose a specific type. See chapter 3.6.1, *Screen Tab*, page 50.

- `--snapshotfolder default|<path>`: Specifies the folder where snapshots are kept for a virtual machine.

- `--firmware bios|efi|efi32|efi64`: Specifies the firmware to be used to boot the VM: Available options are: BIOS, or one of the EFI options: efi, efi32, or efi64. Use EFI options with care.

- `--guestmemoryballoon <size>` Sets the default size of the guest memory balloon. This is the memory allocated by the Oracle VM VirtualBox Guest Additions from the guest operating system and returned to the hypervisor for reuse by other virtual machines. `<size>` must be specified in megabytes. The default size is 0 megabytes. See chapter 4.10.1, *Memory Ballooning*, page 80.

- `--defaultfrontend default|<name>`: Specifies the default frontend to be used when starting this VM. See chapter 8.13, *VBoxManage startvm*, page 151.

### 8.8.2 Networking Settings

The following networking settings are available through `VBoxManage modifyvm`. With all these settings, the decimal number directly following the option name, 1-N in the list below, specifies the virtual network adapter whose settings should be changed.

- `--nic<1-N> none|null|nat|natnetwork|bridged|intnet|hostonly|generic`: Configures the type of networking for each of the VM's virtual network cards. Options are: not present (`none`), not connected to the host (`null`), use network address translation (`nat`), use the new network address translation engine (`natnetwork`), bridged networking (`bridged`), or use internal networking (`intnet`), host-only networking (`hostonly`), or access rarely used sub-modes (`generic`). These options correspond to the modes described in chapter 6.2, *Introduction to Networking Modes*, page 99.

- `--nictype<1-N> Am79C970A|Am79C973|82540EM|82543GC|82545EM|virtio`: Enables you to specify the networking hardware that Oracle VM VirtualBox presents to the guest for a specified VM virtual network card. See chapter 6.1, *Virtual Networking Hardware*, page 98.

- `--cableconnected<1-N> on|off`: Enables you to temporarily disconnect a virtual network interface, as if a network cable had been pulled from a real network card. This might be useful, for example for resetting certain software components in the VM.

- With the `nictrace` options, you can optionally trace network traffic by dumping it to a file, for debugging purposes.

  With `--nictrace<1-N> on|off`, you can enable network tracing for a particular virtual network card.

  If enabled, you must specify with `--nictracefile<1-N> <filename>` the absolute path of the file the trace should be logged to.

- `--nicproperty<1-N> <paramname>="paramvalue"`: This option, in combination with `nicgenericdrv` enables you to pass parameters to rarely-used network backends.

  These parameters are backend engine-specific, and are different between UDP Tunnel and the VDE backend drivers. For examples, see chapter 6.8, *UDP Tunnel Networking*, page 106.

- `--nicspeed<1-N> <kbps>`: Only has an effect if generic networking has been enabled for a particular virtual network card. See the `--nic` option. This mode enables access to rarely used networking sub-modes, such as VDE network or UDP Tunnel. This option specifies the throughput rate in KBps.

- `--nicbootprio<1-N> <priority>`: Specifies the order in which NICs are tried for booting over the network, using PXE. The priority is an integer in the 0 to 4 range. Priority 1 is the highest, priority 4 is low. Priority 0, which is the default unless otherwise specified, is the lowest.

  Note that this option only has an effect when the Intel PXE boot ROM is used.

- `--nicpromisc<1-N> deny|allow-vms|allow-all`: Enables you to specify how promiscuous mode is handled for the specified VM virtual network card. This setting is only relevant for bridged networking. `deny`, the default setting, hides any traffic not intended for the VM. `allow-vms` hides all host traffic from the VM, but allows the VM to see traffic to and from other VMs. `allow-all` removes this restriction completely.

- `--nicbandwidthgroup<1-N> none|<name>`: Adds and removes an assignment of a bandwidth group for the specified virtual network interface. Specifying `none` removes any current bandwidth group assignment from the specified virtual network interface. Specifying `<name>` adds an assignment of a bandwidth group to the specified virtual network interface.

  See chapter 6.10, *Limiting Bandwidth for Network Input/Output*, page 108.

- `--bridgeadapter<1-N> none|<devicename>`: Only has an effect if bridged networking has been enabled for a virtual network card. See the `--nic` option. Use this option to specify which host interface the given virtual network interface will use. See chapter 6.5, *Bridged Networking*, page 103.
- `--hostonlyadapter<1-N> none|<devicename>`: Only has an effect if host-only networking has been enabled for a virtual network card. See the `--nic` option. Use this option to specify which host-only networking interface the given virtual network interface will use. See chapter 6.7, *Host-Only Networking*, page 105.
- `--intnet<1-N> network`: Only has an effect if internal networking has been enabled for a virtual network card. See the `--nic` option. Use this option to specify the name of the internal network. See chapter 6.6, *Internal Networking*, page 104.
- `--nat-network<1-N> <network name>`: If the networking type is set to `natnetwork`, not `nat`, then this setting specifies the name of the NAT network this adapter is connected to. Optional.
- `--nicgenericdrv<1-N> <backend driver>`: Only has an effect if generic networking has been enabled for a virtual network card. See the `--nic` option. This mode enables you to access rarely used networking sub-modes, such as VDE network or UDP Tunnel.
- `--macaddress<1-N> auto|<mac>`: With this option you can set the MAC address of a particular network adapter on the VM. Normally, each network adapter is assigned a random address by Oracle VM VirtualBox at VM creation.

#### 8.8.2.1 NAT Networking Settings

The following NAT networking settings are available through `VBoxManage modifyvm`. With all these settings, the decimal number directly following the option name, 1-N in the list below, specifies the virtual network adapter whose settings should be changed.

- `--natnet<1-N> <network>|default`: If the networking type is set to `nat`, not `natnetwork`, then this setting specifies the IP address range to be used for this network. See chapter 9.10, *Fine Tuning the Oracle VM VirtualBox NAT Engine*, page 221.
- `--natpf<1-N> [<name>],tcp|udp,[<hostip>],<hostport>,[<guestip>], <guestport>`: Defines a NAT port-forwarding rule. See chapter 6.3.1, *Configuring Port Forwarding with NAT*, page 100.
- `--natpf<1-N> delete <name>`: Deletes a NAT port-forwarding rule. See chapter 6.3.1, *Configuring Port Forwarding with NAT*, page 100.
- `--nattftpprefix<1-N> <prefix>`: Defines a prefix for the built-in TFTP server. For example, where the boot file is located. See chapter 6.3.2, *PXE Booting with NAT*, page 101 and chapter 9.10.2, *Configuring the Boot Server (Next Server) of a NAT Network Interface*, page 221.
- `--nattftpfile<1-N> <bootfile>`: Defines the TFT boot file. See chapter 9.10.2, *Configuring the Boot Server (Next Server) of a NAT Network Interface*, page 221.
- `--nattftpserver<1-N> <tftpserver>`: Defines the TFTP server address to boot from. See chapter 9.10.2, *Configuring the Boot Server (Next Server) of a NAT Network Interface*, page 221.
- `--nattbindip<1-N> <ip;>`: Oracle VM VirtualBox's NAT engine normally routes TCP/IP packets through the default interface assigned by the host's TCP/IP stack. Use this setting to instruct the NAT engine to bind to a specified IP address instead. See chapter 9.10.3, *Tuning TCP/IP Buffers for NAT*, page 221.

- `--natdnspassdomain<1-N> on|off`: Specifies whether the built-in DHCP server passes the domain name for network name resolution.
- `--natdnsproxy<1-N> on|off`: Makes the NAT engine proxy all guest DNS requests to the host's DNS servers. See chapter 9.10.5, *Enabling DNS Proxy in NAT Mode*, page 222.
- `--natdnshostresolver<1-N> on|off`: Makes the NAT engine use the host's resolver mechanisms to handle DNS requests. See chapter 9.10.5, *Enabling DNS Proxy in NAT Mode*, page 222.
- `--natsettings<1-N> [<mtu>],[<socksnd>],[<sockrcv>],[<tcpsnd>], [<tcprcv>]`: Controls several NAT settings. See chapter 9.10.3, *Tuning TCP/IP Buffers for NAT*, page 221.
- `--nataliasmode<1-N> default|[log],[proxyonly],[sameports]`: Defines behaviour of the NAT engine core: log - enables logging, proxyonly - switches off aliasing mode and makes NAT transparent, sameports - enforces the NAT engine to send packets through the same port as they originated on, default - disable all aliasing modes. See chapter 9.10.7, *Configuring Aliasing of the NAT Engine*, page 223.

### 8.8.3 Miscellaneous Settings

The following hardware settings, such as serial port, audio, clipboard, drag and drop, monitor, and USB settings are available through `VBoxManage modifyvm`:

- `--mouse <ps2|usb|usbtablet|usbmultitouch>`: Specifies the mode of the mouse to be used in the VM. Available options are: ps2, usb, usbtablet, usbmultitouch.
- `--keyboard <ps2|usb>`: Specifies the mode of the keyboard to be used in the VM. Available options are: ps2, usb.
- `--uart<1-N> off|<I/O base> <IRQ>`: Configures virtual serial ports for the VM. See chapter 3.10, *Serial Ports*, page 54.
- `--uartmode<1-N> <arg>`: Controls how Oracle VM VirtualBox connects a given virtual serial port, configured with the `--uartX` setting, to the host on which the virtual machine is running. As described in chapter 3.10, *Serial Ports*, page 54, for each such port, you can specify `<arg>` as one of the following options:
  - `disconnected`: Even though the serial port is shown to the guest, it has no "other end". This is like a real COM port without a cable.
  - `server <pipename>`: On a Windows host, this tells Oracle VM VirtualBox to create a named pipe on the host named `<pipename>` and connect the virtual serial device to it. Note that Windows requires that the name of a named pipe begins with `\\.\pipe\`.

    On a Linux host, instead of a named pipe, a local domain socket is used.
  - `client <pipename>`: Operates as for `server`, except that the pipe, or local domain socket, is not created by Oracle VM VirtualBox but is assumed to exist already.
  - `tcpserver <port>`: Configures Oracle VM VirtualBox to create a TCP socket on the host with TCP `<port>` and connect the virtual serial device to it. Note that UNIX-like systems require ports over 1024 for normal users.
  - `tcpclient <hostname:port>`: Operates as for `tcpserver`, except that the TCP socket is not created by Oracle VM VirtualBox, but is assumed to exist already.
  - `uarttype <1-N> 16450|16550A|16750`: Confgures the UART type for a virtual serial port. The default UART type is 16550A.

  - `file <file>`: Redirects the serial port output to a raw file <file> specified by its absolute path on the host file system.
  - `<devicename>`: If, instead of the above options, the device name of a physical hardware serial port of the host is specified, the virtual serial port is connected to that hardware port. On a Windows host, the device name will be a COM port such as `COM1`. On a Linux host, the device name will be `/dev/ttyS0` or similar. This enables you to wire up a real serial port to a virtual machine.

- `--lptmode<1-N> <Device>`: Specifies the Device Name of the parallel port that the Parallel Port feature will be using. Use this *before* `--lpt`. This feature depends on the host operating system. For Windows hosts, use a device name such as lpt1. On Linux hosts, use a device name such as /dev/lp0.

- `--lpt<1-N> <I/O base> <IRQ>`: Specifies the I/O address of the parallel port and the IRQ number that the Parallel Port feature will be using. Optional. Use this *after* `--lptmod`. I/O base address and IRQ are the values that guest sees. For example, the values avalable under guest Device Manager.

- `--audio none|null|dsound|oss|alsa|pulse|coreaudio`: Specifies whether the VM should have audio support, and if so, which type. The list of supported audio types depends on the host and can be determined with `VBoxManage modifyvm`.

- `--audiocontroller ac97|hda|sb16`: Specifies the audio controller to be used with the VM.

- `--audiocodec stac9700|ad1980|stac9221|sb16`: Specifies the audio codec to be used with the VM.

- `--audioin on`: Specifies whether capturing audio from the host is enabled or disabled.

- `--audioout on`: Specifies whether audio playback from the guest is enabled or disabled.

- `--clipboard disabled|hosttoguest|guesttohost|bidirectional`: Configues how the guest or host operating system's clipboard should be shared with the host or guest. See chapter 3.4, *General Settings*, page 45. This setting requires that the Guest Additions be installed in the virtual machine.

- `--draganddrop disabled|hosttoguest|guesttohost|bidirectional`: Specifies the drag and drop mode to use between the host and the virtual machine. See chapter 4.4, *Drag and Drop*, page 72. This requires that the Guest Additions be installed in the virtual machine.

- `--monitorcount <count>`: Enables multi-monitor support. See chapter 3.6, *Display Settings*, page 50.

- `--usb on|off`: Enables and disables the VM's virtual USB controller. See chapter 3.11.1, *USB Settings*, page 56.

- `--usbehci on|off`: Enables and disables the VM's virtual USB 2.0 controller. See chapter 3.11.1, *USB Settings*, page 56.

- `--usbxhci on|off`: Enables and disables the VM's virtual USB 3.0 controller. See chapter 3.11.1, *USB Settings*, page 56.

- `--usbrename <oldname> <newname>`: Enables renaming of the VM's virtual USB controller from <oldname> to <newname>.

### 8.8.4 Recording Settings

The `VBoxManage modifyvm` command enables you to modify recording settings for video recording, audio recording, or both.

Use the following options to update the recording settings:

- `--recording on|off` enables or disables the recording of a VM session into a WebM/VP8 file. When this option value is `on`, recording begins when the VM session starts.
- `--recordingscreens all|<screen-ID> [<screen-ID> ...]` enables you to specify which VM screens to record. The recording for each screen that you specify is saved to its own file.
- `--recordingfile <filename>` specifies the file in which to save the recording.
- `--recordingmaxsize <MB>` specifies the maximum size of the recorded video file in megabytes. The recording stops when the file reaches the specified size. If this value is zero, the recording continues until you stop the recording.
- `--recordingmaxtime <seconds>` specifies the maximum amount time to record in seconds. The recording stops after the specified number of seconds elapses. If this value is zero, the recording continues until you stop the recording.
- `--recordingopts <keyword>=<value>[,<keyword>=<value> ...]` specifies additional video-recording options in a comma-separated keyword-value format. For example, `foo=bar,a=b`.

  Only use this option only if you are an advanced user. For information about keywords, see *Oracle VM VirtualBox Programming Guide and Reference*.
- `--recordingvideofps <fps>` specifies the maximum number of video frames per second (FPS) to record. Frames that have a higher frequency are skipped. Increasing this value reduces the number of skipped frames and increases the file size.
- `--recordingvideorate <bit-rate>` specifies the bit rate of the video in kilobits per second. Increasing this value improves the appearance of the video at the cost of an increased file size.
- `--recordingvideores <width>x<height>` specifies the video resolution of the recorded video in pixels.

### 8.8.5 Remote Machine Settings

The following settings that affect remote machine behavior are available through `VBoxManage modifyvm`:

- `--vrde on|off`: Enables and disables the VirtualBox Remote Desktop Extension (VRDE) server.
- `--vrdeproperty "TCP/Ports|Address=<value>"`: Sets the port numbers and IP address on the VM that the VRDE server can bind to.
  - For TCP/Ports, <value> should be a port or a range of ports that the VRDE server can bind to. `default` or `0` means port 3389, the standard port for RDP. See the description for the `--vrdeport` option in chapter 8.8.5, *Remote Machine Settings*, page 143.
  - For TCP/Address, <value> should be the IP address of the host network interface that the VRDE server will bind to. If specified, the server will accept connections only on the specified host network interface. See the description for the `--vrdeaddress` option in chapter 8.8.5, *Remote Machine Settings*, page 143.

- `--vrdeproperty "VideoChannel/Enabled|Quality|DownscaleProtection=<value>"`: Sets the VRDP video redirection properties.
  - For VideoChannel/Enabled, <value> can be set to "1", switching the VRDP video channel on. See chapter 7.1.9, *VRDP Video Redirection*, page 117.
  - For VideoChannel/Quality, <value> should be set between 10 and 100% inclusive, representing a JPEG compression level on the VRDE server video channel. Lower values mean lower quality but higher compression. See chapter 7.1.9, *VRDP Video Redirection*, page 117.
  - For VideoChannel/DownscaleProtection, <value> can be set to "1" to enable the videochannel downscale protection feature. When enabled, if a video's size equals the shadow buffer size, then it is regarded as a full screen video, and is displayed. But if its size is between fullscreen and the downscale threshold then it is *not* displayed, as it could be an application window, which would be unreadable when downscaled. When the downscale protection feature is disabled, an attempt is always made to display videos.
- `--vrdeproperty "Client/DisableDisplay|DisableInput|DisableAudio|DisableUSB=1"`: Disables one of the VRDE server features: Display, Input, Audio or USB respectively. To reenable a feature, use "Client/DisableDisplay=" for example. See chapter 7.1.10, *VRDP Customization*, page 118.
- `--vrdeproperty "Client/DisableClipboard|DisableUpstreamAudio=1"`: Disables one of the VRDE server features: Clipboard or UpstreamAudio respectively. To reenable a feature, use "Client/DisableClipboard=" for example. See chapter 7.1.10, *VRDP Customization*, page 118.
- `--vrdeproperty "Client/DisableRDPDR=1"`: Disables the VRDE server feature: RDP device redirection for smart cards. To reenable this feature, use "Client/DisableRDPR=".
- `--vrdeproperty "H3DRedirect/Enabled=1"`: Enables the VRDE server feature: 3D redirection. To disable this feature, use "H3DRedirect/Enabled=".
- `--vrdeproperty "Security/Method|ServerCertificate|ServerPrivateKey|CACertificate=<value>"` Sets the desired security method and path of server certificate, path of server private key, path of CA certificate, that are used for a connection.
  - `--vrdeproperty "Security/Method=<value>"` sets the desired security method, which is used for a connection. Valid values are:
    - `Negotiate`: Both Enhanced (TLS) and Standard RDP Security connections are allowed. The security method is negotiated with the client. This is the default setting.
    - `RDP`: Only Standard RDP Security is accepted.
    - `TLS`: Only Enhanced RDP Security is accepted. The client must support TLS.

    See chapter 7.1.6, *RDP Encryption*, page 116.
  - `--vrdeproperty "Security/ServerCertificate=<value>"` where <value> is the absolute path of the server certificate. See chapter 7.1.6, *RDP Encryption*, page 116.
  - `--vrdeproperty "Security/ServerPrivateKey=<value>"` where <value> is the absolute path of the server private key. See chapter 7.1.6, *RDP Encryption*, page 116.
  - `--vrdeproperty "Security/CACertificate=<value>"` where <value> is the absolute path of the CA self signed certificate. See chapter 7.1.6, *RDP Encryption*, page 116.

- `--vrdeproperty "Audio/RateCorrectionMode|LogPath=<value>"` sets the audio connection mode, or path of the audio logfile.
  - `--vrdeproperty "Audio/RateCorrectionMode=<value>"` where <value> is the desired rate correction mode. Allowed values are:
    - `VRDP_AUDIO_MODE_VOID`: No mode specified, use to unset any Audio mode already set.
    - `VRDP_AUDIO_MODE_RC`: Rate correction mode.
    - `VRDP_AUDIO_MODE_LPF`: Low pass filter mode.
    - `VRDP_AUDIO_MODE_CS`: Client sync mode to prevent underflow or overflow of the client queue.
  - `--vrdeproperty "Audio/LogPath=<value>"` where <value> is the absolute path of the Audio log file.
- `--vrdeextpack default|<name>`: Specifies the library to use for accessing the VM remotely. The default is to use the RDP code which is part of the Oracle VM VirtualBox Extension Pack.
- `--vrdeport default|<ports>`: A port or a range of ports the VRDE server can bind to. `default` or `0` means port 3389, the standard port for RDP. You can specify a comma-separated list of ports or ranges of ports. Use a dash between two port numbers to specify a range. The VRDE server will bind to *one* of the available ports from the specified list. Only one machine can use a given port at a time. For example, the option `--vrdeport 5000,5010-5012` will tell the server to bind to one of following ports: 5000, 5010, 5011, or 5012.
- `--vrdeaddress <IP address>`: The IP address of the host network interface the VRDE server will bind to. If specified, the server will accept connections only on the specified host network interface.

  The setting can be used to specify whether the VRDP server should accept either IPv4, IPv6, or both connections:
  - Only IPv4: `--vrdeaddress "0.0.0.0"`
  - Only IPv6: `--vrdeaddress "::"`
  - Both IPv6 and IPv4: `--vrdeaddress ""`

    This is the default setting.
- `--vrdeauthtype null|external|guest`: Enables you to indicate use of authorization, and specify how authorization will be performed. See chapter 7.1.5, *RDP Authentication*, page 115.
- `--vrdeauthlibrary default|<name>`: Specifies the library used for RDP authentication. See chapter 7.1.5, *RDP Authentication*, page 115.
- `--vrdemulticon on|off`: Enables multiple connections to be made to the same VRDE server, if the server supports this feature. See chapter 7.1.7, *Multiple Connections to the VRDP Server*, page 117.
- `--vrdereusecon on|off`: This specifies the VRDE server behavior when multiple connections are disabled. When this option is enabled, the server will allow a new client to connect and will drop the existing connection. When this option is disabled, the default setting, a new connection will not be accepted if there is already a client connected to the server.

- `--vrdevideochannel on|off`: Enables video redirection, if it is supported by the VRDE server. See chapter 7.1.9, *VRDP Video Redirection*, page 117.
- `--vrdevideochannelquality <percent>`: Specifies the image quality for video redirection. See chapter 7.1.9, *VRDP Video Redirection*, page 117.

### 8.8.6 Teleporting Settings

With the following commands for `VBoxManage modifyvm` you can configure a machine to be a target for teleporting. See chapter 7.2, *Teleporting*, page 118.

- `--teleporter on|off`: Enables and disables the teleporter feature whereby when the machine is started, it waits to receive a teleporting request from the network instead of booting normally. Teleporting requests are received on the port and address specified using the following parameters.
- `--teleporterport <port>`, `--teleporteraddress <address>`: These settings must be used with `--teleporter`. They specify the port and address the virtual machine should listen to in order to receive a teleporting request sent from another virtual machine. `<port>` can be any free TCP/IP port number, such as 6000. `<address>` can be any IP address or hostname and specifies the TCP/IP socket to bind to. The default is 0.0.0.0, which means any address.
- `--teleporterpassword <password>`: If this optional setting is used, then the teleporting request will only succeed if the source machine specifies the same password as the one given with this command.
- `--teleporterpasswordfile <password>`: If this optional setting is used, then the teleporting request will only succeed if the source machine specifies the same password as the one specified in the file give with this command. Use `stdin` to read the password from stdin.
- `--cpuid <leaf> <eax> <ebx> <ecx> <edx>`: Advanced users can use this setting before a teleporting operation, to restrict the virtual CPU capabilities that Oracle VM VirtualBox presents to the guest operating system. This must be run on both the source and the target machines involved in the teleporting and will then modify what the guest sees when it executes the `CPUID` machine instruction. This might help with misbehaving applications that wrongly assume that certain CPU capabilities are present. The meaning of the parameters is hardware dependent, refer to the AMD or Intel processor documentation.

### 8.8.7 Debugging Settings

The following settings are only relevant for low-level VM debugging. Regular users will never need these settings.

- `--tracing-enabled on|off`: Enables the tracebuffer. This consumes some memory for the tracebuffer and adds extra overhead.
- `--tracing-config <config-string>`: Enables tracing configuration. In particular, this defines which group of tracepoints are enabled.
- `--tracing-allow-vm-access on|off`: Enables and disables VM access to the tracebuffer. By default, this setting is disabled.

### 8.8.8 USB Card Reader Settings

The following setting defines access to a USB Card Reader by the guest environment. USB card readers are typically used for accessing data on memory cards such as CompactFlash (CF), Secure Digital (SD), or MultiMediaCard (MMC).

- `--usbcardreader on|off`: Enables and disables the USB card reader interface.

### 8.8.9 Autostarting VMs During Host System Boot

These settings configure the VM autostart feature, which automatically starts the VM at host system boot-up. Note that there are prerequisites that need to be addressed before using this feature. See chapter 9.23, *Starting Virtual Machines During System Boot*, page 248.

- `--autostart-enabled on|off`: Enables and disables VM autostart at host system boot-up, using the specified user name.
- `--autostart-delay <seconds>`: Specifies a delay, in seconds, following host system boot-up, before the VM autostarts.

## 8.9 VBoxManage clonevm

The `VBoxManage clonevm` command creates a clone of an existing virtual machine (VM). The clone can be a full copy of the VM or a linked copy of a VM.

```
VBoxManage clonevm <vm> [ --basefolder <basefolder> ]
[ --group <group>, ... ] [ --mode machine | machinechildren | all ]
[ --name <name> ] [ --options link | keepallmacs | keepnatmacs | keepdisknames | keephwuuids ]
[ --register ] [ --snapshot <vm> ] [ --uuid <uuid> ]
```

In addition to specifying the name of the VM to clone, which is required, you can specify any of the following options:

- `--basefolder` *`basefolder`* specifies the name of the folder in which to save the configuration for the new VM.
- `--groups <group>, ...` assigns the clone to the specified group or groups. If you specify more than one group, separate each group name with a comma.

  Note that each group is identified by a group ID that starts with a slash character (`/`) so that groups can be nested. By default, a clone is always assigned membership to the `/` group.
- `--mode machine|machineandchildren|all` specifies which of the following cloning modes to use:
  - `machine` mode clones the current state of the existing VM without any snapshots. This is the default mode.
  - `machineandchildren` mode clones the snapshot specified by by the `--snapshot` option and all child snapshots.
  - `all` mode clones all snapshots and the current state of the existing VM.
- `--name <name>` specifies a new name for the new VM. The default value is "*name* Clone", where *name* is the original name of the VM.
- `--options` specifies how to create a new clone.

  - `--options link` creates a linked clone, which can be cloned only from a snapshot.
  - `--options keepallmacs` specifies that the new clone reuses the MAC addresses of each virtual network card from the existing VM.

    If you do not specify this option or the `--options keepnatmacs` option, the default behavior is to reinitialize the MAC addresses of each virtual network card.
  - `--options keepnatmacs` specifies that the new clone reuses the MAC addresses of each virtual network card from the existing VM when the network type is NAT.

    If you do not specify this option or the `--options keepallmacs` option, the default behavior is to reinitialize the MAC addresses of each virtual network card.
  - `--option keepdisknames` specifies that the new clone reuses the disk image names from the existing VM. By default, disk images are renamed. You can preserve source hardware IDs by adding `keephwuuids`.
  - `--option keephwuuids` specifies that the new clone reuses the hardware IDs from the existing VM. By default, new UUIDs are used.

- `--register` automatically registers the new clone in this Oracle VM VirtualBox installation. You can manually register the new VM later by using the `VBoxManage registervm` command. See chapter 8.6, *VBoxManage registervm/unregistervm*, page 134.
- `--snapshot <vm>` specifies the snapshot on which to base the new VM. By default, the clone is created from the current state of the specified VM.
- `--uuid <uuid>` specifies the UUID for the new VM. Ensure that this ID is unique for the Oracle VM VirtualBox instance if you decide to register this new VM. By default, Oracle VM VirtualBox provides a new UUID.

## 8.10 VBoxManage movevm

This command moves a virtual machine to a new location on the host.

Associated files of the virtual machine, such as settings files and disk image files, are moved to the new location. The Oracle VM VirtualBox configuration is updated automatically.

The `movevm` subcommand requires the name of the virtual machine which should be moved.

Also required is the type of move operation, specified by `--type basic`. Other types of move operation may be supported in future releases.

The `--folder` setting configures the new location on the host file system. Enter a relative pathname or a full pathname.

## 8.11 VBoxManage import

This command imports a virtual appliance in OVF format by copying the virtual disk images and creating virtual machines in Oracle VM VirtualBox. See chapter 1.15, *Importing and Exporting Virtual Machines*, page 21 for an introduction to appliances.

The `import` subcommand takes at least the path name of an OVF file as input and expects the disk images, if needed, in the same directory as the OVF file. A lot of additional command-line options are supported to control in detail what is being imported and modify the import parameters, but the details depend on the content of the OVF file.

It is therefore recommended to first run the `import` subcommand with the `--dry-run` or `-n` option. This will then print a description of the appliance's contents to the screen how it would be imported into Oracle VM VirtualBox, together with the optional command-line options to influence the import behavior.

Use of the `--options keepallmacs|keepnatmacs|keepdisknames` option enables additional fine tuning of the clone operation. The first two options enable specification of how the MAC addresses of every virtual network card should be handled. They can either be reinitialized, which is the default setting, left unchanged (`keepallmacs`) or left unchanged when the network type is NAT (`keepnatmacs`). If you add `keepdisknames` all new disk images are assigned the same names as the originals, otherwise they are renamed.

As an example, the following is a screen output for a sample appliance containing a Windows XP guest:

```
VBoxManage import WindowsXp.ovf --dry-run
Interpreting WindowsXp.ovf...
OK.
Virtual system 0:
 0: Suggested OS type: "WindowsXP"
    (change with "--vsys 0 --ostype <type>"; use "list ostypes" to list all)
 1: Suggested VM name "Windows XP Professional_1"
    (change with "--vsys 0 --vmname <name>")
 2: Suggested VM group "/"
    (change with "--vsys 0 --group <group>")
 3: Suggested VM settings file name "/home/klaus/VirtualBox VMs/dummy2 2/dummy2 2.vbox"
    (change with "--vsys 0 --settingsfile <filename>")
 4: Suggested VM base folder "/home/klaus/VirtualBox VMs"
    (change with "--vsys 0 --basefolder <path>")
 5: End-user license agreement
    (display with "--vsys 0 --eula show";
    accept with "--vsys 0 --eula accept")
 6: Number of CPUs: 1
    (change with "--vsys 0 --cpus <n>")
 7: Guest memory: 956 MB (change with "--vsys 0 --memory <MB>")
 8: Sound card (appliance expects "ensoniq1371", can change on import)
    (disable with "--vsys 0 --unit 5 --ignore")
 9: USB controller
    (disable with "--vsys 0 --unit 6 --ignore")
10: Network adapter: orig bridged, config 2, extra type=bridged
11: Floppy
    (disable with "--vsys 0 --unit 8 --ignore")
12: SCSI controller, type BusLogic
    (change with "--vsys 0 --unit 9 --scsitype {BusLogic|LsiLogic}";
    disable with "--vsys 0 --unit 9 --ignore")
13: IDE controller, type PIIX4
    (disable with "--vsys 0 --unit 10 --ignore")
14: Hard disk image: source image=WindowsXp.vmdk,
      target path=/home/user/disks/WindowsXp.vmdk, controller=9;channel=0
    (change controller with "--vsys 0 --unit 11 --controller <id>";
    disable with "--vsys 0 --unit 11 --ignore")
```

The individual configuration items are numbered, and depending on their type support different command-line options. The import subcommand can be directed to ignore many such items with a `--vsys X --unit Y --ignore` option, where X is the number of the virtual system and Y the item number, as printed on the screen. X is zero, unless there are several virtual system descriptions in the appliance.

In the above example, Item #1 specifies the name of the target machine in Oracle VM VirtualBox. Items #9 and #10 specify hard disk controllers, respectively. Item #11 describes a hard disk image. In this case, the additional `--controller` option indicates which item the disk image should be connected to, with the default coming from the OVF file.

You can combine several items for the same virtual system behind the same `--vsys` option. For example, to import a machine as described in the OVF, but without the sound card and without the USB controller, and with the disk image connected to the IDE controller instead of the SCSI controller, use the following command:

```
VBoxManage import WindowsXp.ovf
      --vsys 0 --unit 5 --ignore --unit 6 --ignore --unit 11 --controller 10
```

## 8.12 VBoxManage export

This command exports one or more virtual machines from Oracle VM VirtualBox. You can export to either of the following:

- A virtual appliance in OVF format, including copying their virtual disk images to compressed VMDK.
- A cloud service, such as Oracle Cloud Infrastructure. A single VM can be exported in VMDK format.

See chapter 1.15, *Importing and Exporting Virtual Machines*, page 21 for more details on exporting VMs from Oracle VM VirtualBox.

### 8.12.1 Export to OVF

List the machine, or the machines, that you would like to export to the same OVF file and specify the target OVF file after an additional `--output` or `-o` option. Note that the directory of the target OVF file will also receive the exported disk images in the compressed VMDK format, regardless of the original format, and should have enough disk space left for them.

Beside a simple export of a given virtual machine, you can append several product information to the appliance file. Use `--product`, `--producturl`, `--vendor`, `--vendorurl`, `--version` and `--description` to specify this additional information. For legal reasons you may add a license text or the content of a license file by using the `--eula` and `--eulafile` option respectively.

As with OVF import, you use the `--vsys X` option to apply these options to the correct virtual machine.

For virtualization products which are not fully compatible with the OVF standard 1.0 you can enable an OVF 0.9 legacy mode with the `--legacy09` option. Other options are `--ovf09`, `--ovf10`, `--ovf20`.

To specify options controlling the exact content of the appliance file, you can use `--options` to request the creation of a manifest file, which enables detection of corrupted appliances on import, the additional export of DVD images, and the exclusion of MAC addresses. You can specify a list of options, such as `--options manifest,nomacs`. For details, check the help output of `VBoxManage export`.

### 8.12.2 Export to Oracle Cloud Infrastructure

By default, an exported disk image is converted into stream VMDK format. This ensures compatibility with Oracle Cloud Infrastructure.

List the machine that you want to export to Oracle Cloud Infrastructure and specify the target cloud service provider by using the `--output` or `-o` option.

To export a VM to a cloud service such as Oracle Cloud Infrastructure, use the `--cloud` option to specify the VM to export. This option works in the same way as the `--vsys` option for OVF export.

Some of the following options are settings for the VM instance. As a result, you must enter an Oracle Cloud Identifier (OCID) for a resource. Use the Oracle Cloud Infrastructure Console to view OCIDs.

- `--output/-o`: Specifies the short name of the cloud service provider to which you export. For Oracle Cloud Infrastructure, enter `OCI://`.
- `--cloud` *`number-of-virtual-system`*: Specifies a number that identifies the VM that you are exporting. Numbering starts at `0` for the first VM.
- `--vmname` *`name`*: Specifies the name of the exported VM. This name is used as the VM instance name in Oracle Cloud Infrastructure.

- `--cloudprofile` *`cloud-profile-name`*: Specifies the cloud profile that is used to connect to the cloud service provider. The cloud profile contains your Oracle Cloud Infrastructure account details, such as your user OCID and the fingerprint for your public key. See chapter 1.15.4, *Exporting an Appliance to Oracle Cloud Infrastructure*, page 24.

  To use a cloud profile, you must have the required permissions on Oracle Cloud Infrastructure.

  Add xref to information about the required permissions.

- `--cloudshape` *`shape`*: Specifies the shape used for the VM instance. The shape defines the number of CPUs and the amount of memory allocated to the VM instance. The shape must be compatible with the exported image.

- `--clouddomain` *`domain`*: Specifies the availability domain to use for the VM instance. Enter the OCID for the availability domain.

- `--clouddisksize` *`disk-size-in-GB`*: Specifies the disk size used for the exported disk image in gigabytes. The minimum value is 50 GB and the maximum value is 300 GB.

- `--cloudbucket` *`bucket-name`*: Specifies the bucket in which to store the uploaded files. In Oracle Cloud Infrastructure, a bucket is a logical container for storing objects.

- `--cloudocivcn` *`OCI-vcn-ID`*: Specifies the virtual cloud network (VCN) to use for the VM instance. Enter the OCID for the VCN.

- `--cloudocisubnet` *`OCI-subnet-ID`*: Specifies the subnet of the VCN to use for the VM instance. Enter the OCID for the subnet.

- `--cloudkeepobject true | false`: Specifies whether to store the exported disk image in Oracle Object Storage.

- `--cloudlaunchinstance true | false`: Specifies whether to start the VM instance after the export to Oracle Cloud Infrastructure completes.

- `--cloudpublicip true | false`: Specifies whether to enable a public IP address for the VM instance.

The following example shows a typical command line for exporting a VM to Oracle Cloud Infrastructure.

For the next release, describe exactly what this command does in terms of the command line options.

```
# VBoxManage export myVM --output OCI:// --cloud 0 --vmname myVM_Cloud \
--cloudprofile "standard user" --cloudbucket myBucket \
--cloudshape VM.Standard2.1 --clouddomain aaaa:US-ASHBURN-AD-1 --clouddisksize 50  \
--cloudocivcn ocid1.vcn.oc1.iad.aaaa... --cloudocisubnet ocid1.subnet.oc1.iad.aaaa... \
--cloudkeepobject true --cloudlaunchinstance true --cloudpublicip true
```

## 8.13 VBoxManage startvm

This command starts a virtual machine that is currently in the Powered Off or Saved states.

The optional `--type` specifier determines whether the machine will be started in a window or whether the output should go through `VBoxHeadless`, with VRDE enabled or not. See chapter 7.1.2, *VBoxHeadless, the Remote Desktop Server*, page 112. The list of types is subject to change, and it is not guaranteed that all types are accepted by any product variant.

The global or per-VM default value for the VM frontend type will be taken if the type is not explicitly specified. If none of these are set, the GUI variant will be started.

The following values are allowed:

**gui**

Starts a VM showing a GUI window. This is the default.

**headless**

Starts a VM without a window for remote display only.

**separate**

Starts a VM with a detachable UI. Technically, it is a headless VM with user interface in a separate process. This is an experimental feature as it lacks certain functionality, such as 3D acceleration.

**Note:** If you experience problems with starting virtual machines with particular frontends and there is no conclusive error information, consider starting virtual machines directly by running the respective front-end, as this can give additional error information.

## 8.14 VBoxManage controlvm

The `controlvm` subcommand enables you to change the state of a virtual machine that is currently running. The following can be specified:

- `VBoxManage controlvm <vm> pause`: Temporarily puts a virtual machine on hold, without permanently changing its state. The VM window is gray, to indicate that the VM is currently paused. This is equivalent to selecting the **Pause** item in the **Machine** menu of the GUI.
- Use `VBoxManage controlvm <vm> resume`: Undoes a previous `pause` command. This is equivalent to selecting the **Resume** item in the **Machine** menu of the GUI.
- `VBoxManage controlvm <vm> reset`: Has the same effect on a virtual machine as pressing the Reset button on a real computer. A cold reboot of the virtual machine is done, which immediately restarts and reboots the guest operating system. The state of the VM is not saved beforehand, and data may be lost. This is equivalent to selecting the **Reset** item in the **Machine** menu of the GUI.
- `VBoxManage controlvm <vm> poweroff`: Has the same effect on a virtual machine as pulling the power cable on a real computer. The state of the VM is not saved beforehand, and data may be lost. This is equivalent to selecting the **Close** item in the **Machine** menu of the GUI, or clicking the VM window's close button, and then selecting **Power Off the Machine** in the displayed dialog.

  After this, the VM's state will be Powered Off. From that state, it can be started again. See chapter 8.13, *VBoxManage startvm*, page 151.
- `VBoxManage controlvm <vm> savestate`: Saves the current state of the VM to disk and then stops the VM. This is equivalent to selecting the **Close** item in the **Machine** menu of the GUI or clicking the VM window's close button, and then selecting **Save the Machine State** in the displayed dialog.

  After this, the VM's state will be Saved. From this state, it can be started again. See chapter 8.13, *VBoxManage startvm*, page 151.

- `VBoxManage controlvm <vm> acpipowerbutton`: Sends an ACPI shutdown signal to the VM, as if the power button on a real computer had been pressed. So long as the VM is running a fairly modern guest operating system providing ACPI support, this should trigger a proper shutdown mechanism from within the VM.

- `VBoxManage controlvm <vm> keyboardputscancode <hex> [<hex>...]`: Sends commands using keycodes to the VM. Keycodes are documented in the public domain. For example: http://www.win.tue.nl/~aeb/linux/kbd/scancodes-1.html.

- `VBoxManage controlvm "VM name" teleport --hostname <name> --port <port> [--passwordfile <`
  Makes the machine the source of a teleporting operation and initiates a teleport to the given target. See chapter 7.2, *Teleporting*, page 118. If the optional password is specified, it must match the password that was given to the `modifyvm` command for the target machine. See chapter 8.8.6, *Teleporting Settings*, page 146.

The following extra options are available with `controlvm` that do not directly affect the VM's running state:

- `setlinkstate<1-N>`: Connects or disconnects virtual network cables from their network interfaces.

- `nic<1-N> null|nat|bridged|intnet|hostonly|generic|natnetwork[<devicename>]`: Specifies the type of networking that should be made available on the specified VM virtual network card. They available types are: not connected to the host (`null`), use network address translation (`nat`), bridged networking (`bridged`), communicate with other virtual machines using internal networking (`intnet`), host-only networking (`hostonly`), natnetwork networking (`natnetwork`), or access to rarely used submodes (`generic`). These options correspond to the modes which are described in detail in chapter 6.2, *Introduction to Networking Modes*, page 99.

- With the `nictrace` options, you can optionally trace network traffic by dumping it to a file, for debugging purposes.

  `nictrace<1-N> on|off`: Enables network tracing for a particular virtual network card.

  If enabled, you must specify with `--nictracefile<1-N> <filename>` the pathname of the file to which the trace should be logged.

- `nicpromisc<1-N> deny|allow-vms|allow-all`: Specifies how the promiscious mode is handled for the specified VM virtual network card. This setting is only relevant for bridged networking. The default setting of `deny` hides any traffic not intended for this VM. `allow-vms` hides all host traffic from this VM but enables the VM to see traffic to and from other VMs. `allow-all` removes this restriction completely.

- `nicproperty<1-N> <paramname>="paramvalue"`: This option, in combination with `nicgenericdrv` enables you to pass parameters to rarely-used network backends.

  Those parameters are backend engine-specific, and are different between UDP Tunnel and the VDE backend drivers. See chapter 6.8, *UDP Tunnel Networking*, page 106.

- `natpf<1-N> [<name>],tcp|udp,[<hostip>],<hostport>,[<guestip>], <guestport>`: Specifies a NAT port-forwarding rule. See chapter 6.3.1, *Configuring Port Forwarding with NAT*, page 100.

- `natpf<1-N> delete <name>`: Deletes a NAT port-forwarding rule. See chapter 6.3.1, *Configuring Port Forwarding with NAT*, page 100.

- The `guestmemoryballoon<balloon size in MB>`: Changes the size of the guest memory balloon. This is the memory allocated by the Oracle VM VirtualBox Guest Additions from the guest operating system and returned to the hypervisor for reuse by other virtual machines. This must be specified in megabytes. See chapter 4.10.1, *Memory Ballooning*, page 80.

- `usbattach<uuid|address> [--capturefile <filename>]`

  and `usbdetach <uuid|address> [--capturefile <filename>]`: Makes host USB devices visible or invisible to the virtual machine on the fly, without the need for creating filters first. The USB devices can be specified by UUID (unique identifier) or by address on the host system. Use the `--capturefile` option to specify the absolute path of a file for writing activity logging data.

  You can use `VBoxManage list usbhost` to locate this information.

- `audioin on`: Selects whether capturing audio from the host is enabled or disabled.

- `audioout on`: Selects whether audio playback from the guest is enabled or disabled.

- `clipboard disabled|hosttoguest|guesttohost|bidirectional`: Selects how the guest or host operating system's clipboard should be shared with the host or guest. See chapter 3.4, *General Settings*, page 45. This requires that the Guest Additions be installed in the virtual machine.

- `draganddrop disabled|hosttoguest|guesttohost|bidirectional`: Selects the current drag and drop mode being used between the host and the virtual machine. See chapter 4.4, *Drag and Drop*, page 72. This requires that the Guest Additions be installed in the virtual machine.

- `vrde on|off`: Enables and disables the VRDE server, if it is installed.

- `vrdeport default|<ports>`: Changes the port or a range of ports that the VRDE server can bind to. `default` or `0` means port 3389, the standard port for RDP. See the description for the `--vrdeport` option in chapter 8.8.5, *Remote Machine Settings*, page 143.

- `vrdeproperty "TCP/Ports|Address=<value>"`: Sets the port numbers and IP address on the VM to which the VRDE server can bind.

  - For TCP/Ports, <value> should be a port or a range of ports to which the VRDE server can bind. `default` or `0` means port 3389, the standard port for RDP. See the description for the `--vrdeport` option in chapter 8.8.5, *Remote Machine Settings*, page 143.
  - For TCP/Address, <value>: The IP address of the host network interface that the VRDE server will bind to. If specified, the server will accept connections only on the specified host network interface. See the description for the `--vrdeaddress` option in chapter 8.8.5, *Remote Machine Settings*, page 143.

- `vrdeproperty "VideoChannel/Enabled|Quality|DownscaleProtection=<value>"`: Sets the VRDP video redirection properties.

  - For VideoChannel/Enabled, <value> can be set to "1" switching the VRDP video channel on. See chapter 7.1.9, *VRDP Video Redirection*, page 117.
  - For VideoChannel/Quality, <value> should be set between 10 and 100% inclusive, representing a JPEG compression level on the VRDE server video channel. Lower values mean lower quality but higher compression. See chapter 7.1.9, *VRDP Video Redirection*, page 117.

  - For VideoChannel/DownscaleProtection, <value> can be set to "1" to enable the videochannel downscale protection feature. When enabled, if a video's size equals the shadow buffer size, then it is regarded as a full screen video, and is displayed. If its size is between fullscreen and the downscale threshold it is not displayed, as it could be an application window, which would be unreadable when downscaled. When the downscale protection feature is disabled, an attempt is always made to display videos.

- `vrdeproperty "Client/DisableDisplay|DisableInput|DisableAudio|DisableUSB=1"`: Disables one of the VRDE server features: Display, Input, Audio, or USB. To reenable a feature, use "Client/DisableDisplay=" for example. See chapter 7.1.10, *VRDP Customization*, page 118.

- `vrdeproperty "Client/DisableClipboard|DisableUpstreamAudio=1"`. Disables one of the VRDE server features: Clipboard or UpstreamAudio. To reenable a feature, use "Client/DisableClipboard=" for example. See chapter 7.1.10, *VRDP Customization*, page 118.

- `vrdeproperty "Client/DisableRDPDR=1"`: Disables the VRDE server feature: RDP device redirection for smart cards. To reenable this feature, use "Client/DisableRDPR=".

- `vrdeproperty "H3DRedirect/Enabled=1"`: Enables the VRDE server feature: 3D redirection. To disable this feature, use "H3DRedirect/Enabled=".

- `vrdeproperty "Security/Method|ServerCertificate|ServerPrivateKey|CACertificate=<value>"`: Sets the desired security method, path of the server certificate, path of the server private key, and path of CA certificate, used for a connection.

  - `vrdeproperty "Security/Method=<value>"`: Sets the desired security method, which is used for a connection. Valid values are as follows:

    * `Negotiate`: Both Enhanced (TLS) and Standard RDP Security connections are allowed. The security method is negotiated with the client. This is the default setting.
    * `RDP`: Only Standard RDP Security is accepted.
    * `TLS`: Only Enhanced RDP Security is accepted. The client must support TLS.

    See chapter 7.1.6, *RDP Encryption*, page 116.

  - `vrdeproperty "Security/ServerCertificate=<value>"` where <value> is the absolute path of the server certificate. See chapter 7.1.6, *RDP Encryption*, page 116.
  - `vrdeproperty "Security/ServerPrivateKey=<value>"` where <value> is the absolute path of the server private key. See chapter 7.1.6, *RDP Encryption*, page 116.
  - `vrdeproperty "Security/CACertificate=<value>"` where <value> is the absolute path of the CA self signed certificate. See chapter 7.1.6, *RDP Encryption*, page 116.

- `vrdeproperty "Audio/RateCorrectionMode|LogPath=<value>"`: Sets the audio connection mode, or path of the audio logfile.

  - `vrdeproperty "Audio/RateCorrectionMode=<value>"` where <value> is the desired rate correction mode, allowed values are:

    * `VRDP_AUDIO_MODE_VOID`: No mode specified, use to unset any Audio mode already set.
    * `VRDP_AUDIO_MODE_RC`: Rate correction mode.
    * `VRDP_AUDIO_MODE_LPF`: Low pass filter mode.

    - `VRDP_AUDIO_MODE_CS`: Client sync mode to prevent underflow or overflow of the client queue.
  - `vrdeproperty "Audio/LogPath=<value>"` where <value> is the absolute path of the audio log file.

- `vrdevideochannelquality <percent>`: Sets the image quality for video redirection. See chapter 7.1.9, *VRDP Video Redirection*, page 117.
- `setvideomodehint`: Requests that the guest system change to a particular video mode. This requires that the Guest Additions be installed, and will not work for all guest systems.
- `screenshotpng`: Takes a screenshot of the guest display and saves it in PNG format.
- `recording on|off` enables or disables the recording of a VM session into a WebM/VP8 file. When this option value is `on`, recording begins when the VM session starts.
- `recordingscreens all|`*`screen-ID`* `[`*`screen-ID`* `...]` enables you to specify which VM screens to record. The recording for each screen that you specify is saved to its own file. You cannot modify this setting while recording is enabled.
- `recordingfile` *`filename`* specifies the file in which to save the recording. You cannot modify this setting while recording is enabled.
- `recordingvideores` *`widthxheight`* specifies the resolution of the recorded video in pixels. You cannot modify this setting while recording is enabled.
- `recordingvideorate` *`bit-rate`* specifies the bit rate of the video in kilobits per second. Increasing this value improves the appearance of the video at the cost of an increased file size. You cannot modify this setting while recording is enabled.
- `recordingvideofps` *`fps`* specifies the maximum number of video frames per second (FPS) to record. Frames that have a higher frequency are skipped. Increasing this value reduces the number of skipped frames and increases the file size. You cannot modify this setting while recording is enabled.
- `recordingmaxtime` *`seconds`* specifies the maximum amount time to record in seconds. The recording stops after the specified number of seconds elapses. If this value is zero, the recording continues until you stop the recording.
- `recordingmaxsize` *`MB`* specifies the maximum size of the recorded video file in megabytes. The recording stops when the file reaches the specified size. If this value is zero, the recording continues until you stop the recording. You cannot modify this setting while recording is enabled.
- `recordingopts` *`keyword=value`*`[,`*`keyword=value`* `...]` specifies additional recording options in a comma-separated keyword-value format. For example, `foo=bar,a=b`. You cannot modify this setting while recording is enabled.

  Only use this option only if you are an advanced user. For information about keywords, see *Oracle VM VirtualBox Programming Guide and Reference*.
- `setcredentials`: Used for remote logins on Windows guests. See chapter 9.1, *Automated Guest Logins*, page 206.
- `teleport --host <name> --port <port>`: Configures a VM as a target for teleporting. <name> specifies the virtual machine name. <port> specifies the port on the virtual machine which should listen for teleporting requests from other virtual machines. It can be any free TCP/IP port number, such as 6000. See chapter 7.2, *Teleporting*, page 118.

  - `--maxdowntime <msec>`: Specifies the maximum downtime, in milliseconds, for the teleporting target VM. Optional.
  - `--password <password>`: The teleporting request will only succeed if the source machine specifies the same password as the one given with this command. Optional.
  - `--passwordfile <password file>`: The teleporting request will only succeed if the source machine specifies the same password as the one specified in the password file with the path specified with this command. Use `stdin` to read the password from stdin. Optional.

- `plugcpu|unplugcpu <id>`: If CPU hot-plugging is enabled, this setting adds and removes a virtual CPU to the virtual machine. `<id>` specifies the index of the virtual CPU to be added or removed and must be a number from 0 to the maximum number of CPUs configured. CPU 0 can never be removed.

- The `cpuexecutioncap <1-100>`: Controls how much CPU time a virtual CPU can use. A value of 50 implies a single virtual CPU can use up to 50% of a single host CPU.

- `webcam attach <path|alias> [<keyword=value>[;<keyword=value>...]]`: Attaches a webcam to a running VM. Specify the absolute path of the webcam on the host operating system, or use its alias, obtained by using the command: `VBoxManage list webcams`.

  Note that alias '.0' means the default video input device on the host operating system, '.1', '.2', etc. mean first, second, etc. video input device. The device order is host-specific.

  The optional settings parameter is a `;` delimited list of name-value pairs, enabling configuration of the emulated webcam device.

  The following settings are supported:

  MaxFramerate: Specifies the highest rate in frames per second, at which video frames are sent to the guest. Higher frame rates increase CPU load, so this setting can be useful when there is a need to reduce CPU load. The default setting is `no maximum limit`, thus enabling the guest to use all frame rates supported by the host webcam.

  MaxPayloadTransferSize: Specifies the maximum number of bytes the emulated webcam can send to the guest in one buffer. The default setting is 3060 bytes, which is used by some webcams. Higher values can slightly reduce CPU load, if the guest is able to use larger buffers. Note that higher MaxPayloadTransferSize values may be not supported by some guest operating systems.

- `webcam detach <path|alias>`: Detaches a webcam from a running VM. Specify the absolute path of the webcam on the host, or use its alias obtained from the `webcam list` command.

  Please note the following points, relating to specific host operating systems:

  - Windows hosts: When the webcam device is detached from the host, the emulated webcam device is automatically detached from the guest.
  - Mac OS X hosts: OS X version 10.7 or newer is required.

    When the webcam device is detached from the host, the emulated webcam device remains attached to the guest and must be manually detached using the `VBoxManage controlvm webcam detach` command.
  - Linux hosts: When the webcam is detached from the host, the emulated webcam device is automatically detached from the guest only if the webcam is streaming video. If the emulated webcam is inactive, it should be manually detached using the `VBoxManage controlvm webcam detach` command.

- `webcam list`: Lists webcams attached to the running VM. The output is a list of absolute paths or aliases that were used for attaching the webcams to the VM using the `webcam attach` command.

- `addencpassword <id> <password file>|- [--removeonsuspend <yes|no>]`: Supplies an encrypted VM specified by <id> with the encryption password to enable a headless start. Either specify the absolute path of a password file on the host file system: <password file>, or use `-` to instruct `VBoxManage` to prompt the user for the encryption password.

  `--removeonsuspend <yes|no>`: Specifies whether to remove the passsword or keep the password in VM memory when the VM is suspended. If the VM has been suspended and the password has been removed, the user needs to resupply the password before the VM can be resumed. This feature is useful in cases where the user does not want the password to be stored in VM memory, and the VM is suspended by a host suspend event.

  **Note:** On Oracle VM VirtualBox versions 5.0 and later, data stored on hard disk images can be transparently encrypted for the guest. Oracle VM VirtualBox uses the AES algorithm in XTS mode and supports 128 or 256 bit data encryption keys (DEK). The DEK is stored encrypted in the medium properties, and is decrypted during VM startup by supplying the encryption password.

  The `VBoxManage encryptmedium` command is used to create a DEK encrypted medium. See chapter 9.30.2, *Encrypting Disk Images*, page 252. When starting an encrypted VM from the Oracle VM VirtualBox GUI, the user will be prompted for the encryption password.

  For a headless encrypted VM start, use the following command:

```
VBoxManage startvm "vmname" --type headless
```

  Then supply the required encryption password as follows:

```
VBoxManage "vmname" controlvm "vmname" addencpassword ...
```

- `removeencpassword <id>`: Removes encryption password authorization for password <id> for all encrypted media attached to the VM.

- `removeallencpasswords`: Removes encryption password authorization for all passwords for all encrypted media attached to the VM.

- `changeuartmode <1-N>`: Changes the connection mode for a given virtual serial port.

## 8.15 VBoxManage discardstate

This command discards the saved state of a virtual machine which is not currently running. This will cause the VM's operating system to restart next time you start it. This is the equivalent of pulling out the power cable on a physical machine, and should be avoided if possible.

## 8.16 VBoxManage adoptstate

If you have a Saved state file (`.sav`) that is separate from the VM configuration, you can use this command to *adopt* the file. This will change the VM to saved state and when you start it, Oracle VM VirtualBox will attempt to restore it from the saved state file you indicated. This command should only be used in special setups.

## 8.17 VBoxManage snapshot

This command is used to control snapshots from the command line. A snapshot consists of a complete copy of the virtual machine settings, copied at the time when the snapshot was taken, and optionally a virtual machine saved state file if the snapshot was taken while the machine was running. After a snapshot has been taken, Oracle VM VirtualBox creates a differencing hard disk for each normal hard disk associated with the machine so that when a snapshot is restored, the contents of the virtual machine's virtual hard disks can be quickly reset by simply dropping the preexisting differencing files.

```
VBoxManage snapshot         <uuid|vmname>
                            take <name> [--description <desc>] [--live]
                                 [--uniquename Number,Timestamp,Space,Force] |
                            delete <uuid|snapname> |
                            restore <uuid|snapname> |
                            restorecurrent |
                            edit <uuid|snapname>|--current
                                 [--name <name>]
                                 [--description <desc>] |
                            list [--details|--machinereadable]
                            showvminfo <uuid|snapname>
```

The `take` operation takes a snapshot of the current state of the virtual machine. You must supply a name for the snapshot and can optionally supply a description. The new snapshot is inserted into the snapshots tree as a child of the current snapshot and then becomes the new current snapshot. The `--description` parameter enables you to describe the snapshot. If `--live` is specified, the VM will not be stopped during the snapshot creation. This is called live snapshotting.

The `delete` operation deletes a snapshot, specified by name or by UUID. This can take a while to finish since the differencing images associated with the snapshot might need to be merged with their child differencing images.

The `restore` operation will restore the given snapshot, specified by name or by UUID, by resetting the virtual machine's settings and current state to that of the snapshot. The previous current state of the machine will be lost. After this, the given snapshot becomes the new current snapshot so that subsequent snapshots are inserted under the snapshot from which was restored.

The `restorecurrent` operation is a shortcut to restore the current snapshot, which is the snapshot from which the current state is derived. This subcommand is equivalent to using the `restore` subcommand with the name or UUID of the current snapshot, except that it avoids the extra step of determining that name or UUID.

With the `edit` operation, you can change the name or description of an existing snapshot.

The `list` operation shows all snapshots of a virtual machine.

With the `showvminfo` operation, you can view the virtual machine settings that were stored with an existing snapshot.

## 8.18 VBoxManage closemedium

This command removes a hard disk, DVD, or floppy image from a Oracle VM VirtualBox media registry.

```
VBoxManage closemedium      [disk|dvd|floppy] <uuid|filename>
                            [--delete]
```

Optionally, you can request that the image be deleted. You will get appropriate diagnostics that the deletion failed, however the image will become unregistered in any case.

## 8.19 VBoxManage storageattach

This command attaches, modifies, and removes a storage medium connected to a storage controller that was previously added with the `storagectl` command. The syntax is as follows:

```
VBoxManage storageattach    <uuid|vmname>
                            --storagectl <name>
                            [--port <number>]
                            [--device <number>]
                            [--type dvddrive|hdd|fdd]
                            [--medium none|emptydrive|additions|
                                      <uuid>|<filename>|host:<drive>|iscsi]
                            [--mtype normal|writethrough|immutable|shareable
                                     readonly|multiattach]
                            [--comment <text>]
                            [--setuuid <uuid>]
                            [--setparentuuid <uuid>]
                            [--passthrough on|off]
                            [--tempeject on|off]
                            [--nonrotational on|off]
                            [--discard on|off]
                            [--hotpluggable on|off]
                            [--bandwidthgroup name|none]
                            [--forceunmount]
                            [--server <name>|<ip>]
                            [--target <target>]
                            [--tport <port>]
                            [--lun <lun>]
                            [--encodedlun <lun>]
                            [--username <username>]
                            [--password <password>]
                            [--passwordfile <file>]
                            [--initiator <initiator>]
                            [--intnet]
```

A number of parameters are commonly required. Some parameters are required only for iSCSI targets.

The common parameters are as follows:

**`uuid|vmname`**

The VM UUID or VM Name. Mandatory.

**`--storagectl`**

Name of the storage controller. Mandatory. The list of the storage controllers currently attached to a VM can be obtained with `VBoxManage showvminfo`. See chapter 8.5, *VBoxManage showvminfo*, page 133.

**`--port`**

The number of the storage controller's port which is to be modified. Mandatory, unless the storage controller has only a single port.

**--device**

The number of the port's device which is to be modified. Mandatory, unless the storage controller has only a single device per port.

**--type**

Define the type of the drive to which the medium is being attached, detached, or modified. This argument can only be omitted if the type of medium can be determined from either the medium given with the `--medium` argument or from a previous medium attachment.

**--medium**

Specifies what is to be attached. The following values are supported:

- `none`: Any existing device should be removed from the given slot.
- `emptydrive`: For a virtual DVD or floppy drive only, this makes the device slot behave like a removeable drive into which no media has been inserted.
- `additions`: For a virtual DVD drive only, this attaches the *VirtualBox Guest Additions* image to the given device slot.
- If a UUID is specified, it must be the UUID of a storage medium that is already known to Oracle VM VirtualBox. For example, because it has been attached to another virtual machine. See chapter 8.4, *VBoxManage list*, page 132 for details of how to list known media. This medium is then attached to the given device slot.
- If a filename is specified, it must be the full path of an existing disk image in ISO, RAW, VDI, VMDK, or other format. The disk image is then attached to the given device slot.
- `host:<drive>`: For a virtual DVD or floppy drive only, this connects the given device slot to the specified DVD or floppy drive on the host computer.
- `iscsi`: For virtual hard disks only, this is used for specifying an iSCSI target. In this case, additional parameters must be given. These are described below.

Some of the above changes, in particular for removeable media such as floppies and CDs/DVDs, can be effected while a VM is running. Others, such as device changes or changes in hard disk device slots, require the VM to be powered off.

**--mtype**

Defines how this medium behaves with respect to snapshots and write operations. See chapter 5.4, *Special Image Write Modes*, page 89.

**--comment**

An optional description that you want to have stored with this medium. For example, for an iSCSI target, "Big storage server downstairs". This is purely descriptive and not needed for the medium to function correctly.

**--setuuid, --setparentuuid**

Modifies the UUID or parent UUID of a medium before attaching it to a VM. This is an expert option. Inappropriate use can make the medium unusable or lead to broken VM configurations if any other VM is referring to the same media already. The most frequently used variant is `--setuuid ""`, which assigns a new random UUID to an image. This option is useful for resolving duplicate UUID errors if you duplicated an image using a file copy utility.

**--passthrough**

For a virtual DVD drive only, you can enable DVD writing support. This feature is currently experimental, see chapter 5.9, *CD/DVD Support*, page 94.

**--tempeject**

For a virtual DVD drive only, you can configure the behavior for guest-triggered medium eject. If this is set to on, the eject has only a temporary effect. If the VM is powered off and restarted the originally configured medium will be still in the drive.

**--nonrotational**

Enables you to enable the non-rotational flag for virtual hard disks. Some guests, such as Windows 7 or later, treat such disks like SSDs and do not perform disk fragmentation on such media.

**--discard**

Enables the auto-discard feature for a virtual hard disks. This specifies that a VDI image will be shrunk in response to the trim command from the guest OS. The following requirements must be met:

- The disk format must be VDI.
- The size of the cleared area must be at least 1 MB.
- Oracle VM VirtualBox will only trim whole 1 MB blocks. The VDIs themselves are organized into 1 MB blocks, so this will only work if the space being trimmed is at least a 1 MB contiguous block at a 1 MB boundary. On Windows, occasional defragmentation with `defrag.exe /D`, or on Linux running `btrfs filesystem defrag` as a background cron job may be beneficial.

**Note:** The Guest OS must be configured to issue the `trim` command, and typically this means that the guest OS is made to see the disk as an SSD. Ext4 supports the -o discard mount flag. Mac OS X probably requires additional settings. Windows should automatically detect and support SSDs, at least in versions 7, 8, and 10. The Linux exFAT driver from Samsung supports the `trim` command.

It is unclear whether Microsoft's implementation of exFAT supports this feature, even though that file system was originally designed for flash.

Alternatively, there are other methods to issue trim. For example, the Linux `fstrim` command, part of the util-linux package. Earlier solutions required a user to zero out unused areas, using zerofree or similar, and to compact the disk. This is only possible when the VM is offline.

**--bandwidthgroup**

Sets the bandwidth group to use for the given device. See chapter 5.8, *Limiting Bandwidth for Disk Images*, page 93.

**--forceunmount**

For a virtual DVD or floppy drive only, this forcibly unmounts the DVD/CD/Floppy or mounts a new DVD/CD/Floppy even if the previous one is locked down by the guest for reading. See chapter 5.9, *CD/DVD Support*, page 94.

When `iscsi` is used with the `--medium` parameter for iSCSI support, additional parameters must or can be used. See also chapter 5.10, *iSCSI Servers*, page 95.

**`--server`**

The host name or IP address of the iSCSI target. Required.

**`--target`**

Target name string. This is determined by the iSCSI target and used to identify the storage resource. Required.

**`--tport`**

TCP/IP port number of the iSCSI service on the target. Optional.

**`--lun`**

Logical Unit Number of the target resource. Optional. Often, this value is zero.

**`--encodedlun`**

Hex-encoded Logical Unit Number of the target resource. Optional. Often, this value is zero.

**`--username, --password, --passwordfile`**

Username and password, called the initiator secret, for target authentication, if required. Optional.

**Note:** Username and password are stored without encryption, in clear text, in the XML machine configuration file if no settings password is provided. When a settings password is specified for the first time, the password is stored in encrypted form. As an alternative to providing the password on the command line, a reference to a file containing the text can be provided using the `passwordfile` option.

**`--initiator`**

iSCSI Initiator. Optional.

Microsoft iSCSI Initiator is a system, such as a server that attaches to an IP network and initiates requests and receives responses from an iSCSI target. The SAN components in Microsoft iSCSI Initiator are largely analogous to Fibre Channel SAN components, and they include the following:

- To transport blocks of iSCSI commands over the IP network, an iSCSI driver must be installed on the iSCSI host. An iSCSI driver is included with Microsoft iSCSI Initiator.

- A gigabit Ethernet adapter that transmits 1000 megabits per second (Mbps) is recommended for the connection to an iSCSI target. Like standard 10/100 adapters, most gigabit adapters use a preexisting Category 5 or Category 6E cable. Each port on the adapter is identified by a unique IP address.
- An iSCSI target is any device that receives iSCSI commands. The device can be an end node, such as a storage device, or it can be an intermediate device, such as a network bridge between IP and Fibre Channel devices. Each port on the storage array controller or network bridge is identified by one or more IP addresses

**--intnet**

If specified, connect to the iSCSI target using Internal Networking. This needs further configuration, see chapter 9.8.3, *Access iSCSI Targets Using Internal Networking*, page 220.

## 8.20 VBoxManage storagectl

This command attaches, modifies, and removes a storage controller. After this, virtual media can be attached to the controller with the `storageattach` command.

The syntax for this command is as follows:

```
VBoxManage storagectl       <uuid|vmname>
                            --name <name>
                            [--add ide|sata|scsi|floppy|sas|usb|pcie]
                            [--controller LSILogic|LSILogicSAS|BusLogic|
                                          IntelAhci|PIIX3|PIIX4|ICH6|I82078|
                                          USB|NVMe]
                            [--portcount <1-30>]
                            [--hostiocache on|off]
                            [--bootable on|off]
                            [--rename <name>]
                            [--remove]
```

The parameters are as follows:

**uuid|vmname**

The VM UUID or VM Name. Mandatory.

**--name**

Specifies the name of the storage controller. Mandatory.

**--add**

Specifies the type of the system bus to which the storage controller must be connected.

**--controller**

Enables a choice of chipset type being emulated for the given storage controller.

**--portcount**

This specifies the number of ports the storage controller should support.

**--hostiocache**

Configures the use of the host I/O cache for all disk images attached to this storage controller. See chapter 5.7, *Host Input/Output Caching*, page 93.

**--bootable**

Specifies whether this controller is bootable.

**--rename**

Specifies a new name for the storage controller.

**--remove**

Removes the storage controller from the VM configuration.

## 8.21 VBoxManage bandwidthctl

This command creates, deletes, modifies, and shows bandwidth groups of the given virtual machine.

```
VBoxManage bandwidthctl    <uuid|vmname>
                           add <name> --type disk|network --limit <MBps>[k|m|g|K|M|G] |
                           set <name> --limit <MBps>[k|m|g|K|M|G] |
                           remove <name> |
                           list [--machinereadable]
```

The following subcommands are available:

- `add`: Creates a new bandwidth group of a given type.
- `set`: Modifies the limit for an existing bandwidth group.
- `remove`: Deletes a bandwidth group.
- `list`: Shows all bandwidth groups defined for the given VM. Use the `--machinereadable` option to produce the same output, but in machine readable format. This is of the form: name="value" on a line by line basis.

The parameters are as follows:

**uuid|vmname**

The VM UUID or VM Name. Mandatory.

**--name**

Name of the bandwidth group. Mandatory.

**--type**

Type of the bandwidth group. Mandatory. Two types are supported: `disk` and `network`. See chapter 5.8, *Limiting Bandwidth for Disk Images*, page 93 or chapter 6.10, *Limiting Bandwidth for Network Input/Output*, page 108 for the description of a particular type.

**--limit**

Specifies the limit for the given bandwidth group. This can be changed while the VM is running. The default unit is megabytes per second. The unit can be changed by specifying one of the following suffixes: `k` for kilobits per second, `m` for megabits per second, `g` for gigabits per second, `K` for kilobytes per second, `M` for megabytes per second, `G` for gigabytes per second.

**Note:** The network bandwidth limits apply only to the traffic being sent by virtual machines. The traffic being received by VMs is unlimited.

**Note:** To remove a bandwidth group it must not be referenced by any disks or adapters in the running VM.

## 8.22 VBoxManage showmediuminfo

This command shows information about a medium, notably its size, its size on disk, its type, and the virtual machines which use it.

**Note:** For compatibility with earlier versions of Oracle VM VirtualBox, the `showvdiinfo` command is also supported and mapped internally to the `showmediuminfo` command.

```
VBoxManage showmediuminfo     [disk|dvd|floppy] <uuid|filename>
```

The medium must be specified either by its UUID, if the medium is registered, or by its filename. Registered images can be listed using `VBoxManage list hdds`, `VBoxManage list dvds`, or `VBoxManage list floppies`, as appropriate. See chapter 8.4, *VBoxManage list*, page 132.

## 8.23 VBoxManage createmedium

This command creates a new medium. The syntax is as follows:

```
VBoxManage createmedium     [disk|dvd|floppy]    --filename <filename>
                            [--size <megabytes>|--sizebyte <bytes>]
                            [--diffparent <uuid>|<filename>
                            [--format VDI|VMDK|VHD] (default: VDI)
                            [--variant Standard,Fixed,Split2G,Stream,ESX]
```

The parameters are as follows:

**--filename <filename>**

Specifies a file name <filename> as an absolute path on the host file system. Mandatory.

**--size <megabytes>**

Specifies the image capacity, in 1 MB units. Optional.

**--diffparent <uuid>|<filename>**

Specifies the differencing image parent, either as a UUID or by the absolute pathname of the file on the host file system. Useful for sharing a base box disk image among several VMs.

**--format VDI|VMDK|VHD**

Specifies the file format for the output file. Available options are VDI, VMDK, VHD. The default format is VDI. Optional.

**--variant**

Specifies any required file format variants for the output file. This is a comma-separated list of variant flags. Options are Standard,Fixed,Split2G,Stream,ESX. Not all combinations are supported, and specifying mutually incompatible flags results in an error message. Optional.

**Note:** For compatibility with earlier versions of Oracle VM VirtualBox, the `createvdi` and `createhd` commands are also supported and mapped internally to the `createmedium` command.

## 8.24 VBoxManage modifymedium

With the `modifymedium` command, you can change the characteristics of a disk image after it has been created.

```
VBoxManage modifymedium  [disk|dvd|floppy]    <uuid|filename>
                         [--type normal|writethrough|immutable|shareable|
                                 readonly|multiattach]
                         [--autoreset on|off]
                         [--property <name=[value]>]
                         [--compact]
                         [--resize <megabytes>|--resizebyte <bytes>]
                         [--move <path>]
                         [--setlocation <path>]
```

**Note:** For compatibility with earlier versions of Oracle VM VirtualBox, the `modifyvdi` and `modifyhd` commands are also supported and mapped internally to the `modifymedium` command.

The disk image to modify must be specified either by its UUID, if the medium is registered, or by its filename. Registered images can be listed using `VBoxManage list hdds`, see chapter 8.4, *VBoxManage list*, page 132. A filename must be specified as a valid path, either as an absolute path or as a relative path starting from the current directory.

The following options are available:

- With the `--type` argument, you can change the type of an existing image between the normal, immutable, write-through and other modes. See chapter 5.4, *Special Image Write Modes*, page 89.

- For immutable hard disks only, the `--autoreset on|off` option determines whether the disk is automatically reset on every VM startup. See chapter 5.4, *Special Image Write Modes*, page 89. By default, autoreset is on.

- The `--compact` option can be used to compact disk images. Compacting removes blocks that only contains zeroes. Using this option will shrink a dynamically allocated image. It will reduce the *physical* size of the image without affecting the logical size of the virtual disk. Compaction works both for base images and for differencing images created as part of a snapshot.

  For this operation to be effective, it is required that free space in the guest system first be zeroed out using a suitable software tool. For Windows guests, you can use the `sdelete` tool provided by Microsoft. Run `sdelete -z` in the guest to zero the free disk space, before compressing the virtual disk image. For Linux, use the `zerofree` utility which supports ext2/ext3 filesystems. For Mac OS X guests, use the `diskutil secureErase freespace 0 /` command from an elevated Terminal.

  Please note that compacting is currently only available for VDI images. A similar effect can be achieved by zeroing out free blocks and then cloning the disk to any other dynamically allocated format. You can use this workaround until compacting is also supported for disk formats other than VDI.

- The `--resize x` option, where x is the desired new total space in megabytes enables you to change the capacity of an existing image. This adjusts the *logical* size of a virtual disk without affecting the physical size much.

  This option currently works only for VDI and VHD formats, and only for the dynamically allocated variants. It can only be used to expand, but not shrink, the capacity. For example, if you originally created a 10 GB disk which is now full, you can use the `--resize 15360` command to change the capacity to 15 GB (15,360 MB) without having to create a new image and copy all data from within a virtual machine. Note however that this only changes the drive capacity. You will typically next need to use a partition management tool inside the guest to adjust the main partition to fill the drive.

  The `--resizebyte x` option does almost the same thing, except that x is expressed in bytes instead of megabytes.

- The `--move <path>` option can be used to relocate a medium to a different location <path> on the host file system. The path can be either relative to the current directory or absolute.

- The `--setlocation <path>` option can be used to set the new location <path> of the medium on the host file system if the medium has been moved for any reasons. The path can be either relative to the current directory or absolute.

> **Note:** The new location is used as is, without any sanity checks. The user is responsible for setting the correct path.

## 8.25 VBoxManage clonemedium

This command duplicates a virtual disk, DVD, or floppy medium to a new medium, usually an image file, with a new unique identifier (UUID). The new image can be transferred to another host system or reimported into Oracle VM VirtualBox using the Virtual Media Manager. See chapter 5.3, *The Virtual Media Manager*, page 87 and chapter 5.6, *Cloning Disk Images*, page 92. The syntax is as follows:

```
VBoxManage clonemedium      [disk|dvd|floppy] <uuid|inputfile> <uuid|outputfile>

                            [--format VDI|VMDK|VHD|RAW|<other>]
                            [--variant Standard,Fixed,Split2G,Stream,ESX]
                            [--existing]
```

The medium to clone as well as the target image must be described either by its UUIDs, if the mediums are registered, or by its filename. Registered images can be listed by `VBoxManage list hdds`. See chapter 8.4, *VBoxManage list*, page 132. A filename must be specified as valid path, either as an absolute path or as a relative path starting from the current directory.

The following options are available:

**`--format`**

Set a file format for the output file different from the file format of the input file.

**`--variant`**

Set a file format variant for the output file. This is a comma-separated list of variant flags. Not all combinations are supported, and specifying inconsistent flags will result in an error message.

**`--existing`**

Perform the clone operation to an already existing destination medium. Only the portion of the source medium which fits into the destination medium is copied. This means if the destination medium is smaller than the source only a part of it is copied, and if the destination medium is larger than the source the remaining part of the destination medium is unchanged.

**Note:** For compatibility with earlier versions of Oracle VM VirtualBox, the `clonevdi` and `clonehd` commands are still supported and mapped internally to the `clonehd disk` command.

## 8.26 VBoxManage mediumproperty

This command sets, gets, or deletes a medium property. The syntax is as follows:

```
VBoxManage mediumproperty [disk|dvd|floppy] set <uuid|filename>
                                            <property> <value>
```

- Use `<disk|dvd|floppy>` to optionally specify the type of medium: disk (hard drive), dvd, or floppy.
- Use `<uuid|filename>` to supply either the UUID or absolute path of the medium or image.
- Use `<property>` to supply the name of the property.
- Use `<value>` to supply the property value.

```
VBoxManage mediumproperty [disk|dvd|floppy] get <uuid|filename>
                                            <property>
```

- Use <disk|dvd|floppy> to optionally specify the type of medium: disk (hard drive), dvd, or floppy.
- Use <uuid|filename> to supply either the UUID or absolute path of the medium or image.
- Use <property> to supply the name of the property.

```
VBoxManage mediumproperty [disk|dvd|floppy] delete <uuid|filename>
                                                   <property>
```

- Use <disk|dvd|floppy> to optionally specify the type of medium: disk (hard drive), dvd, or floppy.
- Use <uuid|filename> to supply either the UUID or absolute path of the medium or image.
- Use <property> to supply the name of the property.

## 8.27 VBoxManage encryptmedium

This command is used to create a DEK encrypted medium or image. See chapter 9.30.2, *Encrypting Disk Images*, page 252.

The syntax is as follows:

```
VBoxManage encryptmedium <uuid|filename>
                         [--newpassword <file|->]
                         [--oldpassword <file|->]
                         [--cipher <cipher id>]
                         [--newpasswordid <password id>]
```

- Use <uuid|filename> to supply the UUID or absolute path of the medium or image to be encrypted.
- Use --newpassword <file|-> to supply a new encryption password. Either specify the absolute pathname of a password file on the host operating system, or - to prompt you for the password on the command line. Always use the --newpasswordid option with this option.
- Use --oldpassword <file|-> to supply any old encryption password. Either specify the absolute pathname of a password file on the host operating system, or - to prompt you for the old password on the command line.

  Use this option to gain access to an encrypted medium or image to either change its password using --newpassword or change its encryption using --cipher.
- Use --cipher <cipher> to specify the cipher to use for encryption. This can be either AES-XTS128-PLAIN64 or AES-XTS256-PLAIN64.

  Use this option to change any existing encryption on the medium or image, or to set up new encryption on it for the first time.
- Use --newpasswordid <password id> to supply the new password identifier. This can be chosen by the user, and is used for correct identification when supplying multiple passwords during VM startup.

  If the user uses the same password when encrypting multiple images and also the same password identifier, the user needs to supply the password only once during VM startup.

## 8.28 VBoxManage checkmediumpwd

This command is used to check the current encryption password on a DEK encrypted medium or image. See chapter 9.30.2, *Encrypting Disk Images*, page 252.

The syntax is as follows:

```
VBoxManage checkmediumpwd <uuid|filename>
                                    <pwd file|->
```

- Use `<uuid|filename>` to supply the UUID or absolute path of the medium or image to be checked.
- Use `<pwd file|->` to supply the password identifier to be checked. Either specify the absolute pathname of a password file on the host operating system, or `-` to prompt you for the password on the command line.

## 8.29 VBoxManage convertfromraw

This command converts a raw disk image to a Oracle VM VirtualBox Disk Image (VDI) file. The syntax is as follows:

```
VBoxManage convertfromraw   <filename> <outputfile>
                            [--format VDI|VMDK|VHD]
                            [--variant Standard,Fixed,Split2G,Stream,ESX]
                            [--uuid <uuid>]
VBoxManage convertfromraw   stdin <outputfile> <bytes>
                            [--format VDI|VMDK|VHD]
                            [--variant Standard,Fixed,Split2G,Stream,ESX]
                            [--uuid <uuid>]
```

The parameters are as follows:

**`--bytes`**

The size of the image file, in bytes, provided through stdin.

**`--format`**

Select the disk image format to create. The default format is VDI. Other options are VMDK and VHD.

**`--variant`**

Choose a file format variant for the output file. This is a comma-separated list of variant flags. Not all combinations are supported, and specifying inconsistent flags will result in an error message.

**`--uuid`**

Specify the UUID of the output file.

The `stdin` form of the command forces `VBoxManage` to read the content of the disk image from standard input. This useful when using the command in a pipe.

> **Note:** For compatibility with earlier versions of Oracle VM VirtualBox, the `convertdd` command is also supported and mapped internally to the `convertfromraw` command.

## 8.30 VBoxManage getextradata/setextradata

These commands enable you to attach and retrieve string data for a virtual machine or for a Oracle VM VirtualBox configuration, by specifying `global` instead of a virtual machine name. You must specify a keyword as a text string to associate the data with, which you can later use to retrieve it. For example:

```
VBoxManage setextradata Fedora5 installdate 2006.01.01
VBoxManage setextradata SUSE10 installdate 2006.02.02
```

This example would associate the string "2006.01.01" with the keyword installdate for the virtual machine Fedora5, and "2006.02.02" on the machine SUSE10. You could then retrieve the information as follows:

```
VBoxManage getextradata Fedora5 installdate
```

This would return the following:

```
VirtualBox Command Line Management Interface Version <version-number>
(C) 2005-2018 Oracle Corporation
All rights reserved.

Value: 2006.01.01
```

You could retrieve the information for all keywords as follows:

```
VBoxManage getextradata Fedora5 enumerate
```

To remove a keyword, the `setextradata` command must be run without specifying data, only the keyword. For example:

```
VBoxManage setextradata Fedora5 installdate
```

## 8.31 VBoxManage setproperty

This command is used to change global settings which affect the entire Oracle VM VirtualBox installation. Some of these correspond to the settings in the **Global Settings** dialog in the graphical user interface. The following properties are available:

**`machinefolder`**

Specifies the default folder in which virtual machine definitions are kept. See chapter 10.1, *Where Oracle VM VirtualBox Stores its Files*, page 265.

**`hwvirtexclusive`**

Specifies whether Oracle VM VirtualBox will make exclusive use of the hardware virtualization extensions (Intel VT-x or AMD-V) of the host system's processor. See chapter 10.3, *Hardware vs. Software Virtualization*, page 270. If you wish to share these extensions with other hypervisors running at the same time, you must disable this setting. Doing so has negative performance implications.

**`vrdeauthlibrary`**

Specifies which library to use when external authentication has been selected for a particular virtual machine. See chapter 7.1.5, *RDP Authentication*, page 115.

**websrvauthlibrary**

Specifies which library the web service uses to authenticate users. For details about the Oracle VM VirtualBox web service, see the Oracle VM VirtualBox SDK reference, chapter 11, *Oracle VM VirtualBox Programming Interfaces*, page 276.

**vrdeextpack**

Specifies which library implements the VirtualBox Remote Desktop Extension.

**loghistorycount**

Selects how many rotated VM logs are retained.

**autostartdbpath**

Selects the path to the autostart database. See chapter 9.23, *Starting Virtual Machines During System Boot*, page 248.

**defaultfrontend**

Selects the global default VM frontend setting. See chapter 8.13, *VBoxManage startvm*, page 151.

**logginglevel**

Configures the VBoxSVC release logging details. See `http://www.virtualbox.org/wiki/VBoxLogging`.

**proxymode**

Configures the mode for an HTTP proxy server.

**proxyurl**

Configures the URL for an HTTP proxy server. Used when a manual proxy is configured using the `manual` setting of the `proxymode` property.

## 8.32 VBoxManage usbfilter add/modify/remove

```
VBoxManage usbfilter        add <index,0-N>
                           --target <uuid|vmname>global
                           --name <string>
                           --action ignore|hold (global filters only)
                          [--active yes|no (yes)]
                          [--vendorid <XXXX> (null)]
                          [--productid <XXXX> (null)]
                          [--revision <IIFF> (null)]
                          [--manufacturer <string> (null)]
                          [--product <string> (null)]
                          [--remote yes|no (null, VM filters only)]
                          [--serialnumber <string> (null)]
                          [--maskedinterfaces <XXXXXXXX>]
```

```
VBoxManage usbfilter        modify <index,0-N>
                           --target <uuid|vmname>global
                          [--name <string>]
                          [--action ignore|hold (global filters only)]
                          [--active yes|no]
                          [--vendorid <XXXX>]
                          [--productid <XXXX>]
                          [--revision <IIFF>]
                          [--manufacturer <string>]
                          [--product <string>]
                          [--remote yes|no (null, VM filters only)]
                          [--serialnumber <string>]
                          [--maskedinterfaces <XXXXXXXX>]


VBoxManage usbfilter        remove <index,0-N>
                           --target <uuid|vmname>global
```

The `usbfilter` commands are used for working with USB filters in virtual machines, or global filters which affect the whole Oracle VM VirtualBox setup. Global filters are applied before machine-specific filters, and may be used to prevent devices from being captured by any virtual machine. Global filters are always applied in a particular order, and only the first filter which fits a device is applied. For example, if the first global filter says to hold, or make available, a particular Kingston memory stick device and the second filter says to ignore all Kingston devices. That particular Kingston memory stick will be available to any machine with the appropriate filter, but no other Kingston device will.

When creating a USB filter using `usbfilter add`, you must supply three or four mandatory parameters. The index specifies the position in the list at which the filter should be placed. If there is already a filter at that position, then it and the following ones will be shifted back one place. Otherwise, the new filter will be added onto the end of the list. The `target` parameter selects the virtual machine that the filter should be attached to or use `global` to apply it to all virtual machines. `name` is a name for the new filter. For global filters, `action` says whether to allow VMs access to devices that fit the filter description (hold) or not to give them access (ignore). In addition, you should specify parameters to filter by. You can find the parameters for devices attached to your system using `VBoxManage list usbhost`. Finally, you can specify whether the filter should be active. For local filters, whether they are for local devices, remote devices over an RDP connection, or either.

When you modify a USB filter using `usbfilter modify`, you must specify the filter by index and by target, which is either a virtual machine or `global`. See the output of `VBoxManage list usbfilters` to find global filter indexes and `VBoxManage showvminfo` to find indexes for individual machines. The properties which can be changed are the same as for `usbfilter add`. To remove a filter, use `usbfilter remove` and specify the index and the target.

The following is a list of the additional `usbfilter add` and `usbfilter modify` options, with details of how to use them.

- `--action ignore|hold`: Specifies whether devices that fit the filter description are allowed access by machines (hold), or have access denied (ignore). Applies to global filters only.
- `--active yes|no`: Specifies whether the USB Filter is active or temporarily disabled. For `usbfilter create` the default is active.
- `--vendorid <XXXX>|""`: Specifies a vendor ID filter. The string representation for an exact match has the form XXXX, where X is the hexadecimal digit, including leading zeroes.
- `--productid <XXXX>|""`: Specifies a product ID filter. The string representation for an exact match has the form XXXX, where X is the hexadecimal digit, including leading zeroes.

- `--revision <IIFF>|""`: Specifies a revision ID filter. The string representation for an exact match has the form IIFF, where I is the decimal digit of the integer part of the revision, and F is the decimal digit of its fractional part, including leading and trailing zeros. Note that for interval filters, it is best to use the hexadecimal form, because the revision is stored as a 16-bit packed BCD value. Therefore, the expression int:0x0100-0x0199 will match any revision from 1.0 to 1.99 inclusive.
- `--manufacturer <string>|""`: Specifies a manufacturer ID filter, as a string.
- `--product <string>|""`: Specifies a product ID filter, as a string.
- `--remote yes|no""`: Specifies a remote filter, indicating whether the device is physically connected to a remote VRDE client or to a local host machine. Applies to VM filters only.
- `--serialnumber <string>|""`: Specifies a serial number filter, as a string.
- `--maskedinterfaces <XXXXXXXX>`: Specifies a masked interface filter, for hiding one or more USB interfaces from the guest. The value is a bit mask where the set bits correspond to the USB interfaces that should be hidden, or masked off. This feature only works on Linux hosts.

## 8.33 VBoxManage sharedfolder add/remove

```
VBoxManage sharedfolder     add <uuid|vmname>
                                --name <name> --hostpath <hostpath>
                                [--transient] [--readonly] [--automount]
```

This command enables you to share folders on the host computer with guest operating systems. For this, the guest systems must have a version of the Oracle VM VirtualBox Guest Additions installed which supports this functionality.

Parameters are as follows:

- `<uuid|vmname>`: Specifies the UUID or name of the VM whose guest operating system will be sharing folders with the host computer. Mandatory.
- `--name <name>`: Specifies the name of the share. Each share has a unique name within the namespace of the host operating system. Mandatory.
- `-hostpath <hostpath>`: Specifies the absolute path on the host operating system of the directory to be shared with the guest operating system. Mandatory.
- `-transient`: Specifies that the share is transient, meaning that it can be added and removed at runtime and does not persist after the VM has stopped. Optional.
- `-readonly`: Specifies that the share has only read-only access to files at the host path.

  By default, shared folders have read/write access to the files on the host path. On Linux distributions, shared folders are mounted with 770 file permissions with root user and vboxsf as the group. Using this option the file permissions change to 700. Optional.
- `-automount`: Specifies that the share will be automatically mounted. On Linux distributions, this will be to either `/media/USER/sf_<name>` or `/media/sf_<name>`, where <name> is the share named. The actual location depends on the guest OS. Optional.

```
VBoxManage sharedfolder     remove <uuid|vmname>
                            --name <name> [--transient]
```

This command enables you to delete shared folders on the host computer shares with the guest operating systems. For this, the guest systems must have a version of the Oracle VM VirtualBox Guest Additions installed which supports this functionality.

Parameters are as follows:

- `<uuid|vmname>`: Specifies the UUID or name of the VM whose guest operating system is sharing folders with the host computer. Mandatory.
- `--name <name>`: Specifies the name of the share to be removed. Each share has a unique name within the namespace of the host operating system. Mandatory.
- `-transient`: Specifies that the share is transient, meaning that it can be added and removed at runtime and does not persist after the VM has stopped. Optional.

Shared folders are described in chapter 4.3, *Shared Folders*, page 69.

## 8.34 VBoxManage guestproperty

The `guestproperty` commands enable you to get or set properties of a running virtual machine. See chapter 4.7, *Guest Properties*, page 76. Guest properties are arbitrary keyword-value string pairs which can be written to and read from by either the guest or the host, so they can be used as a low-volume communication channel for strings, provided that a guest is running and has the Guest Additions installed. In addition, a number of values whose keywords begin with `/VirtualBox/`are automatically set and maintained by the Guest Additions.

The following subcommands are available, where `<vm>` can either be a VM name or a VM UUID, as with the other `VBoxManage` commands:

- `enumerate <vm> [--patterns <pattern>]`: Lists all the guest properties that are available for the given VM, including the value. This list will be very limited if the guest's service process cannot be contacted, for example because the VM is not running or the Guest Additions are not installed.

  If `--patterns <pattern>` is specified, it acts as a filter to only list properties that match the given pattern. The pattern can contain the following wildcard characters:

  - `*` (asterisk): Represents any number of characters. For example, "`/VirtualBox*`" would match all properties beginning with "/VirtualBox".
  - `?` (question mark): Represents a single arbitrary character. For example, "`fo?`" would match both "foo" and "for".
  - `|` (pipe symbol): Can be used to specify multiple alternative patterns. For example, "`s*|t*`" would match anything starting with either "s" or "t".

- `get <vm> <property>`: Retrieves the value of a single property only. If the property cannot be found, for example because the guest is not running, the following message is shown:

  ```
  No value set!
  ```

- `set <vm> <property> [<value> [--flags <flags>]]`: Enables you to set a guest property by specifying the keyword and value. If `<value>` is omitted, the property is deleted. With `--flags`, you can specify additional behavior. You can combine several flags by separating them with commas.

  - `TRANSIENT`: The value will not be stored with the VM data when the VM exits.
  - `TRANSRESET`: The value will be deleted as soon as the VM restarts or exits.

  - RDONLYGUEST: The value can only be changed by the host, but the guest can only read it.
  - RDONLYHOST: The value can only be changed by the guest, but the host can only read it.
  - READONLY: The value cannot be changed at all.

- wait <vm> <pattern> --timeout <timeout>: Waits for a particular value described by the pattern string to change or to be deleted or created. The pattern rules are the same as for the enumerate subcommand.
- delete <vm> <property>: Deletes a guest property which has been set previously.

## 8.35 VBoxManage guestcontrol

The guestcontrol commands enable control of the guest from the host. See chapter 4.9, *Guest Control of Applications*, page 79 for an introduction.

The guestcontrol command has two sets of subcommands. The first set requires guest credentials to be specified, the second does not.

The first set of subcommands is of the following form:

```
VBoxManage guestcontrol <uuid|vmname> <sub-command>
            [--username <name> ]
            [--passwordfile <file> | --password <password>]
            [--domain <domain> ]
            [-v|--verbose] [-q|quiet] ...
```

The common options are as follows:

```
          [--username <name> ]
          [--passwordfile <file> | --password <password>]
          [--domain <domain> ]
          [-v|--verbose] [-q|quiet]
```

The common options for the first set of subcommands are explained in the following list.

**<uuid|vmname>**

Specifies the VM UUID or VM name. Mandatory.

**--username <name>**

Specifies the user name on guest OS under which the process should run. This user name must already exist on the guest OS. If unspecified, the host user name is used. Optional

**--passwordfile <file>|--password**

Specifies the absolute path on guest file system of password file containing the password for the specified user account or password for the specified user account. Optional. If both are omitted, empty password is assumed.

**--domain <domain>**

User domain for Windows guests. Optional.

**-v|--verbose**

Makes the subcommand execution more verbose. Optional

**-q|--quiet**

Makes the subcommand execution quieter. Optional.

The first set of subcommands are as follows:

- `run`: Executes a guest program, forwarding stdout, stderr, and stdin to and from the host until it completes.

```
VBoxManage guestcontrol <uuid|vmname> run [common-options]
            --exe <path to executable> [--timeout <msec>]
           [-E|--putenv <NAME>[=<VALUE>]] [--unquoted-args]
           [--ignore-operhaned-processes] [--profile]
           [--no-wait-stdout|--wait-stdout]
           [--no-wait-stderr|--wait-stderr]
           [--dos2unix] [--unix2dos]
            -- <program/arg0> [argument1] ... [argumentN]]
```

**<uuid|vmname>**

Specifies the VM UUID or VM name. Mandatory.

**--exe <path to executable>**

Specifies the absolute path of the executable on the guest OS file system. Mandatory. For example: `C:\Windows\System32\calc.exe`.

**--timeout <msec>**

Specifies the maximum time, in microseconds, that the executable can run, during which `VBoxManage` receives its output. Optional. If unspecified, `VBoxManage` waits indefinitely for the process to end, or an error occurs.

**-E|--putenv <NAME>=<VALUE>**

Sets, modifies, and unsets environment variables in the environment in which the program will run. Optional.

The guest process is created with the standard default guest OS environment. Use this option to modify that default environment. To set or modify a variable use: `<NAME>=<VALUE>`. To unset a variable use: `<NAME>=`

Any spaces in names and values should be enclosed by quotes.

To set, modify, and unset multiple variables, use multiple instances of the `--E|--putenv` option.

**--unquoted-args**

Disables escaped double quoting, such as \“fred\“, on arguments passed to the executed program. Optional.

**--ignore-operhaned-processes**

Ignore orphaned processes. Not yet implemented. Optional.

**--profile**

Use Profile. Not yet implemented. Optional.

**--no-wait-stdout|--wait-stdout**

Does not wait or waits until the guest process ends and receives its exit code and reason/flags. In the case of `--wait-stdout`, `VBoxManage` receives its stdout while the process runs. Optional.

**--no-wait-stderr|--wait-stderr**

Does not wait or waits until the guest process ends and receives its exit code, error messages, and flags. In the case of `--wait-stderr`, `VBoxManage` receives its stderr while the process runs. Optional.

**--dos2unix**

Converts output from DOS/Windows guests to UNIX/Linux-compatible line endings, CR + LF to LF. Not yet implemented. Optional.

**--unix2dos**

Converts output from a UNIX/Linux guests to DOS/Windows-compatible line endings, LF to CR + LF. Not yet implemented. Optional.

**[-- <program/arg0> [<argument1>] ... [<argumentN>]]**

Specifies the program name, followed by one or more arguments to pass to the program. Optional.

Any spaces in arguments should be enclosed by quotes.

**Note:** On Windows there are certain limitations for graphical applications. See chapter 14, *Known Limitations*, page 301.

Examples of using the `guestcontrol run` command are as follows:

```
VBoxManage --nologo guestcontrol "My VM" run --exe "/bin/ls"
          --username foo --passwordfile bar.txt --wait-exit --wait-stdout -- -l /usr

VBoxManage --nologo guestcontrol "My VM" run --exe "c:\\windows\\system32\\ipconfig.exe"
          --username foo --passwordfile bar.txt --wait-exit --wait-stdout
```

Note that the double backslashes in the second example are only required on UNIX hosts.

**Note:** For certain commands a user name of an existing user account on the guest must be specified. Anonymous executions are not supported for security reasons. A user account password, however, is optional and depends on the guest's OS security policy or rules. If no password is specified for a given user name, an empty password will be used. On certain OSes like Windows the security policy may needs to be adjusted in order to allow user accounts with an empty password set. Also, global domain rules might apply and therefore cannot be changed.

Starting at Oracle VM VirtualBox 4.1.2 guest process execution by default is limited to serve up to five guest processes at a time. If a new guest process gets started which would exceed this limit, the oldest not running guest process will be discarded in order to be able to run that new process. Also, retrieving output from this old guest process will not be possible anymore then. If all five guest processes are still active and running, starting a new guest process will result in an appropriate error message.

To raise or lower the guest process execution limit, either use the guest property `/VirtualBox/GuestAdd/VBoxService/--control-procs-max-kept` or `VBoxService` command line by specifying `--control-procs-max-kept` needs to be modified. A restart of the guest OS is required afterwards. To serve unlimited guest processes, a value of `0` needs to be set, but this is not recommended.

- `start`: Executes a guest program until it completes.

```
VBoxManage guestcontrol <uuid|vmname> start [common-options]
           [--exe <path to executable>] [--timeout <msec>]
           [-E|--putenv <NAME>[=<VALUE>]] [--unquoted-args]
           [--ignore-operhaned-processes] [--profile]
            -- <program/arg0> [argument1] ... [argumentN]]
```

Where the options are as follows:

**`<uuid|vmname>`**

Specifies the VM UUID or VM name. Mandatory.

**`--exe <path to executable>`**

Specifies the absolute path of the executable on the guest OS file system. Mandatory. For example: `C:\Windows\System32\calc.exe`

**`--timeout <msec>`**

Specifies the maximum time, in microseconds, that the executable can run. Optional. If unspecified, `VBoxManage` waits indefinitely for the process to end, or an error occurs.

**`-E|--putenv <NAME>=<VALUE>`**

Sets, modifies, and unsets environment variables in the environment in which the program will run. Optional.

The guest process is created with the standard default guest OS environment. Use this option to modify that default environment. To set or modify a variable use: `<NAME>=<VALUE>`. To unset a variable use: `<NAME>=`

Any spaces in names and values should be enclosed by quotes.

To set, modify, or unset multiple variables, use multiple instances of the `--E|--putenv` option.

**`--unquoted-args`**

Disables escaped double quoting, such as \“fred\“, on arguments passed to the executed program. Optional.

**`--ignore-operhaned-processes`**

Ignores orphaned processes. Not yet implemented. Optional.

**--profile**

Use a profile. Not yet implemented. Optional.

**[-- <program/arg0> [<argument1>] ... [<argumentN>]]**

Specifies the program name, followed by one or more arguments to pass to the program. Optional.

Any spaces in arguments should be enclosed by quotes.

**Note:** On Windows there are certain limitations for graphical applications. See chapter 14, *Known Limitations*, page 301.

Examples of using the `guestcontrol start` command are as follows:

```
VBoxManage --nologo guestcontrol "My VM" start --exe "/bin/ls"
          --username foo --passwordfile bar.txt --wait-exit --wait-stdout -- -l /usr

VBoxManage --nologo guestcontrol "My VM" start --exe "c:\\windows\\system32\\ipconfig.exe"
          --username foo --passwordfile bar.txt --wait-exit --wait-stdout
```

Note that the double backslashes in the second example are only required on UNIX hosts.

**Note:** For certain commands a user name of an existing user account on the guest must be specified. Anonymous executions are not supported for security reasons. A user account password, however, is optional and depends on the guest's OS security policy or rules. If no password is specified for a given user name, an empty password will be used. On certain OSes like Windows the security policy may needs to be adjusted in order to allow user accounts with an empty password set. Also, global domain rules might apply and therefore cannot be changed.

Starting at Oracle VM VirtualBox 4.1.2 guest process execution by default is limited to serve up to five guest processes at a time. If a new guest process gets started which would exceed this limit, the oldest not running guest process will be discarded in order to be able to run that new process. Also, retrieving output from this old guest process will not be possible anymore then. If all five guest processes are still active and running, starting a new guest process will result in an appropriate error message.

To raise or lower the guest process execution limit, either use the guest property `/VirtualBox/GuestAdd/VBoxService/--control-procs-max-kept` or `VBoxService` command line by specifying `--control-procs-max-kept` needs to be modified. A restart of the guest OS is required afterwards. To serve unlimited guest processes, a value of `0` needs to be set, but this is not recommended.

- `copyfrom`: Copies files from the guest to the host file system. Only available with Guest Additions 4.0 or later installed.

```
VBoxManage guestcontrol <uuid|vmname> copyfrom [common-options]
          [--follow] [--R|recursive]
           --target-directory <host-dst-dir>
           <guest-src0> [<guest-src1> [...]]
```

Where the parameters are as follows:

**<uid|vmname>**

Specifies the VM UUID or VM name. Mandatory.

**--follow**

Enables symlink following on the guest file system. Optional.

**-R|--recursive**

Enables recursive copying of files and directories from the specified guest file system directory. Optional.

**--target-directory <host-dst-dir>**

Specifies the absolute path of the host file system destination directory. Mandatory. For example: `C:\Temp`.

**<guest-src0> [<guest-src1> [...]]**

Specifies the absolute paths of guest file system files to be copied. Mandatory. For example: `C:\Windows\System32\calc.exe`. Wildcards can be used in the expressions. For example: `C:\Windows\System*\*.dll`.

- `copyto`: Copies files from the host to the guest file system. Only available with Guest Additions 4.0 or later installed.

```
VBoxManage guestcontrol <uuid|vmname> copyto [common-options]
           [--follow] [--R|recursive]
            --target-directory <guest-dst>
            <host-src0> [<host-src1> [...]]
```

Where the parameters are as follows:

**<uuid|vmname>**

Specifies the VM UUID or VM name. Mandatory.

**--follow**

Enables symlink following on the host file system. Optional.

**-R|--recursive**

Enables recursive copying of files and directories from the specified host file system directory. Optional.

**--target-directory <guest-dst>**

Specifies the absolute path of the guest file system destination directory. Mandatory. For example: `C:\Temp`.

**<host-src0> [<host-src1> [...]]**

Specifies the absolute paths of host file system files to be copied. Mandatory. For example: `C:\Windows\System32\calc.exe`. Wildcards can be used in the expressions. For example: `C:\Windows\System*\*.dll`.

- `md|mkdir|createdir|createdirectory`: Creates one or more directories on the guest file system. Only available with Guest Additions 4.0 or later installed.

```
VBoxManage guestcontrol <uuid|vmname>  md|mkdir|createdir|createdirectory [common-options]
           [--parents] [--mode <mode>]
           <guest-dir0> [<guest-dir1> [...]]
```

Where the parameters are as follows:

**`<uuid|vmname>`**

Specifies the VM UUID or VM name. Mandatory.

**`--parents`**

Creates any absent parent directories of the specified directory. Optional.

For example: If specified directory is `D:\Foo\Bar` and `D:\Foo` is absent, it will be created. In such a case, had the `--parents` option not been used, this command would have failed.

**`--mode <mode>`**

Specifies the permission mode on the specified directories, and any parents, if the `--parents` option is used. Currently octal modes only, such as. `0755`, are supported.

**`<guest-dir0> [<guest-dir1> [...]]`**

Specifies a list of absolute paths of directories to be created on guest file system. Mandatory. For example: `D:\Foo\Bar`.

All parent directories must already exist unless the switch `--parents` is used. For example, in the above example `D:\Foo`. The specified user must have sufficient rights to create the specified directories, and any parents that need to be created.

- `rmdir|removedir|removedirectory`: Deletes specified guest file system directories. Only available with installed Guest Additions 4.3.2 and later.

```
VBoxManage guestcontrol <uuid|vmname> rmdir|removedir|removedirectory [common-options]
          [--recursive|-R]
           <guest-dir0> [<guest-dir1> [...]]
```

Where the parameters are as follows:

**`<uuid|vmname>`**

Specifies the VM UUID or VM name. Mandatory.

**`--recursive`**

Recursively removes directories and contents. Optional.

**`<guest-dir0> [<guest-dir1> [...]]`**

Specifies a list of the absolute paths of directories to be deleted on guest file system. Mandatory. Wildcards are allowed. For example: `D:\Foo\*Bar`. The specified user must have sufficient rights to delete the specified directories.

- `rm|removefile`: Deletes specified files on the guest file system. Only available with installed Guest Additions 4.3.2 and later.

```
VBoxManage guestcontrol <uuid|vmname> rm|removefile [common-options]
           [-f|--force]
            <guest-file0> [<guest-file1> [...]]
```

Where the parameters are as follows:

**<uuid|vmname>**

Specifies the VM UUID or VM name. Mandatory.

**-f|--force**

Enforce operation and override any requests for confirmations. Optional.

**<guest-file0> [<guest-file1> [...]]**

Specifies a list of absolute paths of files to be deleted on guest file system. Mandatory. Wildcards are allowed. For example: `D:\Foo\Bar\text*.txt`. The specified user should have sufficient rights to delete the specified files.

- `mv|move|ren|rename`: Renames files and/or directories on the guest file system. Only available with installed Guest Additions 4.3.2 and later.

```
VBoxManage guestcontrol <uuid|vmname> mv|move|ren|rename [common-options]
           <guest-source0> [<guest-source1> [...]] <guest-dest>
```

Where the parameters are as follows:

**<uuid|vmname>**

Specifies the VM UUID or VM name. Mandatory.

**<guest-source0> [<guest-source1> [...]]**

Specifies absolute paths of files or a single directory to be moved and renamed on guest file system. Mandatory. Wildcards are allowed in file names. The specified user should have sufficient rights to access the specified files.

**<dest>**

Specifies the absolute path of the destination file or directory to which the files are to be moved. Mandatory. If only one file to be moved, <dest> can be file or directory, else it must be a directory. The specified user must have sufficient rights to access the destination file or directory.

- `mktemp|createtemp|createtemporary`: Creates a temporary file or directory on the guest file system, to assist subsequent copying of files from the host to the guest file systems. By default, the file or directory is created in the guest's platform specific temp directory. Not currently supported. Only available with installed Guest Additions 4.2 and later.

```
VBoxManage guestcontrol <uuid|vmname> mktemp|createtemp|createtemporary [common-options]
           [--directory] [--secure] [--mode <mode>] [--tmpdir <directory>]
            <template>
```

The parameters are as follows:

**<uuid|vmname>**

Specifies the VM UUID or VM name. Mandatory.

**--directory**

Creates a temporary directory instead of a file, specified by the <template> parameter. Optional.

**--secure**

Enforces secure file and directory creation. Optional. The permission mode is set to `0755`. Operation fails if it cannot be performed securely.

**--mode <mode>**

Specifies the permission mode of the specified directory. Optional. Currently only octal modes, such as `0755`, are supported.

**--tmpdir <directory>**

Specifies the absolute path of the directory on the guest file system where the file or directory specified will be created. Optional. If unspecified, the platform-specific temp directory is used.

**<template>**

Specifies a file name without a directory path, containing at least one sequence of three consecutive X characters, or ending in X. Mandatory.

- `stat`: Displays file or file system statuses on the guest.

```
VBoxManage guestcontrol <uuid|vmname> stat [common-options]
          <file0> [<file1> [...]]
```

Where the parameters are as follows:

**<uuid|vmname>**

Specifies the VM UUID or VM name. Mandatory.

**<file0> [<file1> [...]]**

Specifies absolute paths of files or file systems on the guest file system. Mandatory. For example: `/home/foo/a.out`. The specified user should have sufficient rights to access the specified files or file systems.

The second set of subcommands is of the form:

```
VBoxManage guestcontrol <uuid|vmname> <sub-command>
          [-v|--verbose] [-q|quiet] ...
```

The common options are as follows:

```
          [-v|--verbose] [-q|--quiet]
```

Details of the common options for the second set of subcommands are as follows:

**-v|--verbose**

Makes the subcommand execution more verbose. Optional.

**-q|--quiet**

Makes the subcommand execution quieter. Optional.

The second set of subcommands are as follows:

- `list`: Lists guest control configuration and status data. For example: open guest sessions, guest processes, and files.

```
VBoxManage guestcontrol <uuid|vmname> list [common-opts]
          <all|sessions|processes|files>
```

Where the parameters are as follows:

**<uuid|vmname>**

Specifies the VM UUID or VM name. Mandatory.

**all|sessions|processes|files**

Indicates whether to list all available data or guest sessions, processes or files. Mandatory.

- `closeprocess`: Terminates guest processes specified by PIDs running in a guest session, specified by the session ID or name.

```
VBoxManage guestcontrol <uuid|vmname> closeprocess [common-options]
          --session-id <ID> | --session-name <name or pattern>
          <PID0> [<PID1> [...]]
```

Where the parameters are as follows:

**<uuid|vmname>**

Specifies the VM UUID or VM name. Mandatory.

**--session-id <ID>**

Specifies the guest session by its ID. Optional.

**--session-name <name or pattern>**

Specifies the guest session by its name, or multiple sessions using a pattern containing wildcards. Optional.

**<PID0> [<PID1> [...]]**

Specifies a list of process identifiers (PIDs) of guest processes to be terminated. Mandatory.

- `closesession`: Closes specified guest sessions, specified either by session ID or name.

```
VBoxManage guestcontrol <uuid|vmname> closesession [common-options]
          --session-id <ID> | --session-name <name or pattern> | --all
```

Where the parameters are as follows:

**<uuid|vmname>**

Specifies the VM UUID or VM name. Mandatory.

**--session-id <ID>**

Specifies the guest session to be closed by ID. Optional.

**--session-name <name or pattern>**

Specifies the guest session to be closed by name. Optional. Multiple sessions can be specified by using a pattern containing wildcards.

**--all**

Close all guest sessions. Optional.

- `updatega|updateadditions|updateguestadditions`: Ugrades Guest Additions already installed on the guest. Only available for already installed Guest Additions 4.0 and later.

```
VBoxManage guestcontrol <uuid|vmname> updatega|updateadditions|updateguestadditions
          [common-options]
          [--source <New .ISO path>]
          [--wait-start]
          [-- <argument0> [<argument1> [...]]]
```

Where the parameters are as follows:

**<uuid|vmname>**

Specifies the VM UUID or VM name. Mandatory.

**--source <New .ISO path>**

Specifies the absolute path on the guest file system of the .ISO file for the Guest Additions update. Mandatory.

**--wait-start**

Indicates that `VBoxManage` starts the usual updating process on the guest and then waits until the actual Guest Additions updating begins, at which point `VBoxManage` self-terminates. Optional.

Default behavior is that `VBoxManage` waits for completion of the Guest Additions update before terminating. Use of this option is sometimes necessary, as a running `VBoxManage` can affect the interaction between the installer and the guest OS.

**[-- <argument0> [<argument1> [...]]]**

Specifies optional command line arguments to be supplied to the Guest Additions updater. Useful for retrofitting features which are not currently installed.

Arguments containing spaces should be enclosed by quotes.

- watch: Prints current guest control activity.

```
VBoxManage guestcontrol <uuid|vmname> watch [common-options]
```

Where the parameters are as follows:

**<uuid|vmname>**

Specifies the VM UUID or VM name. Mandatory.

## 8.36 VBoxManage metrics

This command supports monitoring the usage of system resources. Resources are represented by various metrics associated with the host system or a particular VM. For example, the host system has a CPU/Load/User metric that shows the percentage of time CPUs spend executing in user mode over a specific sampling period.

Metric data is collected and retained internally. It may be retrieved at any time with the VBoxManage metrics query subcommand. The data is available as long as the background VBoxSVC process is alive. That process terminates shortly after all VMs and frontends have been closed.

By default no metrics are collected at all. Metrics collection does not start until VBoxManage metrics setup is invoked with a proper sampling interval and the number of metrics to be retained. The interval is measured in seconds. For example, to enable collecting the host processor and memory usage metrics every second and keeping the five most current samples, the following command can be used:

```
VBoxManage metrics setup --period 1 --samples 5 host CPU/Load,RAM/Usage
```

Metric collection can only be enabled for started VMs. Collected data and collection settings for a particular VM will disappear as soon as it shuts down. Use the VBoxManage metrics list subcommand to see which metrics are currently available. You can also use the --list option with any subcommand that modifies metric settings to find out which metrics were affected.

Note that the VBoxManage metrics setup subcommand discards all samples that may have been previously collected for the specified set of objects and metrics.

To enable or disable metrics collection without discarding the data, VBoxManage metrics enable and VBoxManage metrics disable subcommands can be used. Note that these subcommands expect metrics as parameters, not submetrics such as CPU/Load or RAM/Usage. In other words enabling CPU/Load/User while disabling CPU/Load/Kernel is not supported.

The host and VMs have different sets of associated metrics. Available metrics can be listed with VBoxManage metrics list subcommand.

A complete metric name may include an aggregate function. The name has the following form: Category/Metric[/SubMetric][:aggregate]. For example, RAM/Usage/Free:min stands for the minimum amount of available memory over all retained data if applied to the host object.

Subcommands may apply to all objects and metrics or can be limited to one object and a list of metrics. If no objects or metrics are given in the parameters, the subcommands will apply to all available metrics of all objects. You may use an asterisk "∗" to explicitly specify that the command should be applied to all objects or metrics. Use host as the object name to limit the scope of the command to host-related metrics. To limit the scope to a subset of metrics, use a metric list with names separated by commas.

For example, to query metric data on the CPU time spent in user and kernel modes by the virtual machine named test, use the following command:

```
VBoxManage metrics query test CPU/Load/User,CPU/Load/Kernel
```

The following list summarizes the available subcommands:

**list**

Shows the parameters of the currently existing metrics. Note that VM-specific metrics are only available when a particular VM is running.

**setup**

Sets the interval between taking two samples of metric data and the number of samples retained internally. The retained data is available for displaying with the `query` subcommand. The `--list` option shows which metrics have been modified as the result of the command execution.

**enable**

Resumes data collection after it has been stopped with the `disable` subcommand. Note that specifying submetrics as parameters will not enable underlying metrics. Use `--list` to find out if the command worked as expected.

**disable**

Suspends data collection without affecting collection parameters or collected data. Note that specifying submetrics as parameters will not disable underlying metrics. Use `--list` to find out if the command worked as expected.

**query**

Retrieves and displays the currently retained metric data.

**Note:** The `query` subcommand does not remove or flush retained data. If you query often enough you will see how old samples are gradually being phased out by new samples.

**collect**

Sets the interval between taking two samples of metric data and the number of samples retained internally. The collected data is displayed periodically until Ctrl+C is pressed, unless the `--detach` option is specified. With the `--detach` option, this subcommand operates the same way as `setup` does. The `--list` option shows which metrics match the specified filter.

## 8.37 VBoxManage natnetwork

NAT networks use the Network Address Translation (NAT) service, which works in a similar way to a home router. It groups systems using it into a network and prevents outside systems from directly accessing those inside, while letting systems inside communicate with each other and outside systems using TCP and UDP over IPv4 and IPv6.

A NAT service is attached to an internal network. Virtual machines to make use of one should be attached to it. The name of an internal network is chosen when the NAT service is created, and the internal network will be created if it does not already exist. The following is an example command to create a NAT network:

```
VBoxManage natnetwork add --netname natnet1 --network "192.168.15.0/24" --enable
```

Here, `natnet1` is the name of the internal network to be used and `192.168.15.0/24` is the network address and mask of the NAT service interface. By default, in this static configuration the gateway will be assigned the address 192.168.15.1, the address after the interface address, though this is subject to change.

To add a DHCP server to the NAT network after creation, run the following command:

```
VBoxManage natnetwork modify --netname natnet1 --dhcp on
```

The subcommands for `VBoxManage natnetwork` are as follows:

```
VBoxManage natnetwork add --netname <name>
                          [--network <network>]
                          [--enable|--disable]
                          [--dhcp on|off]
                          [--port-forward-4 <rule>]
                          [--loopback-4 <rule>]
                          [--ipv6 on|off]
                          [--port-forward-6 <rule>]
                          [--loopback-6 <rule>]
```

`VBoxManage natnetwork add`: Creates a new internal network interface, and adds a NAT network service. This command is a prerequisite for enabling attachment of VMs to the NAT network. Parameters are as follows:

**`--netname <name>`**

Where <name> is the name of the new internal network interface on the host OS.

**`--network <network>`**

Where <network> specifies the static or DHCP network address and mask of the NAT service interface. The default is a static network address.

**`--enable|--disable`**

Enables and disables the NAT network service.

**`--dhcp on|off`**

Enables and disables a DHCP server specified by `--netname`. Use of this option also indicates that it is a DHCP server.

**`--port-forward-4 <rule>`**

Enables IPv4 port forwarding, with a rule specified by <rule>.

**`--loopback-4 <rule>`**

Enables the IPv4 loopback interface, with a rule specified by <rule>.

**`--ipv6 on|off`**

Enables and disables IPv6. The default setting is IPv4, disabling IPv6 enables IPv4.

**--port-forward-6 <rule>**

Enables IPv6 port forwarding, with a rule specified by <rule>.

**--loopback-6 <rule>**

Enables the IPv6 loopback interface, with a rule specified by <rule>.

```
VBoxManage natnetwork remove --netname <name>
```

`VBoxManage natnetwork remove`: Removes a NAT network service. Parameters are as follows:

**--netname <name>**

Where <name> specifies an existing NAT network service. Does not remove any DHCP server enabled on the network.

```
VBoxManage natnetwork modify --netname <name>
                             [--network <network>]
                             [--enable|--disable]
                             [--dhcp on|off]
                             [--port-forward-4 <rule>]
                             [--loopback-4 <rule>]
                             [--ipv6 on|off]
                             [--port-forward-6 <rule>]
                             [--loopback-6 <rule>]
```

`VBoxManage natnetwork modify`: Modifies an existing NAT network service. Parameters are as follows:

**--netname <name>**

Where <name> specifies an existing NAT network service.

**--network <network>**

Where <network> specifies the new static or DHCP network address and mask of the NAT service interface. The default is a static network address.

**--enable|--disable**

Enables and disables the NAT network service.

**--dhcp on|off**

Enables and disables a DHCP server. If a DHCP server is not present, using enable adds a new DHCP server.

**--port-forward-4 <rule>**

Enables IPv4 port forwarding, with a rule specified by <rule>.

**--loopback-4 <rule>**

Enables the IPv4 loopback interface, with a rule specified by <rule>.

**--ipv6 on|off**

Enables and disables IPv6. The default setting is IPv4, disabling IPv6 enables IPv4.

**--port-forward-6 <rule>**

Enables IPv6 port forwarding, with a rule specified by <rule>.

**--loopback-6 <rule>**

Enables IPv6 loopback interface, with a rule specified by <rule>.

```
VBoxManage natnetwork start --netname <name>
```

`VBoxManage natnetwork start`: Starts the specified NAT network service and any associated DHCP server. Parameters are as follows:

**--netname <name>**

Where <name> specifies an existing NAT network service.

```
VBoxManage natnetwork stop --netname <name>
```

`VBoxManage natnetwork stop`: Stops the specified NAT network service and any DHCP server. Parameters are as follows:

**--netname <name>**

Where <name> specifies an existing NAT network service.

```
VBoxManage natnetwork list [<pattern>]
```

`VBoxManage natnetwork list`: Lists all NAT network services, with optional filtering. Parameters are as follows:

**[<pattern>]**

Where <pattern> is an optional filtering pattern.

## 8.38 VBoxManage hostonlyif

The `hostonlyif` command enables you to change the IP configuration of a host-only network interface. For a description of host-only networking, see chapter 6.7, *Host-Only Networking*, page 105. Each host-only interface is identified by a name and can either use the internal DHCP server or a manual IP configuration, both IP4 and IP6.

The following list summarizes the available subcommands:

**ipconfig "<name>"**

Configures a host-only interface.

**create**

Creates a new vboxnet<N> interface on the host OS. This command is essential before you can attach VMs to a host-only network.

**remove vboxnet<N>**

Removes a vboxnet<N> interface from the host OS.

## 8.39 VBoxManage dhcpserver

The `dhcpserver` commands enable you to control the DHCP server that is built into Oracle VM VirtualBox. You may find this useful when using internal or host-only networking. Theoretically, you can also enable it for a bridged network, but that may cause conflicts with other DHCP servers in your physical network.

Use the following command line options:

- If you use internal networking for a virtual network adapter of a virtual machine, use `VBoxManage dhcpserver add --netname <network_name>`, where `<network_name>` is the same network name you used with `VBoxManage modifyvm <vmname> --intnet<X> <network_name>`.
- If you use host-only networking for a virtual network adapter of a virtual machine, use `VBoxManage dhcpserver add --ifname <hostonly_if_name>` instead, where `<hostonly_if_name>` is the same host-only interface name you used with `VBoxManage modifyvm <vmname> --hostonlyadapter<X> <hostonly_if_name>`.

  Alternatively, you can also use the `--netname` option as with internal networks if you know the host-only network's name. You can see the names with `VBoxManage list hostonlyifs`. See chapter 8.4, *VBoxManage list*, page 132.

The following additional parameters are required when first adding a DHCP server:

- With `--ip`, specify the IP address of the DHCP server.
- With `--netmask`, specify the netmask of the network.
- With `--lowerip` and `--upperip`, you can specify the lowest and highest IP address that the DHCP server will assign to clients.

You can specify additional DHCP options with the `--options` command option. Use `--id` and `--value` to configure a number and string pair corresponding to the DHCP option. Use `--remove` to remove a DHCP option.

The `--vm` and `--nic` settings enable you to configure DHCP options for a specific network adapter used by the named VM.

Finally, you must specify `--enable` or the DHCP server will be created in the disabled state and will not be running.

After this, Oracle VM VirtualBox will automatically start the DHCP server for the specified internal network or host-only network as soon as the first virtual machine which uses that network is started.

Use `VBoxManage dhcpserver remove` with the given `--netname <network_name>` or `--ifname <hostonly_if_name>` to remove the DHCP server for the specified internal network or host-only network.

To modify the settings of a DHCP server created using `VBoxManage dhcpserver add`, you can use `VBoxManage dhcpserver modify` for a given network or host-only interface name. This has the same parameters as `VBoxManage dhcpserver add`.

## 8.40 VBoxManage usbdevsource

The usbdevsource commands enable you to add and remove USB devices globally.
The following command adds a USB device.

```
VBoxManage usbdevsource add <source name>
                           --backend <backend>
                           --address <address>
```

Where the command line options are as follows:

- <source name>: Specifies the ID of the source USB device to be added. Mandatory.
- --backend <backend>: Specifies the USB proxy service backend to use. Mandatory.
- --address <address>: Specifies the backend specific address. Mandatory.

The following command removes a USB device.

```
VBoxManage usbdevsource remove <source name>
```

Where the command line options are as follows:

- <source name>: Specifies the ID of the source USB device to be removed. Mandatory.

## 8.41 VBoxManage mediumio

Medium content access.

### Synopsis

```
VBoxManage mediumio <[--disk=uuid|filename] | [--dvd=uuid|filename] | [--floppy=uuid|filenam
    [--password-file-|filename] formatfat [--quick]
VBoxManage mediumio <[--disk=uuid|filename] | [--dvd=uuid|filename] | [--floppy=uuid|filenam
    [--password-file-|filename] cat [--hex] [--offset=byte-offset] [--size=bytes]
    [--output=-|filename]
VBoxManage mediumio <[--disk=uuid|filename] | [--dvd=uuid|filename] | [--floppy=uuid|filenam
    [--password-file-|filename] stream [--format=image-format] [--variant=image-variant]
    [--output=-|filename]
```

### Description

#### Common options

The subcommands of mediumio all operate on a medium which need to be specified, optionally with an encryption password. The following common options can be placed before or after the sub-command:

**–disk=*uuid*|*filename***
Either the UUID or filename of a harddisk image, e.g. VDI, VMDK, VHD, ++.

**–dvd=*uuid*|*filename***
Either the UUID or filename of a DVD image, e.g. ISO, DMG, CUE.

**–floppy=*uuid*|*filename***
Either the UUID or filename of a floppy image, e.g. IMG.

**–password-file=-|*filename***
The name of a file containing the medium encryption password. If - is specified, the password will be read from stdin.

### mediumio formatfat

```
VBoxManage mediumio <[--disk=uuid|filename] | [--dvd=uuid|filename] | [--floppy=uuid|filenam
    [--password-file-|filename] formatfat [--quick]
```

Formats a floppy medium with the FAT file system. This will erase the content of the medium.

**--quick**
Quickformat the medium.

### mediumio cat

```
VBoxManage mediumio <[--disk=uuid|filename] | [--dvd=uuid|filename] | [--floppy=uuid|filenam
    [--password-file-|filename] cat [--hex] [--offset=byte-offset] [--size=bytes]
    [--output=-|filename]
```

Dumps the medium content to stdout or the specified file.

**--hex**
Dump as hex bytes.

**--offset**
The byte offset in the medium to start.

**--size**
The number of bytes to dump.

**--output**
The output filename. As usual - is take to mean stdout.

### mediumio stream

```
VBoxManage mediumio <[--disk=uuid|filename] | [--dvd=uuid|filename] | [--floppy=uuid|filenam
    [--password-file-|filename] stream [--format=image-format] [--variant=image-variant]
    [--output=-|filename]
```

Converts the medium to a streamable format and dumps it to the given output.

**--format**
The format of the destination image.

**--variant**
The medium variant for the destination.

**--output**
The output filename. As usual - is take to mean stdout.

## 8.42 VBoxManage debugvm

Introspection and guest debugging.

## Synopsis

```
VBoxManage debugvm <uuid|vmname> dumpvmcore [--filename=name]
VBoxManage debugvm <uuid|vmname> info <item> [args...]
VBoxManage debugvm <uuid|vmname> injectnmi
VBoxManage debugvm <uuid|vmname> log [[--release] | [--debug]] [group-settings...]
VBoxManage debugvm <uuid|vmname> logdest [[--release] | [--debug]] [destinations...]
VBoxManage debugvm <uuid|vmname> logflags [[--release] | [--debug]] [flags...]
VBoxManage debugvm <uuid|vmname> osdetect
VBoxManage debugvm <uuid|vmname> osinfo
VBoxManage debugvm <uuid|vmname> osdmesg [--lines=lines]
VBoxManage debugvm <uuid|vmname> getregisters [--cpu=id] [reg-set.reg-name...]
VBoxManage debugvm <uuid|vmname> setregisters [--cpu=id] [reg-set.reg-name=value...]
VBoxManage debugvm <uuid|vmname> show [[--human-readable] | [--sh-export] |
    [--sh-eval] | [--cmd-set]] [settings-item...]
VBoxManage debugvm <uuid|vmname> stack [--cpu=id]
VBoxManage debugvm <uuid|vmname> statistics [--reset] [--descriptions] [--pattern=pattern]
```

## Description

The “debugvm” commands are for experts who want to tinker with the exact details of virtual machine execution. Like the VM debugger described in chapter 12.1.3, *The Built-In VM Debugger*, page 278, these commands are only useful if you are very familiar with the details of the PC architecture and how to debug software.

### Common options

The subcommands of `debugvm` all operate on a running virtual machine:

***uuid|vmname***
: Either the UUID or the name (case sensitive) of a VM.

### debugvm dumpvmcore

```
VBoxManage debugvm <uuid|vmname> dumpvmcore [--filename=name]
```

Creates a system dump file of the specified VM. This file will have the standard ELF core format (with custom sections); see chapter 12.1.4, *VM Core Format*, page 280.

This corresponds to the `writecore` command in the debugger.

**`--filename=<`*`filename`*`>`**
: The name of the output file.

### debugvm info

```
VBoxManage debugvm <uuid|vmname> info <item> [args...]
```

Displays info items relating to the VMM, device emulations and associated drivers.

This corresponds to the `info` command in the debugger.

***info***
: Name of the info item to display. The special name `help` will list all the available info items and hints about optional arguments.

***args***
: Optional argument string for the info item handler. Most info items does not take any extra arguments. Arguments not recognized are generally ignored.

### debugvm injectnmi

```
VBoxManage debugvm <uuid|vmname> injectnmi
```

Causes a non-maskable interrupt (NMI) to be injected into the guest. This might be useful for certain debugging scenarios. What happens exactly is dependent on the guest operating system, but an NMI can crash the whole guest operating system. Do not use unless you know what you're doing.

### debugvm log

```
VBoxManage debugvm <uuid|vmname> log [[--release] | [--debug]] [group-settings...]
```

Changes the group settings for either debug (`--debug`) or release (`--release`) logger of the VM process.

The *group-settings* are typically strings on the form `em.e.f.l`, `hm=~0` and `-em.f`. Basic wildcards are supported for group matching. The `all` group is an alias for all the groups.

Please do keep in mind that the group settings are applied as modifications to the current ones.

This corresponds to the `log` command in the debugger.

### debugvm logdest

```
VBoxManage debugvm <uuid|vmname> logdest [[--release] | [--debug]] [destinations...]
```

Changes the destination settings for either debug (`--debug`) or release (`--release`) logger of the VM process. For details on the destination format, the best source is src/VBox/Runtime/common/log/log.cpp.

The *destinations* is one or more mnemonics, optionally prefixed by "no" to disable them. Some of them take values after a ":" or "=" separator. Multiple mnemonics can be separated by space or given as separate arguments on the command line.

List of available destination:

**file[=<*file*>], nofile**
: Specifies a log file. It no filname is given, one will be generated based on the current UTC time and VM process name and placed in the current directory of the VM process. Note that this will currently not have any effect if the log file has already been opened.

**dir=<*directory*>, nodir**
: Specifies the output directory for log files. Note that this will currently not have any effect if the log file has already been opened.

**history=<*count*>, nohistory**
: A non-zero value enables log historization, with the value specifying how many old log files to keep.

**histsize=<*bytes*>**
: The max size of a log file before it is historized. Default is infinite.

**histtime=<*seconds*>**
: The max age (in seconds) of a log file before it is historized. Default is infinite.

**ringbuffer, noringbuffer**
: Only log to the log buffer until an explicit flush (e.g. via an assertion) occurs. This is fast and saves diskspace.

**stdout, nostdout**
: Write the log content to standard output.

**stdout, nostdout**
Write the log content to standard error.

**debugger, nodebugger**
Write the log content to the debugger, if supported by the host OS.

**com, nocom**
Writes logging to the COM port. This is only applicable for raw-mode and ring-0 logging.

**user, nouser**
Custom destination which has no meaning to VM processes..

This corresponds to the `logdest` command in the debugger.

### debugvm logflags

```
VBoxManage debugvm <uuid|vmname> logflags [[--release] | [--debug]] [flags...]
```

Changes the flags on either debug (`--debug`) or release (`--release`) logger of the VM process. Please note that the modifications are applied onto the existing changes, they are not replacing them.

The *flags* are a list of flag mnemonics, optionally prefixed by a "no", "i", "~" or "-" to negate their meaning. The "+" prefix can be used to undo previous negation or use as a separator, though better use whitespace or separate arguments for that.

List of log flag mnemonics, with their counter form where applicable (asterisk indicates defaults):

**enabled*, disabled**
Enables or disables logging.

**buffered, unbuffered***
Enabling buffering of log output before it hits the destinations.

**writethrough(/writethru)**
Whether to open the destination file with writethru buffering settings or not.

**flush**
Enables flushing of the output file (to disk) after each log statement.

**lockcnts**
Prefix each log line with lock counts for the current thread.

**cpuid**
Prefix each log line with the ID of the current CPU.

**pid**
Prefix each log line with the current process ID.

**flagno**
Prefix each log line with the numberic flags corresponding to the log statement.

**flag**
Prefix each log line with the flag mnemonics corresponding to the log statement.

**groupno**
Prefix each log line with the log group number for the log statement producing it.

**group**
Prefix each log line with the log group name for the log statement producing it.

**tid**
Prefix each log line with the current thread identifier.

**thread**
Prefix each log line with the current thread name.

**time**
Prefix each log line with the current UTC wall time.

**timeprog**
Prefix each log line with the current monotonic time since the start of the program.

**msprog**
Prefix each log line with the current monotonic timestamp value in milliseconds since the start of the program.

**ts**
Prefix each log line with the current monotonic timestamp value in nanoseconds.

**tsc**
Prefix each log line with the current CPU timestamp counter (TSC) value.

**rel, abs***
Selects the whether `ts` and `tsc` prefixes should be displayed as relative to the previous log line or as absolute time.

**hex*, dec**
Selects the whether the `ts` and `tsc` prefixes should be formatted as hexadecimal or decimal.

**custom**
Custom log prefix, has by default no meaning for VM processes.

**usecrlf, uself***
Output with DOS style (CRLF) or just UNIX style (LF) line endings.

**overwrite*, append**
Overwrite the destination file or append to it.

This corresponds to the `logflags` command in the debugger.

### debugvm osdetect

```
VBoxManage debugvm <uuid|vmname> osdetect
```

Make the VMM's debugger facility (re)-detect the guest operating system (OS). This will first load all debugger plug-ins.
This corresponds to the `detect` command in the debugger.

### debugvm osinfo

```
VBoxManage debugvm <uuid|vmname> osinfo
```

Displays information about the guest operating system (OS) previously detected by the VMM's debugger facility.

### debugvm osdmesg

```
VBoxManage debugvm <uuid|vmname> osdmesg [--lines=lines]
```

Displays the guest OS kernel log, if detected and supported.

**--lines=<*lines*>**
Number of lines of the log to display, counting from the end. The default is infinite.

### debugvm getregisters

```
VBoxManage debugvm <uuid|vmname> getregisters [--cpu=id] [reg-set.reg-name...]
```

Retrieves register values for guest CPUs and emulated devices.

***reg-set.reg-name***
One of more registers, each having one of the following forms:

1. register-set.register-name.sub-field
2. register-set.register-name
3. cpu-register-name.sub-field
4. cpu-register-name
5. all

The *all* form will cause all registers to be shown (no sub-fields). The registers names are case-insensitive.

**--cpu=<*id*>**
Selects the CPU register set when specifying just a CPU register (3rd and 4th form). The default is 0.

### debugvm setregisters

```
VBoxManage debugvm <uuid|vmname> setregisters [--cpu=id] [reg-set.reg-name=value...]
```

Changes register values for guest CPUs and emulated devices.

***reg-set.reg-name=value***
One of more register assignment, each having one of the following forms:

1. register-set.register-name.sub-field=value
2. register-set.register-name=value
3. cpu-register-name.sub-field=value
4. cpu-register-name=value

The value format should be in the same style as what `getregisters` displays, with the exception that both octal and decimal can be used instead of hexadecimal.

**--cpu=<*id*>**
Selects the CPU register set when specifying just a CPU register (3rd and 4th form). The default is 0.

### debugvm show

```
VBoxManage debugvm <uuid|vmname> show [[--human-readable] | [--sh-export] |
    [--sh-eval] | [--cmd-set]] [settings-item...]
```

Shows logging settings for the VM.

**--human-readable**
Selects human readable output.

**--sh-export**
Selects output format as bourne shell style `export` commands.

**--sh-eval**
Selects output format as bourne shell style `eval` command input.

**--cmd-set**
Selects output format as DOS style `SET` commands.

***settings-item***
What to display. One or more of the following:

- logdbg-settings - debug log settings.
- logrel-settings - release log settings.
- log-settings - alias for both debug and release log settings.

### debugvm stack

```
VBoxManage debugvm <uuid|vmname> stack [--cpu=id]
```

Unwinds the guest CPU stacks to the best of our ability. It is recommended to first run the `osdetect` command, as this gives both symbols and perhaps unwind information.

**--cpu=<*id*>**
Selects a single guest CPU to display the stack for. The default is all CPUs.

### debugvm statistics

```
VBoxManage debugvm <uuid|vmname> statistics [--reset] [--descriptions] [--pattern=pattern]
```

Displays or resets VMM statistics.
Retrieves register values for guest CPUs and emulated devices.

**--pattern=<*pattern*>**
DOS/NT-style wildcards patterns for selecting statistics. Multiple patterns can be specified by using the '|' (pipe) character as separator.

**--reset**
Select reset instead of display mode.

## 8.43 VBoxManage extpack

Extension package management.

## Synopsis

```
VBoxManage extpack install [--replace] <tarball>
VBoxManage extpack uninstall [--force] <name>
VBoxManage extpack cleanup
```

## Description

### extpack install

```
VBoxManage extpack install [--replace] <tarball>
```

Installs a new extension pack on the system. This command will fail if an older version of the same extension pack is already installed. The `--replace` option can be used to uninstall any old package before the new one is installed.

**`--replace`**
: Uninstall existing extension pack version.

***`tarball`***
: The file containing the extension pack to be installed.

### extpack uninstall

```
VBoxManage extpack uninstall [--force] <name>
```

Uninstalls an extension pack from the system. The subcommand will also succeed in the case where the specified extension pack is not present on the system. You can use `VBoxManage list extpacks` to show the names of the extension packs which are currently installed.

**`--force`**
: Overrides most refusals to uninstall an extension pack

***`name`***
: The name of the extension pack to be uninstalled.

### extpack cleanup

```
VBoxManage extpack cleanup
```

Used to remove temporary files and directories that may have been left behind if a previous install or uninstall command failed.

## Examples

How to list extension packs:

```
$ VBoxManage list extpacks
Extension Packs: 1
Pack no. 0:   Oracle VM VirtualBox Extension Pack
Version:      4.1.12
Revision:     77218
Edition:
Description:  USB 2.0 Host Controller, VirtualBox RDP, PXE ROM with E1000 support.
VRDE Module:  VBoxVRDP
Usable:       true
Why unusable:
```

How to remove an extension pack:

```
$ VBoxManage extpack uninstall "Oracle VM VirtualBox Extension Pack"
0%...10%...20%...30%...40%...50%...60%...70%...80%...90%...100%
Successfully uninstalled "Oracle VM VirtualBox Extension Pack".
```

## 8.44 VBoxManage unattended

Unattended guest OS installation.

### Synopsis

```
VBoxManage unattended detect <--iso=install-iso> [--machine-readable]
VBoxManage unattended install <uuid|vmname> <--iso=install-iso> [--user=login]
    [--password=password] [--password-file=file] [--full-user-name=name]
    [--key=product-key] [--install-additions] [--no-install-additions] [--additions-iso=add-
    [--install-txs] [--no-install-txs] [--validation-kit-iso=testing-iso]
    [--locale=ll_CC] [--country=CC] [--time-zone=tz] [--hostname=fqdn] [--package-selection-
    [--dry-run] [--auxiliary-base-path=path] [--image-index=number] [--script-template=file]
    [--post-install-template=file] [--post-install-command=command] [--extra-install-kernel-
    [--language=lang] [--start-vm=session-type]
```

### Description

#### unattended detect

```
VBoxManage unattended detect <--iso=install-iso> [--machine-readable]
```

Detects the guest operating system (OS) on the specified installation ISO and displays the result. This can be used as input when creating a VM for the ISO to be installed in.

**--iso=<*install-iso*>**
The installation ISO to run the detection on.

**--machine-readable**
Produce output that is simpler to parse from a script.

#### unattended install

```
VBoxManage unattended install <uuid|vmname> <--iso=install-iso> [--user=login]
    [--password=password] [--password-file=file] [--full-user-name=name]
    [--key=product-key] [--install-additions] [--no-install-additions] [--additions-iso=add-
    [--install-txs] [--no-install-txs] [--validation-kit-iso=testing-iso]
    [--locale=ll_CC] [--country=CC] [--time-zone=tz] [--hostname=fqdn] [--package-selection-
    [--dry-run] [--auxiliary-base-path=path] [--image-index=number] [--script-template=file]
    [--post-install-template=file] [--post-install-command=command] [--extra-install-kernel-
    [--language=lang] [--start-vm=session-type]
```

Reconfigures the specified VM for installation and optionally starts it up.

***uuid|vmname***
Either the UUID or the name (case sensitive) of a VM.

**--iso=<*install-iso*>**
The installation ISO to run the detection on.

**--user=<*login*>**
The login name. (default: vboxuser)

**--password=<*password*>**
The login password. This is used for the user given by `--user` as well as the root/administrator user. (default: changeme)

**--password-file=<*file*>**
Alternative to `--password` for providing the password. Special filename `stdin` can be used to read the password from standard input.

**--full-user-name=<*name*>**
The full user name. (default: –user)

**--key=<*product-key*>**
The guest OS product key. Not all guest OSes requires this.

**--install-additions, --no-install-additions**
Whether to install the VirtualBox guest additions. (default: –no-install-addations)

**--additions-iso=<*add-iso*>**
Path to the VirtualBox guest additions ISO. (default: installed/downloaded GAs)

**--install-txs, --no-install-txs**
Whether to install the test execution service (TXS) from the VirtualBox ValidationKit. This is useful when preparing VMs for testing or similar. (default: –no-install-txs)

**--validation-kit-iso=<*testing-iso*>**
Path to the VirtualBox ValidationKit ISO. This is required if `--install-txs` is specified.

**--locale=<*ll_CC*>**
The base locale specification for the guest, like en_US, de_CH, or nn_NO. (default: host or en_US)

**--country=<*CC*>**
The two letter country code if it differs from the specified by `--location`.

**--time-zone=<*tz*>**
The time zone to set up the guest OS with. (default: host time zone or UTC)

**--hostname=<*fqdn*>**
The fully qualified domain name of the guest machine. (default: *vmname*.myguest.virtualbox.org)

**--package-selection-adjustment=<*keyword*>**
Adjustments to the guest OS packages/components selection. This can be specfied more than once. Currently the only recognized keyword is `minimal` which triggers a minimal installation for some of the guest OSes.

**--dry-run**
Do not create any files or make any changes to the VM configuration.

**--start-vm=<*session-type*>**
Start the VM using the front end given by *session-type*. This is the same as the `--type` option for the `startvm` command, but we have add `none` for indicating that the VM should not be started. (default: `none`)

Advanced options:

**--auxiliary-base-path=<*path*>**
The path prefix to the media related files generated for the installation. (default: *vm-config-dir*/Unattended-*vm-uuid*-)

**--image-index=<*number*>**
Windows installation image index. (default: 1)

**--script-template=<*file*>**
The unattended installation script template. (default: IMachine::OSTypeId dependent)

**--post-install-template=<*file*>**
The post installation script template. (default: IMachine::OSTypeId dependent)

**--post-install-command=<*command*>**
A single command to run after the installation is completed. The exact format and exactly when this is run is guest OS installer dependent.

**--extra-install-kernel-parameters=<*params*>**

List of extra linux kernel parameters to use during the installation. (default: IMachine::OSTypeId dependent)

**--language=<*lang*>**

Specifies the UI language for a Windows installation. The *lang* is generally on the form {ll}-{CC}. See detectedOSLanguages results from `VBoxManage unattended detect`. (default: detectedOSLanguages[0])

# 9 Advanced Topics

## 9.1 Automated Guest Logins

Oracle VM VirtualBox provides Guest Addition modules for Windows, Linux, and Oracle Solaris to enable automated logins on the guest.

When a guest operating system is running in a virtual machine, it might be desirable to perform coordinated and automated logins using credentials from a master login system. Credentials are user name, password, and domain name, where each value might be empty.

### 9.1.1 Automated Windows Guest Logins

Since Windows NT, Windows has provided a modular system login subsystem, called Winlogon, which can be customized and extended by means of so-called GINA (Graphical Identification and Authentication) modules. With Windows Vista and Windows 7, the GINA modules were replaced with a new mechanism called credential providers. The Oracle VM VirtualBox Guest Additions for Windows come with both, a GINA and a credential provider module, and therefore enable any Windows guest to perform automated logins.

To activate the Oracle VM VirtualBox GINA or credential provider module, install the Guest Additions using the command line switch `/with_autologon`. All the following manual steps required for installing these modules will be then done by the installer.

To manually install the Oracle VM VirtualBox GINA module, extract the Guest Additions as shown in chapter 4.2.1.4, *Manual File Extraction*, page 66 and copy the file `VBoxGINA.dll` to the Windows `SYSTEM32` directory. Then, in the registry, create the following key with a value of `VBoxGINA.dll`:

```
HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Winlogon\GinaDLL
```

> **Note:** The Oracle VM VirtualBox GINA module is implemented as a wrapper around the standard Windows GINA module, `MSGINA.DLL`. As a result, it may not work correctly with third party GINA modules.

To manually install the Oracle VM VirtualBox credential provider module, extract the Guest Additions as shown in chapter 4.2.1.4, *Manual File Extraction*, page 66 and copy the file `VBoxCredProv.dll` to the Windows `SYSTEM32` directory. In the registry, create the following keys:

```
HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Windows\CurrentVersion\
Authentication\Credential Providers\{275D3BCC-22BB-4948-A7F6-3A3054EBA92B}

HKEY_CLASSES_ROOT\CLSID\{275D3BCC-22BB-4948-A7F6-3A3054EBA92B}

HKEY_CLASSES_ROOT\CLSID\{275D3BCC-22BB-4948-A7F6-3A3054EBA92B}\InprocServer32
```

All default values, the key named `Default`, must be set to `VBoxCredProv`.

Create a new string named as follows, with a value of `Apartment`.

```
HKEY_CLASSES_ROOT\CLSID\{275D3BCC-22BB-4948-A7F6-3A3054EBA92B}\InprocServer32\ThreadingModel
```

To set credentials, use the following command on a *running* VM:

```
VBoxManage controlvm "Windows XP" setcredentials "John Doe" "secretpassword" "DOMTEST"
```

While the VM is running, the credentials can be queried by the Oracle VM VirtualBox login modules, GINA or credential provider, using the Oracle VM VirtualBox Guest Additions device driver. When Windows is in *logged out* mode, the login modules will constantly poll for credentials and if they are present, a login will be attempted. After retrieving the credentials, the login modules will erase them so that the above command will have to be repeated for subsequent logins.

For security reasons, credentials are not stored in any persistent manner and will be lost when the VM is reset. Also, the credentials are write-only. There is no way to retrieve the credentials from the host side. Credentials can be reset from the host side by setting empty values.

Depending on the particular variant of the Windows guest, the following restrictions apply:

- For **Windows XP guests.** The login subsystem needs to be configured to use the classic login dialog, as the Oracle VM VirtualBox GINA module does not support the XP-style welcome dialog.
- **Windows Vista, Windows 7, Windows 8, and Windows 10 guests.** The login subsystem does not support the so-called Secure Attention Sequence, `Ctrl+Alt+Del`. As a result, the guest's group policy settings need to be changed to not use the Secure Attention Sequence. Also, the user name given is only compared to the true user name, not the user friendly name. This means that when you rename a user, you still have to supply the original user name as Windows never renames user accounts internally.
- Automatic login handling of the built-in **Windows Remote Desktop Service**, formerly known as Terminal Services, is disabled by default. To enable it, create the following registry key with a `DWORD` value of `1`.

  ```
  HKEY_LOCAL_MACHINE\SOFTWARE\Oracle\VirtualBox Guest Additions\AutoLogon
  ```

The following command forces Oracle VM VirtualBox to keep the credentials after they were read by the guest and on VM reset:

```
VBoxManage setextradata "Windows XP" VBoxInternal/Devices/VMMDev/0/Config/KeepCredentials 1
```

Note that this is a potential security risk, as a malicious application running on the guest could request this information using the proper interface.

### 9.1.2 Automated Linux and UNIX Guest Logins

Oracle VM VirtualBox provides a custom PAM module (Pluggable Authentication Module) which can be used to perform automated guest logins on platforms which support this framework. Virtually all modern Linux and UNIX distributions rely on PAM.

For automated logins on Ubuntu, or Ubuntu-derived, distributions using LightDM as the display manager. See chapter 9.1.2.1, *Oracle VM VirtualBox Greeter for Ubuntu/LightDM*, page 209.

The `pam_vbox.so` module itself *does not* do an actual verification of the credentials passed to the guest OS. Instead it relies on other modules such as `pam_unix.so` or `pam_unix2.so` down in the PAM stack to do the actual validation using the credentials retrieved by `pam_vbox.so`. Therefore `pam_vbox.so` has to be on top of the authentication PAM service list.

> **Note:** The `pam_vbox.so` module only supports the `auth` primitive. Other primitives such as `account`, `session`, or `password` are not supported.

The pam_vbox.so module is shipped as part of the Guest Additions but it is not installed and/or activated on the guest OS by default. In order to install it, it has to be copied from /opt/VBoxGuestAdditions-<version>/other/ to the security modules directory. This is usually /lib/security/ on 32-bit Linux guests or /lib64/security/ on 64-bit Linux guests. Please refer to your guest OS documentation for the correct PAM module directory.

For example, to use pam_vbox.so with a Ubuntu Linux guest OS and the GNOME Desktop Manager (GDM) to log in users automatically with the credentials passed by the host, configure the guest OS as follows:

1. Copy the pam_vbox.so module to the security modules directory. In this case, /lib/security.
2. Edit the PAM configuration file for GDM, found at /etc/pam.d/gdm. Add the line auth requisite pam_vbox.so at the top. Additionally, in most Linux distributions there is a file called /etc/pam.d/common-auth. This file is included in many other services, like the GDM file mentioned above. There you also have to add the line auth requisite pam_vbox.so.
3. If authentication against the shadow database using pam_unix.so or pam_unix2.so is desired, the argument try_first_pass for pam_unix.so or use_first_pass for pam_unix2.so is needed in order to pass the credentials from the Oracle VM VirtualBox module to the shadow database authentication module. For Ubuntu, this needs to be added to /etc/pam.d/common-auth, to the end of the line referencing pam_unix.so. This argument tells the PAM module to use credentials already present in the stack, such as the ones provided by the Oracle VM VirtualBox PAM module.

**Warning:** An incorrectly configured PAM stack can effectively prevent you from logging into your guest system.

To make deployment easier, you can pass the argument debug right after the pam_vbox.so statement. Debug log output will then be recorded using syslog.

**Note:** By default, pam_vbox will not wait for credentials to arrive from the host. When a login prompt is shown, for example by GDM/KDM or the text console, and pam_vbox does not yet have credentials it does not wait until they arrive. Instead the next module in the PAM stack, depending on the PAM configuration, will have the chance for authentication.

pam_vbox supports various guest property parameters that are located in /VirtualBox/GuestAdd/PAM/. These parameters allow pam_vbox to wait for credentials to be provided by the host and optionally can show a message while waiting for those. The following guest properties can be set:

- CredsWait: Set to 1 if pam_vbox should start waiting until credentials arrive from the host. Until then no other authentication methods such as manually logging in will be available. If this property is empty or gets deleted no waiting for credentials will be performed and pam_vbox will act like before. This property must be set read-only for the guest (RDONLYGUEST).
- CredsWaitAbort: Aborts waiting for credentials when set to any value. Can be set from host and the guest.

- `CredsWaitTimeout`: Timeout, in seconds, to let pam_vbox wait for credentials to arrive. When no credentials arrive within this timeout, authentication of pam_vbox will be set to failed and the next PAM module in chain will be asked. If this property is not specified, set to 0 or an invalid value, an infinite timeout will be used. This property must be set read-only for the guest (`RDONLYGUEST`).

To customize pam_vbox further there are the following guest properties:

- `CredsMsgWaiting`: Custom message showed while pam_vbox is waiting for credentials from the host. This property must be set read-only for the guest (`RDONLYGUEST`).
- `CredsMsgWaitTimeout`: Custom message showed when waiting for credentials by pam_vbox has timed out. For example, they did not arrive within time. This property must be set read-only for the guest (`RDONLYGUEST`).

**Note:** If a pam_vbox guest property does not have the correct flag set (`RDONLYGUEST`) the property is ignored and, depending on the property, a default value will be used. This can result in pam_vbox not waiting for credentials. Consult the appropriate syslog file for more information and use the `debug` option.

#### 9.1.2.1 Oracle VM VirtualBox Greeter for Ubuntu/LightDM

Oracle VM VirtualBox comes with an own greeter module, named vbox-greeter. The module can be used with LightDM 1.0.1 or later. LightDM is the default display manager since Ubuntu 10.11 and therefore also can be used for automated guest logins.

vbox-greeter does not need the `pam_vbox` module described above in order to function. It comes with its own authentication mechanism provided by LightDM. However, to provide maximum of flexibility both modules can be used together on the same guest.

As with the `pam_vbox` module, vbox-greeter is shipped as part of the Guest Additions but it is not installed or activated on the guest OS by default. To install vbox-greeter automatically upon Guest Additions installation, use the `--with-autologon` switch when starting the VBoxLinuxAdditions.run file:

```
# ./VBoxLinuxAdditions.run -- --with-autologon
```

For manual or postponed installation, the `vbox-greeter.desktop` file has to be copied from `/opt/VBoxGuestAdditions-<version>/other/` to the `xgreeters` directory This is usually `/usr/share/xgreeters/`. Please refer to your guest OS documentation for the correct LightDM greeter directory.

The vbox-greeter module itself already was installed by the Oracle VM VirtualBox Guest Additions installer and resides in `/usr/sbin/`. To enable vbox-greeter as the standard greeter module, the file `/etc/lightdm/lightdm.conf` needs to be edited:

```
[SeatDefaults]
greeter-session=vbox-greeter
```

**Note:**

- The LightDM server needs to be fully restarted in order for vbox-greeter to be used as the default greeter. As root, run `service lightdm --full-restart` on Ubuntu, or simply restart the guest.
- vbox-greeter is independent of the graphical session chosen by the user, such as Gnome, KDE, or Unity. However, it requires FLTK 1.3 for representing its own user interface.

There are numerous guest properties which can be used to further customize the login experience. For automatically logging in users, the same guest properties apply as for pam_vbox. See chapter 9.1.2, *Automated Linux and UNIX Guest Logins*, page 207.

In addition to the above mentioned guest properties, vbox-greeter allows further customization of its user interface. These special guest properties all reside in `/VirtualBox/GuestAdd/Greeter/`:

- `HideRestart`: Set to 1 if vbox-greeter should hide the button to restart the guest. This property must be set read-only for the guest (`RDONLYGUEST`).
- `HideShutdown`: Set to 1 if vbox-greeter should hide the button to shutdown the guest. This property must be set read-only for the guest (`RDONLYGUEST`).
- `BannerPath`: Path to a .PNG file for using it as a banner on the top. The image size must be 460 x 90 pixels, any bit depth. This property must be set read-only for the guest (`RDONLYGUEST`).
- `UseTheming`: Set to 1 for turning on the following theming options. This property must be set read-only for the guest (`RDONLYGUEST`).
- `Theme/BackgroundColor`: Hexadecimal RRGGBB color for the background. This property must be set read-only for the guest (`RDONLYGUEST`).
- `Theme/LogonDialog/HeaderColor`: Hexadecimal RRGGBB foreground color for the header text. This property must be set read-only for the guest (`RDONLYGUEST`).
- `Theme/LogonDialog/BackgroundColor`: Hexadecimal RRGGBB color for the login dialog background. This property must be set read-only for the guest (`RDONLYGUEST`).
- `Theme/LogonDialog/ButtonColor`: Hexadecimal RRGGBB background color for the login dialog button. This property must be set read-only for the guest (`RDONLYGUEST`).

**Note:** The same restrictions for the guest properties above apply as for the ones specified in the pam_vbox section.

## 9.2 Advanced Configuration for Windows Guests

### 9.2.1 Automated Windows System Preparation

Beginning with Windows NT 4.0, Microsoft offers a system preparation tool called Sysprep, to prepare a Windows system for deployment or redistribution. Whereas Windows 2000 and XP ship with Sysprep on the installation medium, the tool also is available for download on the Microsoft web site. In a standard installation of Windows Vista and 7, Sysprep is already included. Sysprep mainly consists of an executable called `sysprep.exe` which is invoked by the user to put the Windows installation into preparation mode.

The Guest Additions offer a way to launch a system preparation on the guest operating system in an automated way, controlled from the host system. See chapter 4.9, *Guest Control of Applications*, page 79 for details of how to use this feature with the special identifier `sysprep` as the program to execute, along with the user name `sysprep` and password `sysprep` for the credentials. Sysprep then gets launched with the required system rights.

**Note:** Specifying the location of "sysprep.exe" is **not possible**. Instead the following paths are used, based on the operating system:

- `C:\sysprep\sysprep.exe` for Windows NT 4.0, 2000 and XP
- `%WINDIR%\System32\Sysprep\sysprep.exe` for Windows Vista, 2008 Server and 7

The Guest Additions will automatically use the appropriate path to execute the system preparation tool.

## 9.3 Advanced Configuration for Linux and Oracle Solaris Guests

### 9.3.1 Manual Setup of Selected Guest Services on Linux

The Oracle VM VirtualBox Guest Additions contain several different drivers. If for any reason you do not wish to set them all up, you can install the Guest Additions using the following command:

```
sh ./VBoxLinuxAdditions.run no_setup
```

After this, you will need to at least compile the kernel modules by running the command as root:

```
rcvboxadd setup
```

You will need to replace *lib* by *lib64* on some 64bit guests, and on older guests without the udev service you will need to add the *vboxadd* service to the default runlevel to ensure that the modules get loaded.

To setup the time synchronization service, add the service vboxadd-service to the default runlevel. To set up the X11 and OpenGL part of the Guest Additions, run the following command:

```
rcvboxadd-x11 setup
```

You do not need to enable any services for this.

To recompile the guest kernel modules, use this command:

```
rcvboxadd setup
```

After compilation you should reboot your guest to ensure that the new modules are actually used.

### 9.3.2 Guest Graphics and Mouse Driver Setup in Depth

This section assumes that you are familiar with configuring the X.Org server using xorg.conf and optionally the newer mechanisms using hal or udev and xorg.conf.d. If not you can learn about them by studying the documentation which comes with X.Org.

The Oracle VM VirtualBox Guest Additions include the following drivers for X.Org versions:

- X11R6.8/X11R6.9 and XFree86 version 4.3 (vboxvideo_drv_68.o and vboxmouse_drv_68.o)
- X11R7.0 (vboxvideo_drv_70.so and vboxmouse_drv_70.so)
- X11R7.1 (vboxvideo_drv_71.so and vboxmouse_drv_71.so)

- X.Org Server versions 1.3 and later (vboxvideo_drv_13.so vboxmouse_drv_13.so, and later versions).

By default these drivers can be found in the folowing directory:
`/opt/VBoxGuestAdditions-<version>/other/`
The correct versions for the X server are symbolically linked into the X.Org driver directories.

For graphics integration to work correctly, the X server must load the vboxvideo driver. Many recent X server versions look for it automatically if they see that they are running in Oracle VM VirtualBox. For an optimal user experience the guest kernel drivers must be loaded and the Guest Additions tool VBoxClient must be running as a client in the X session. For mouse integration to work correctly, the guest kernel drivers must be loaded and in addition, in X servers from X.Org X11R6.8 to X11R7.1 and in XFree86 version 4.3 the right vboxmouse driver must be loaded and associated with /dev/mouse or /dev/psaux. In X.Org server 1.3 or later a driver for a PS/2 mouse must be loaded and the right vboxmouse driver must be associated with /dev/vboxguest.

The Oracle VM VirtualBox guest graphics driver can use any graphics configuration for which the virtual resolution fits into the virtual video memory allocated to the virtual machine, minus a small amount used by the guest driver, as described in chapter 3.6, *Display Settings*, page 50. The driver will offer a range of standard modes at least up to the default guest resolution for all active guest monitors. In X.Org Server 1.3 and later the default mode can be changed by setting the output property VBOX_MODE to "<width>x<height>" for any guest monitor. When VBoxClient and the kernel drivers are active this is done automatically when the host requests a mode change. The driver for older versions can only receive new modes by querying the host for requests at regular intervals.

With X Servers before version 1.3, you can also add your own modes to the X server configuration file. You simply need to add them to the "Modes" list in the "Display" subsection of the "Screen" section. For example, the following section has a custom 2048x800 resolution mode added:

```
Section "Screen"
        Identifier    "Default Screen"
        Device        "VirtualBox graphics card"
        Monitor       "Generic Monitor"
        DefaultDepth  24
        SubSection "Display"
                Depth         24
                Modes         "2048x800" "800x600" "640x480"
        EndSubSection
EndSection
```

## 9.4 CPU Hot-Plugging

With virtual machines running modern server operating systems, Oracle VM VirtualBox supports CPU hot-plugging.

On a physical computer CPU hot-plugging would mean that a CPU can be added or removed while the machine is running. Oracle VM VirtualBox supports adding and removing of virtual CPUs while a virtual machine is running.

CPU hot-plugging works only with guest operating systems that support the feature. So far this applies only to Linux and Windows Server. Windows supports only hot-add, while Linux supports hot-add and hot-remove. To use this feature with more than 8 CPUs, a 64-bit Linux guest is required.

CPU hot-plugging is done using the `VBoxManage` command-line interface. First, hot-plugging needs to be enabled for a virtual machine:

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

The `--cpus` option is used to specify the maximum number of CPUs that the virtual machine can have:

```
VBoxManage modifyvm "VM name" --cpus 8
```

When the VM is off, you can then add and remove virtual CPUs with the `modifyvm --plugcpu` and `--unplugcpu` subcommands, which take the number of the virtual CPU as a parameter, as follows:

```
VBoxManage modifyvm "VM name" --plugcpu 3
VBoxManage modifyvm "VM name" --unplugcpu 3
```

Note that CPU 0 can never be removed.

While the VM is running, CPUs can be added and removed with the `controlvm plugcpu` and `unplugcpu` commands instead, as follows:

```
VBoxManage controlvm "VM name" plugcpu 3
VBoxManage controlvm "VM name" unplugcpu 3
```

See chapter 8.8, *VBoxManage modifyvm*, page 135 and chapter 8.14, *VBoxManage controlvm*, page 152 for details.

With Linux guests, the following applies:

To prevent ejection while the CPU is still used it has to be ejected from within the guest before. The Linux Guest Additions contain a service which receives hot-remove events and ejects the CPU. Also, after a CPU is added to the VM it is not automatically used by Linux. The Linux Guest Additions service will take care of that if installed. If not a CPU can be started with the following command:

```
echo 1 > /sys/devices/system/cpu/cpu<id>/online
```

## 9.5 PCI Passthrough

When running on Linux hosts with a kernel version later than `2.6.31`, experimental host PCI devices passthrough is available.

> **Note:** The PCI passthrough module is shipped as a Oracle VM VirtualBox extension package, which must be installed separately. See chapter 1.6, *Installing Oracle VM VirtualBox and Extension Packs*, page 6.

This feature enables a guest to directly use physical PCI devices on the host, even if host does not have drivers for this particular device. Both, regular PCI and some PCI Express cards, are supported. AGP and certain PCI Express cards are not supported at the moment if they rely on Graphics Address Remapping Table (GART) unit programming for texture management as it does rather non-trivial operations with pages remapping interfering with IOMMU. This limitation may be lifted in future releases.

To be fully functional, PCI passthrough support in Oracle VM VirtualBox depends upon an IOMMU hardware unit which is not yet too widely available. If the device uses bus mastering, for example it performs DMA to the OS memory on its own, then an IOMMU is required. Otherwise such DMA transactions may write to the wrong physical memory address as the device DMA engine is programmed using a device-specific protocol to perform memory transactions. The IOMMU functions as translation unit mapping physical memory access requests from the device using knowledge of the guest physical address to host physical addresses translation rules.

Intel's solution for IOMMU is called Intel Virtualization Technology for Directed I/O (VT-d), and AMD's solution is called AMD-Vi. Check your motherboard datasheet for the appropriate technology. Even if your hardware does not have a IOMMU, certain PCI cards may work, such as serial PCI adapters, but the guest will show a warning on boot and the VM execution will terminate if the guest driver will attempt to enable card bus mastering.

It is very common that the BIOS or the host OS disables the IOMMU by default. So before any attempt to use it please make sure that the following apply:

- Your motherboard has an IOMMU unit.
- Your CPU supports the IOMMU.
- The IOMMU is enabled in the BIOS.
- The VM must run with VT-x/AMD-V and nested paging enabled.
- Your Linux kernel was compiled with IOMMU support, including DMA remapping. See the `CONFIG_DMAR` kernel compilation option. The PCI stub driver (`CONFIG_PCI_STUB`) is required as well.
- Your Linux kernel recognizes and uses the IOMMU unit. The `intel_iommu=on` boot option could be needed. Search for DMAR and PCI-DMA in kernel boot log.

Once you made sure that the host kernel supports the IOMMU, the next step is to select the PCI card and attach it to the guest. To figure out the list of available PCI devices, use the `lspci` command. The output will look as follows:

```
01:00.0 VGA compatible controller: ATI Technologies Inc Cedar PRO [Radeon HD 5450]
01:00.1 Audio device: ATI Technologies Inc Manhattan HDMI Audio [Mobility Radeon HD 5000 Series]
02:00.0 Ethernet controller: Realtek Semiconductor Co., Ltd. RTL8111/8168B PCI Express Gigabit
        Ethernet controller (rev 03)
03:00.0 SATA controller: JMicron Technology Corp. JMB362/JMB363 Serial ATA Controller (rev 03)
03:00.1 IDE interface: JMicron Technology Corp. JMB362/JMB363 Serial ATA Controller (rev 03)
06:00.0 VGA compatible controller: nVidia Corporation G86 [GeForce 8500 GT] (rev a1)
```

The first column is a PCI address, in the format `bus:device.function`. This address could be used to identify the device for further operations. For example, to attach a PCI network controller on the system listed above to the second PCI bus in the guest, as device 5, function 0, use the following command:

```
VBoxManage modifyvm "VM name" --pciattach 02:00.0@01:05.0
```

To detach the same device, use:

```
VBoxManage modifyvm "VM name" --pcidetach 02:00.0
```

Please note that both host and guest could freely assign a different PCI address to the card attached during runtime, so those addresses only apply to the address of the card at the moment of attachment on the host, and during BIOS PCI init on the guest.

If the virtual machine has a PCI device attached, certain limitations apply:

- Only PCI cards with non-shared interrupts, such as those using MSI on the host, are supported at the moment.
- No guest state can be reliably saved or restored. The internal state of the PCI card cannot be retrieved.
- Teleportation, also called live migration, does not work. The internal state of the PCI card cannot be retrieved.
- No lazy physical memory allocation. The host will preallocate the whole RAM required for the VM on startup, as we cannot catch physical hardware accesses to the physical memory.

## 9.6 Webcam Passthrough

### 9.6.1 Using a Host Webcam in the Guest

Oracle VM VirtualBox 4.3 includes an experimental feature which enables a guest to use a host webcam. This complements the general USB passthrough support which was the typical way of using host webcams in earlier versions. The webcam passthrough support can handle non-USB video sources in theory, but this is completely untested.

> **Note:** The webcam passthrough module is shipped as part of the Oracle VM VirtualBox extension pack, which must be installed separately. See chapter 1.6, *Installing Oracle VM VirtualBox and Extension Packs*, page 6.

The host webcam can be attached to the VM using the **Devices** menu in the VM menu bar. The **Webcams** menu contains a list of available video input devices on the host. Clicking on a webcam name attaches or detaches the corresponding host device.

The `VBoxManage` command line tool can be used to enable webcam passthrough. Please see the host-specific sections below for additional details. The following commands are available:

- Get a list of host webcams, or other video input devices:

  ```
  VBoxManage list webcams
  ```

  The output format is as follows:

  ```
  alias "user friendly name"
  host path or identifier
  ```

  The alias can be used as a shortcut in other commands. Alias '.0' means the default video input device on the host. Alias '.1', '.2'means first, second video input device, and so on. The device order is host-specific.

- Attach a webcam to a running VM, as follows:

  ```
  VBoxManage controlvm "VM name" webcam attach [host_path|alias [settings]]
  ```

  This attaches a USB webcam device to the guest.

  The `settings` parameter is a string `Setting1=Value1;Setting2=Value2`, which enables you to configure the emulated webcam device. The following settings are supported:

  - `MaxFramerate`: The highest rate at which video frames are sent to the guest. A higher frame rate requires more CPU power. Therefore sometimes it is useful to set a lower limit. Default is no limit and allow the guest to use all frame rates supported by the host webcam.
  - `MaxPayloadTransferSize`: How many bytes the emulated webcam can send to the guest at a time. Default value is 3060 bytes, which is used by some webcams. Higher values can slightly reduce CPU load, if the guest is able to use larger buffers. However, a high `MaxPayloadTransferSize` might be not supported by some guests.

- Detach a webcam from a running VM, as follows:

  ```
  VBoxManage controlvm "VM name" webcam detach [host_path|alias]
  ```

- List the webcams attached to a running VM, as follows:

  ```
  VBoxManage controlvm "VM name" webcam list
  ```

  The output contains the path or alias which was used in the `webcam attach` command for each attached webcam.

### 9.6.2 Windows Hosts

When the webcam device is detached from the host, the emulated webcam device is automatically detached from the guest.

### 9.6.3 Mac OS X Hosts

OS X version 10.9 or later is required.

When the webcam device is detached from the host, the emulated webcam device remains attached to the guest and must be manually detached using the `VBoxManage controlvm "VM name" webcam detach` command.

### 9.6.4 Linux and Oracle Solaris Hosts

When the webcam is detached from the host the emulated webcam device is automatically detached from the guest only if the webcam is streaming video. If the emulated webcam is inactive it should be manually detached using the `VBoxManage controlvm "VM name" webcam detach` command.

Aliases `.0` and `.1` are mapped to `/dev/video0`, alias `.2` is mapped to `/dev/video1` and so forth.

## 9.7 Advanced Display Configuration

### 9.7.1 Custom VESA Resolutions

Apart from the standard VESA resolutions, the Oracle VM VirtualBox VESA BIOS enables you to add up to 16 custom video modes which will be reported to the guest operating system. When using Windows guests with the Oracle VM VirtualBox Guest Additions, a custom graphics driver will be used instead of the fallback VESA solution so this information does not apply.

Additional video modes can be configured for each VM using the extra data facility. The extra data key is called `CustomVideoMode<x>` with `x` being a number from 1 to 16. Please note that modes will be read from 1 until either the following number is not defined or 16 is reached. The following example adds a video mode that corresponds to the native display resolution of many notebook computers:

```
VBoxManage setextradata "VM name" "CustomVideoMode1" "1400x1050x16"
```

The VESA mode IDs for custom video modes start at `0x160`. In order to use the above defined custom video mode, the following command line has to be supplied to Linux:

```
vga = 0x200 | 0x160
vga = 864
```

For guest operating systems with Oracle VM VirtualBox Guest Additions, a custom video mode can be set using the video mode hint feature.

### 9.7.2 Configuring the Maximum Resolution of Guests When Using the Graphical Frontend

When guest systems with the Guest Additions installed are started using the graphical frontend, the normal Oracle VM VirtualBox application, they will not be allowed to use screen resolutions greater than the host's screen size unless the user manually resizes them by dragging the window, switching to full screen or seamless mode or sending a video mode hint using `VBoxManage`. This behavior is what most users will want, but if you have different needs, it is possible to change it by issuing one of the following commands from the command line:

```
VBoxManage setextradata global GUI/MaxGuestResolution any
```

will remove all limits on guest resolutions.

```
VBoxManage setextradata global GUI/MaxGuestResolution >width,height<
```

manually specifies a maximum resolution.

```
VBoxManage setextradata global GUI/MaxGuestResolution auto
```

restores the default settings. Note that these settings apply globally to all guest systems, not just to a single machine.

## 9.8 Advanced Storage Configuration

### 9.8.1 Using a Raw Host Hard Disk From a Guest

As an alternative to using virtual disk images as described in chapter 5, *Virtual Storage*, page 83, Oracle VM VirtualBox can also present either entire physical hard disks or selected partitions as virtual disks to virtual machines.

With Oracle VM VirtualBox, this type of access is called *raw hard disk access*. It enables a guest operating system to access its virtual hard disk without going through the host OS file system. The actual performance difference for image files vs. raw disk varies greatly depending on the overhead of the host file system, whether dynamically growing images are used, and on host OS caching strategies. The caching indirectly also affects other aspects such as failure behavior. For example, whether the virtual disk contains all data written before a host OS crash. Consult your host OS documentation for details on this.

> **Warning:** Raw hard disk access is for expert users only. Incorrect use or use of an outdated configuration can lead to **total loss of data** on the physical disk. Most importantly, *do not* attempt to boot the partition with the currently running host operating system in a guest. This will lead to severe data corruption.

Raw hard disk access, both for entire disks and individual partitions, is implemented as part of the VMDK image format support. As a result, you will need to create a special VMDK image file which defines where the data will be stored. After creating such a special VMDK image, you can use it like a regular virtual disk image. For example, you can use the VirtualBox Manager, see chapter 5.3, *The Virtual Media Manager*, page 87, or `VBoxManage` to assign the image to a virtual machine.

#### 9.8.1.1 Access to Entire Physical Hard Disk

While this variant is the simplest to set up, you must be aware that this will give a guest operating system direct and full access to an *entire physical disk*. If your *host* operating system is also booted from this disk, please take special care to not access the partition from the guest at all. On the positive side, the physical disk can be repartitioned in arbitrary ways without having to recreate the image file that gives access to the raw disk.

On a Linux host, to create an image that represents an entire physical hard disk which will not contain any actual data, as this will all be stored on the physical disk, use the following command:

```
VBoxManage internalcommands createrawvmdk -filename /path/to/file.vmdk
  -rawdisk /dev/sda
```

This creates the image `/path/to/file.vmdk`, which must be an absolute path. All data will be read and written from `/dev/sda`.

On a Windows host, instead of the above device specification, for example use `\\.\PhysicalDrive0`. On a Mac OS X host, instead of the above device specification use for example `/dev/disk1`. Note that on OS X you can only get access to an entire disk if no volume is mounted from it.

Creating the image requires read/write access for the given device. Read/write access is also later needed when using the image from a virtual machine. On some host platforms, such as Windows Vista and later, raw disk access may be restricted and not permitted by the host OS in some situations.

Just like with regular disk images, this does not automatically attach the newly created image to a virtual machine. This can be done as follows:

```
VBoxManage storageattach WindowsXP --storagectl "IDE Controller"
 --port 0 --device 0 --type hdd --medium /path/to/file.vmdk
```

When this is done the selected virtual machine will boot from the specified physical disk.

#### 9.8.1.2 Access to Individual Physical Hard Disk Partitions

This *raw partition support* is quite similar to the full hard disk access described above. However, in this case, any partitioning information will be stored inside the VMDK image. This means that you can install a different boot loader in the virtual hard disk without affecting the host's partitioning information. While the guest will be able to *see* all partitions that exist on the physical disk, access will be filtered in that reading from partitions for which no access is allowed the partitions will only yield zeroes, and all writes to them are ignored.

To create a special image for raw partition support, which will contain a small amount of data, on a Linux host, use the command:

```
VBoxManage internalcommands createrawvmdk -filename /path/to/file.vmdk
 -rawdisk /dev/sda -partitions 1,5
```

The command is identical to the one for full hard disk access, except for the additional `-partitions` parameter. This example would create the image `/path/to/file.vmdk`, which must be absolute, and partitions 1 and 5 of `/dev/sda` would be made accessible to the guest.

Oracle VM VirtualBox uses the same partition numbering as your Linux host. As a result, the numbers given in the above example would refer to the first primary partition and the first logical drive in the extended partition, respectively.

On a Windows host, instead of the above device specification, use for example `\\.\PhysicalDrive0`. On a Mac OS X host, instead of the above device specification use `/dev/disk1`, for example. Note that on OS X you can only use partitions which are not mounted. Eject the respective volume first. Partition numbers are the same on Linux, Windows, and Mac OS X hosts.

The numbers for the list of partitions can be taken from the output of the following command:

```
VBoxManage internalcommands listpartitions -rawdisk /dev/sda
```

The output lists the partition types and sizes to give the user enough information to identify the partitions necessary for the guest.

Images which give access to individual partitions are specific to a particular host disk setup. You cannot transfer these images to another host. Also, whenever the host partitioning changes, the image *must be recreated*.

Creating the image requires read/write access for the given device. Read/write access is also later needed when using the image from a virtual machine. If this is not feasible, there is a special variant for raw partition access, currently only available on Linux hosts, that avoids having to give the current user access to the entire disk. To set up such an image, use:

```
VBoxManage internalcommands createrawvmdk -filename /path/to/file.vmdk
 -rawdisk /dev/sda -partitions 1,5 -relative
```

When used from a virtual machine, the image will then refer not to the entire disk, but only to the individual partitions. In this example, `/dev/sda1` and `/dev/sda5`. As a consequence, read/write access is only required for the affected partitions, not for the entire disk. During creation however, read-only access to the entire disk is required to obtain the partitioning information.

In some configurations it may be necessary to change the MBR code of the created image. For example, to replace the Linux boot loader that is used on the host by another boot loader. This enables for example the guest to boot directly to Windows, while the host boots Linux from the "same" disk. For this purpose the `-mbr` parameter is provided. It specifies a file name from which to take the MBR code. The partition table is not modified at all, so a MBR file from a system with totally different partitioning can be used. An example of this is:

```
VBoxManage internalcommands createrawvmdk -filename /path/to/file.vmdk
 -rawdisk /dev/sda -partitions 1,5 -mbr winxp.mbr
```

The modified MBR will be stored inside the image, not on the host disk.

The created image can be attached to a storage controller in a VM configuration as usual.

### 9.8.2 Configuring the Hard Disk Vendor Product Data (VPD)

Oracle VM VirtualBox reports vendor product data for its virtual hard disks which consist of hard disk serial number, firmware revision and model number. These can be changed using the following commands:

```
VBoxManage setextradata "VM name"
      "VBoxInternal/Devices/ahci/0/Config/Port0/SerialNumber" "serial"
VBoxManage setextradata "VM name"
      "VBoxInternal/Devices/ahci/0/Config/Port0/FirmwareRevision" "firmware"
VBoxManage setextradata "VM name"
      "VBoxInternal/Devices/ahci/0/Config/Port0/ModelNumber" "model"
```

The serial number is a 20 byte alphanumeric string, the firmware revision an 8 byte alphanumeric string and the model number a 40 byte alphanumeric string. Instead of Port0, referring to the first port, specify the desired SATA hard disk port.

The above commands apply to virtual machines with an AHCI (SATA) controller. The commands for virtual machines with an IDE controller are:

```
VBoxManage setextradata "VM name"
      "VBoxInternal/Devices/piix3ide/0/Config/PrimaryMaster/SerialNumber" "serial"
VBoxManage setextradata "VM name"
      "VBoxInternal/Devices/piix3ide/0/Config/PrimaryMaster/FirmwareRevision" "firmware"
VBoxManage setextradata "VM name"
      "VBoxInternal/Devices/piix3ide/0/Config/PrimaryMaster/ModelNumber" "model"
```

For hard disks it is also possible to mark the drive as having a non-rotational medium with:

```
VBoxManage setextradata "VM name"
      "VBoxInternal/Devices/ahci/0/Config/Port0/NonRotational" "1"
```

Additional three parameters are needed for CD/DVD drives to report the vendor product data:

```
VBoxManage setextradata "VM name"
      "VBoxInternal/Devices/ahci/0/Config/Port0/ATAPIVendorId" "vendor"
VBoxManage setextradata "VM name"
      "VBoxInternal/Devices/ahci/0/Config/Port0/ATAPIProductId" "product"
VBoxManage setextradata "VM name"
      "VBoxInternal/Devices/ahci/0/Config/Port0/ATAPIRevision" "revision"
```

The vendor id is an 8 byte alphanumeric string, the product id an 16 byte alphanumeric string and the revision a 4 byte alphanumeric string. Instead of Port0, referring to the first port, specify the desired SATA hard disk port.

### 9.8.3 Access iSCSI Targets Using Internal Networking

As an experimental feature, Oracle VM VirtualBox enables access to an iSCSI target running in a virtual machine which is configured to use Internal Networking mode. See chapter 5.10, *iSCSI Servers*, page 95, chapter 6.6, *Internal Networking*, page 104, and chapter 8.19, *VBoxManage storageattach*, page 160.

The IP stack accessing Internal Networking must be configured in the virtual machine which accesses the iSCSI target. A free static IP and a MAC address not used by other virtual machines must be chosen. In the example below, adapt the name of the virtual machine, the MAC address, the IP configuration, and the Internal Networking name (MyIntNet) according to your needs. The following eight commands must first be issued:

```
VBoxManage setextradata "VM name" VBoxInternal/Devices/IntNetIP/0/Trusted 1
VBoxManage setextradata "VM name" VBoxInternal/Devices/IntNetIP/0/Config/MAC 08:00:27:01:02:0f
VBoxManage setextradata "VM name" VBoxInternal/Devices/IntNetIP/0/Config/IP 10.0.9.1
VBoxManage setextradata "VM name" VBoxInternal/Devices/IntNetIP/0/Config/Netmask 255.255.255.0
VBoxManage setextradata "VM name" VBoxInternal/Devices/IntNetIP/0/LUN#0/Driver IntNet
VBoxManage setextradata "VM name" VBoxInternal/Devices/IntNetIP/0/LUN#0/Config/Network MyIntNet
VBoxManage setextradata "VM name" VBoxInternal/Devices/IntNetIP/0/LUN#0/Config/TrunkType 2
VBoxManage setextradata "VM name" VBoxInternal/Devices/IntNetIP/0/LUN#0/Config/IsService 1
```

Finally the iSCSI disk must be attached with the `--intnet` option to tell the iSCSI initiator to use internal networking, as follows:

```
VBoxManage storageattach ... --medium iscsi
--server 10.0.9.30 --target iqn.2008-12.com.sun:sampletarget --intnet
```

Compared to a regular iSCSI setup, the IP address of the target *must* be specified as a numeric IP address, as there is no DNS resolver for internal networking.

The virtual machine with the iSCSI target should be started before the VM using it is powered on. If a virtual machine using an iSCSI disk is started without having the iSCSI target powered up, it can take up to 200 seconds to detect this situation. The VM will fail to power up.

## 9.9 Legacy Commands for Using Serial Ports

In legacy releases, Oracle VM VirtualBox provided support for virtual serial ports. This was rather complicated to set up, requiring a sequence of `VBoxManage setextradata` statements. That method of setting up serial ports is no longer necessary and *deprecated*. To set up virtual serial ports, use the methods described in chapter 3.10, *Serial Ports*, page 54.

> **Note:** For backwards compatibility, the legacy `setextradata` statements, whose description is retained below from the old version of the manual, take *precedence* over the new way of configuring serial ports. As a result, if configuring serial ports the new way does not work, make sure the VM in question does not have old configuration data such as below still active.

The legacy sequence of configuring a serial port used the following commands:

```
VBoxManage setextradata "VM name"
      "VBoxInternal/Devices/serial/0/Config/IRQ" 4
VBoxManage setextradata "VM name"
      "VBoxInternal/Devices/serial/0/Config/IOBase" 0x3f8
VBoxManage setextradata "VM name"
      "VBoxInternal/Devices/serial/0/LUN#0/Driver" Char
VBoxManage setextradata "VM name"
      "VBoxInternal/Devices/serial/0/LUN#0/AttachedDriver/Driver" NamedPipe
VBoxManage setextradata "VM name"
      "VBoxInternal/Devices/serial/0/LUN#0/AttachedDriver/Config/Location" "\\.\pipe\vboxCOM1"
VBoxManage setextradata "VM name"
      "VBoxInternal/Devices/serial/0/LUN#0/AttachedDriver/Config/IsServer" 1
```

This sets up a serial port in the guest with the default settings for COM1 (IRQ 4, I/O address 0x3f8) and the `Location` setting assumes that this configuration is used on a Windows host, because the Windows named pipe syntax is used. Keep in mind that on Windows hosts a named pipe must always start with `\\.\pipe\`. On Linux the same configuration settings apply, except that the path name for the `Location` can be chosen more freely. Local domain sockets can be placed anywhere, provided the user running Oracle VM VirtualBox has the permission to create a new file in the directory. The final command above defines that Oracle VM VirtualBox acts as a server. It creates the named pipe itself instead of connecting to an already existing one.

## 9.10 Fine Tuning the Oracle VM VirtualBox NAT Engine

### 9.10.1 Configuring the Address of a NAT Network Interface

In NAT mode, the guest network interface is assigned to the IPv4 range `10.0.x.0/24` by default where `x` corresponds to the instance of the NAT interface +2. So `x` is 2 when there is only one NAT instance active. In that case the guest is assigned to the address `10.0.2.15`, the gateway is set to `10.0.2.2` and the name server can be found at `10.0.2.3`.

If the NAT network needs to be changed, use the following command:

```
VBoxManage modifyvm "VM name" --natnet1 "192.168/16"
```

This command would reserve the network addresses from `192.168.0.0` to `192.168.254.254` for the first NAT network instance of "VM name". The guest IP would be assigned to `192.168.0.15` and the default gateway could be found at `192.168.0.2`.

### 9.10.2 Configuring the Boot Server (Next Server) of a NAT Network Interface

For network booting in NAT mode, by default Oracle VM VirtualBox uses a built-in TFTP server at the IP address 10.0.2.4. This default behavior should work fine for typical remote-booting scenarios. However, it is possible to change the boot server IP and the location of the boot image with the following commands:

```
VBoxManage modifyvm "VM name" --nattftpserver1 10.0.2.2
VBoxManage modifyvm "VM name" --nattftpfile1 /srv/tftp/boot/MyPXEBoot.pxe
```

### 9.10.3 Tuning TCP/IP Buffers for NAT

The Oracle VM VirtualBox NAT stack performance is often determined by its interaction with the host's TCP/IP stack and the size of several buffers, `SO_RCVBUF` and `SO_SNDBUF`. For certain setups users might want to adjust the buffer size for a better performance. This can by achieved using the following commands, where values are in kilobytes and can range from 8 to 1024:

```
VBoxManage modifyvm "VM name" --natsettings1 16000,128,128,0,0
```

This example illustrates tuning the NAT settings. The first parameter is the MTU, then the size of the socket's send buffer and the size of the socket's receive buffer, the initial size of the TCP send window, and lastly the initial size of the TCP receive window. Note that specifying zero means fallback to the default value.

Each of these buffers has a default size of 64KB and default MTU is 1500.

### 9.10.4 Binding NAT Sockets to a Specific Interface

By default, Oracle VM VirtualBox's NAT engine will route TCP/IP packets through the default interface assigned by the host's TCP/IP stack. The technical reason for this is that the NAT engine uses sockets for communication. If you want to change this behavior, you can tell the NAT engine to bind to a particular IP address instead. For example, use the following command:

```
VBoxManage modifyvm "VM name" --natbindip1 "10.45.0.2"
```

After this, all outgoing traffic will be sent through the interface with the IP address 10.45.0.2. Ensure that this interface is up and running before changing the NAT bind address.

### 9.10.5 Enabling DNS Proxy in NAT Mode

The NAT engine by default offers the same DNS servers to the guest that are configured on the host. In some scenarios, it can be desirable to hide the DNS server IPs from the guest, for example when this information can change on the host due to expiring DHCP leases. In this case, you can tell the NAT engine to act as DNS proxy using the following command:

```
VBoxManage modifyvm "VM name" --natdnsproxy1 on
```

### 9.10.6 Using the Host's Resolver as a DNS Proxy in NAT Mode

For resolving network names, the DHCP server of the NAT engine offers a list of registered DNS servers of the host. If for some reason you need to hide this DNS server list and use the host's resolver settings, thereby forcing the Oracle VM VirtualBox NAT engine to intercept DNS requests and forward them to host's resolver, use the following command:

```
VBoxManage modifyvm "VM name" --natdnshostresolver1 on
```

Note that this setting is similar to the DNS proxy mode, however whereas the proxy mode just forwards DNS requests to the appropriate servers, the resolver mode will interpret the DNS requests and use the host's DNS API to query the information and return it to the guest.

#### 9.10.6.1 User-Defined Host Name Resolving

In some cases it might be useful to intercept the name resolving mechanism, providing a user-defined IP address on a particular DNS request. The intercepting mechanism enables the user to map not only a single host but domains and even more complex naming conventions if required.

The following command sets a rule for mapping a name to a specified IP:

```
VBoxManage setextradata "VM name" \
      "VBoxInternal/Devices/{pcnet,e1000}/0/LUN#0/AttachedDriver/Config/HostResolverMappings/ \
      <unique rule name of interception rule>/HostIP" <IPv4>
VBoxManage setextradata "VM name" \
      "VBoxInternal/Devices/{pcnet,e1000}/0/LUN#0/AttachedDriver/Config/HostResolverMappings/ \
      <unique rule name>/HostName" <name of host>
```

The following command sets a rule for mapping a pattern name to a specified IP:

```
VBoxManage setextradata "VM name" \
      "VBoxInternal/Devices/{pcnet,e1000}/0/LUN#0/AttachedDriver/Config/HostResolverMappings/ \
      <unique rule name>/HostIP" <IPv4>
VBoxManage setextradata "VM name" \
      "VBoxInternal/Devices/{pcnet,e1000}/0/LUN#0/AttachedDriver/Config/HostResolverMappings/ \
      <unique rule name>/HostNamePattern" <hostpattern>
```

The host pattern may include "|", "?" and "*".

This example demonstrates how to instruct the host-resolver mechanism to resolve all domain and probably some mirrors of www.blocked-site.info site with IP 127.0.0.1:

```
VBoxManage setextradata "VM name" \
      "VBoxInternal/Devices/e1000/0/LUN#0/AttachedDriver/Config/HostResolverMappings/ \
      all_blocked_site/HostIP" 127.0.0.1
VBoxManage setextradata "VM name" \
      "VBoxInternal/Devices/e1000/0/LUN#0/AttachedDriver/Config/HostResolverMappings/ \
      all_blocked_site/HostNamePattern" "*.blocked-site.*|*.fb.org"
```

The host resolver mechanism should be enabled to use user-defined mapping rules, otherwise they do not have any effect.

### 9.10.7 Configuring Aliasing of the NAT Engine

By default, the NAT core uses aliasing and uses random ports when generating an alias for a connection. This works well for the most protocols like SSH, FTP and so on. Though some protocols might need a more transparent behavior or may depend on the real port number the packet was sent from. It is possible to change the NAT mode using the `VBoxManage` frontend with the following commands:

```
VBoxManage modifyvm "VM name" --nataliasmode1 proxyonly

VBoxManage modifyvm "Linux Guest" --nataliasmode1 sameports
```

The first example disables aliasing and switches NAT into transparent mode, the second example enforces preserving of port values. These modes can be combined if necessary.

## 9.11 Configuring the BIOS DMI Information

The DMI data that Oracle VM VirtualBox provides to guests can be changed for a specific VM. Use the following commands to configure the DMI BIOS information. In case your VM is configured to use EFI firmware you need to replace `pcbios` by `efi` in the keys.

- DMI BIOS information (type 0)

```
VBoxManage setextradata "VM name"
      "VBoxInternal/Devices/pcbios/0/Config/DmiBIOSVendor"        "BIOS Vendor"
VBoxManage setextradata "VM name"
      "VBoxInternal/Devices/pcbios/0/Config/DmiBIOSVersion"       "BIOS Version"
VBoxManage setextradata "VM name"
      "VBoxInternal/Devices/pcbios/0/Config/DmiBIOSReleaseDate"   "BIOS Release Date"
VBoxManage setextradata "VM name"
      "VBoxInternal/Devices/pcbios/0/Config/DmiBIOSReleaseMajor"  1
VBoxManage setextradata "VM name"
      "VBoxInternal/Devices/pcbios/0/Config/DmiBIOSReleaseMinor"  2
VBoxManage setextradata "VM name"
      "VBoxInternal/Devices/pcbios/0/Config/DmiBIOSFirmwareMajor" 3
VBoxManage setextradata "VM name"
      "VBoxInternal/Devices/pcbios/0/Config/DmiBIOSFirmwareMinor" 4
```

- DMI system information (type 1)

```
VBoxManage setextradata "VM name"
      "VBoxInternal/Devices/pcbios/0/Config/DmiSystemVendor"      "System Vendor"
VBoxManage setextradata "VM name"
      "VBoxInternal/Devices/pcbios/0/Config/DmiSystemProduct"     "System Product"
VBoxManage setextradata "VM name"
      "VBoxInternal/Devices/pcbios/0/Config/DmiSystemVersion"     "System Version"
VBoxManage setextradata "VM name"
      "VBoxInternal/Devices/pcbios/0/Config/DmiSystemSerial"      "System Serial"
VBoxManage setextradata "VM name"
      "VBoxInternal/Devices/pcbios/0/Config/DmiSystemSKU"         "System SKU"
```

```
VBoxManage setextradata "VM name"
      "VBoxInternal/Devices/pcbios/0/Config/DmiSystemFamily"      "System Family"
VBoxManage setextradata "VM name"
      "VBoxInternal/Devices/pcbios/0/Config/DmiSystemUuid"
       "9852bf98-b83c-49db-a8de-182c42c7226b"
```

- DMI board information (type 2)

```
VBoxManage setextradata "VM name"
      "VBoxInternal/Devices/pcbios/0/Config/DmiBoardVendor"       "Board Vendor"
VBoxManage setextradata "VM name"
      "VBoxInternal/Devices/pcbios/0/Config/DmiBoardProduct"      "Board Product"
VBoxManage setextradata "VM name"
      "VBoxInternal/Devices/pcbios/0/Config/DmiBoardVersion"      "Board Version"
VBoxManage setextradata "VM name"
      "VBoxInternal/Devices/pcbios/0/Config/DmiBoardSerial"       "Board Serial"
VBoxManage setextradata "VM name"
      "VBoxInternal/Devices/pcbios/0/Config/DmiBoardAssetTag"     "Board Tag"
VBoxManage setextradata "VM name"
      "VBoxInternal/Devices/pcbios/0/Config/DmiBoardLocInChass"   "Board Location"
VBoxManage setextradata "VM name"
      "VBoxInternal/Devices/pcbios/0/Config/DmiBoardBoardType"    10
```

- DMI system enclosure or chassis (type 3)

```
VBoxManage setextradata "VM name"
      "VBoxInternal/Devices/pcbios/0/Config/DmiChassisVendor"     "Chassis Vendor"
VBoxManage setextradata "VM name"
      "VBoxInternal/Devices/pcbios/0/Config/DmiChassisType"       3
VBoxManage setextradata "VM name"
      "VBoxInternal/Devices/pcbios/0/Config/DmiChassisVersion"    "Chassis Version"
VBoxManage setextradata "VM name"
      "VBoxInternal/Devices/pcbios/0/Config/DmiChassisSerial"     "Chassis Serial"
VBoxManage setextradata "VM name"
      "VBoxInternal/Devices/pcbios/0/Config/DmiChassisAssetTag"   "Chassis Tag"
```

- DMI processor information (type 4)

```
VBoxManage setextradata "VM name"
      "VBoxInternal/Devices/pcbios/0/Config/DmiProcManufacturer"  "GenuineIntel"
VBoxManage setextradata "VM name"
      "VBoxInternal/Devices/pcbios/0/Config/DmiProcVersion"       "Pentium(R) III"
```

- DMI OEM strings (type 11)

```
VBoxManage setextradata "VM name"
      "VBoxInternal/Devices/pcbios/0/Config/DmiOEMVBoxVer"        "vboxVer_1.2.3"
VBoxManage setextradata "VM name"
      "VBoxInternal/Devices/pcbios/0/Config/DmiOEMVBoxRev"        "vboxRev_12345"
```

If a DMI string is not set, the default value of Oracle VM VirtualBox is used. To set an empty string use `"<EMPTY>"`.

Note that in the above list, all quoted parameters (DmiBIOSVendor, DmiBIOSVersion but not DmiBIOSReleaseMajor) are expected to be strings. If such a string is a valid number, the parameter is treated as number and the VM will most probably refuse to start with an `VERR_CFGM_NOT_STRING` error. In that case, use `"string:<value>"`. For example:

```
VBoxManage setextradata "VM name"
      "VBoxInternal/Devices/pcbios/0/Config/DmiSystemSerial"      "string:1234"
```

Changing this information can be necessary to provide the DMI information of the host to the guest to prevent Windows from asking for a new product key. On Linux hosts, the DMI BIOS information can be obtained with the following command:

```
dmidecode -t0
```

The DMI system information can be obtained as follows:

```
dmidecode -t1
```

## 9.12 Configuring Custom ACPI Tables

Oracle VM VirtualBox can be configured to present up to four custom ACPI tables to the guest. A command such as the following can be used to configure custom ACPI tables. Note that CustomTable1, CustomTable2, and CustomTable3 are available in addition to CustomTable0.

```
VBoxManage setextradata "VM name"
"VBoxInternal/Devices/acpi/0/Config/CustomTable0" "/path/to/table.bin"
```

Configuring custom ACPI tables can for example avoid the need for asking for a new product key on Windows Vista, Windows 7, Windows 8 and later guests. On Linux hosts, one of the system's ACPI tables can be read from `/sys/firmware/acpi/tables/`.

## 9.13 Fine Tuning Timers and Time Synchronization

### 9.13.1 Configuring the Guest Time Stamp Counter (TSC) to Reflect Guest Execution

By default, Oracle VM VirtualBox keeps all sources of time visible to the guest synchronized to a single time source, the monotonic host time. This reflects the assumptions of many guest operating systems, which expect all time sources to reflect "wall clock" time. In special circumstances it may be useful however to make the time stamp counter (TSC) in the guest reflect the time actually spent executing the guest.

This special TSC handling mode can be enabled on a per-VM basis, and for best results must be used only in combination with hardware virtualization. To enable this mode use the following command:

```
VBoxManage setextradata "VM name" "VBoxInternal/TM/TSCTiedToExecution" 1
```

To revert to the default TSC handling mode use:

```
VBoxManage setextradata "VM name" "VBoxInternal/TM/TSCTiedToExecution"
```

Note that if you use the special TSC handling mode with a guest operating system which is very strict about the consistency of time sources you may get a warning or error message about the timing inconsistency. It may also cause clocks to become unreliable with some guest operating systems depending on how they use the TSC.

### 9.13.2 Accelerate or Slow Down the Guest Clock

For certain purposes it can be useful to accelerate or to slow down the virtual guest clock. This can be achieved as follows:

```
VBoxManage setextradata "VM name" "VBoxInternal/TM/WarpDrivePercentage" 200
```

The above example will double the speed of the guest clock while

```
VBoxManage setextradata "VM name" "VBoxInternal/TM/WarpDrivePercentage" 50
```

will halve the speed of the guest clock. Note that changing the rate of the virtual clock can confuse the guest and can even lead to abnormal guest behavior. For instance, a higher clock rate means shorter timeouts for virtual devices with the result that a slightly increased response time of a virtual device due to an increased host load can cause guest failures. Note further that any time synchronization mechanism will frequently try to resynchronize the guest clock with the reference clock, which is the host clock if the Oracle VM VirtualBox Guest Additions are active. Therefore any time synchronization should be disabled if the rate of the guest clock is changed as described above. See chapter 9.13.3, *Tuning the Guest Additions Time Synchronization Parameters*, page 226.

### 9.13.3 Tuning the Guest Additions Time Synchronization Parameters

The Oracle VM VirtualBox Guest Additions ensure that the guest's system time is synchronized with the host time. There are several parameters which can be tuned. The parameters can be set for a specific VM using the following command:

```
VBoxManage guestproperty set "VM name" "/VirtualBox/GuestAdd/VBoxService/PARAMETER" VALUE
```

where `PARAMETER` is one of the following:

**`--timesync-interval`**

Specifies the interval at which to synchronize the time with the host. The default is 10000 ms (10 seconds).

**`--timesync-min-adjust`**

The minimum absolute drift value measured in milliseconds to make adjustments for. The default is 1000 ms on OS/2 and 100 ms elsewhere.

**`--timesync-latency-factor`**

The factor to multiply the time query latency with to calculate the dynamic minimum adjust time. The default is 8 times, which means as follows:

Measure the time it takes to determine the host time, the guest has to contact the VM host service which may take some time. Multiply this value by 8 and do an adjustment only if the time difference between host and guest is bigger than this value. Do not do any time adjustment otherwise.

**`--timesync-max-latency`**

The max host timer query latency to accept. The default is 250 ms.

**`--timesync-set-threshold`**

The absolute drift threshold, given as milliseconds where to start setting the time instead of trying to smoothly adjust it. The default is 20 minutes.

**`--timesync-set-start`**

Set the time when starting the time sync service.

**--timesync-set-on-restore 0|1**

Set the time after the VM was restored from a saved state when passing 1 as parameter. This is the default. Disable by passing 0. In the latter case, the time will be adjusted smoothly, which can take a long time.

All these parameters can be specified as command line parameters to VBoxService as well.

### 9.13.4 Disabling the Guest Additions Time Synchronization

Once installed and started, the Oracle VM VirtualBox Guest Additions will try to synchronize the guest time with the host time. This can be prevented by forbidding the guest service from reading the host clock:

```
VBoxManage setextradata "VM name" "VBoxInternal/Devices/VMMDev/0/Config/GetHostTimeDisabled" 1
```

## 9.14 Installing the Alternate Bridged Networking Driver on Oracle Solaris 11 hosts

Oracle VM VirtualBox includes a network filter driver that utilizes Oracle Solaris 11's Crossbow functionality. By default, this new driver is installed for Oracle Solaris 11 hosts, builds 159 and above, that have support for it.

To force installation of the older STREAMS based network filter driver, execute as root the following command before installing the Oracle VM VirtualBox package:

```
touch /etc/vboxinst_vboxflt
```

To force installation of the Crossbow based network filter driver, execute as root the following command before installing the Oracle VM VirtualBox package:

```
touch /etc/vboxinst_vboxbow
```

To check which driver is currently being used by Oracle VM VirtualBox, execute:

```
modinfo | grep vbox
```

If the output contains "vboxbow", it indicates Oracle VM VirtualBox is using the Crossbow network filter driver, while the name "vboxflt" indicates usage of the older STREAMS network filter.

## 9.15 Oracle VM VirtualBox VNIC Templates for VLANs on Oracle Solaris 11 Hosts

Oracle VM VirtualBox supports Virtual Network Interface (VNIC) templates for configuring VMs over VLANs. An Oracle VM VirtualBox VNIC template is a VNIC whose name starts with `vboxvnic_template`. The string is case-sensitive.

On Oracle Solaris 11 hosts, when Crossbow-based bridged networking is used, a VNIC template may be used to specify the VLAN ID to use while bridging over a network link.

The following is an example of how to use a VNIC template to configure a VM over a VLAN. Create an Oracle VM VirtualBox VNIC template, by executing as root:

```
dladm create-vnic -t -l nge0 -v 23 vboxvnic_template0
```

This will create a temporary VNIC template over interface “nge0” with the VLAN ID 23. To create VNIC templates that are persistent across host reboots, skip the `-t` parameter in the above command. You may check the current state of links using the following command:

```
$ dladm show-link
LINK        CLASS     MTU    STATE    BRIDGE     OVER
nge0        phys      1500   up       --         --
nge1        phys      1500   down     --         --
vboxvnic_template0 vnic 1500 up       --         nge0

$ dladm show-vnic
LINK         OVER         SPEED  MACADDRESS        MACADDRTYPE         VID
vboxvnic_template0 nge0   1000   2:8:20:25:12:75   random              23
```

Once the VNIC template is created, any VMs that need to be on VLAN 23 over the interface `nge0` can be configured to bridge using this VNIC template.

VNIC templates makes managing VMs on VLANs simpler and efficient. The VLAN details are not stored as part of every VM’s configuration but rather inherited from the VNIC template while starting the VM. The VNIC template itself can be modified anytime using the `dladm` command.

VNIC templates can be created with additional properties such as bandwidth limits, CPU fanout etc. Refer to your Oracle Solaris network documentation on how to accomplish this. These additional properties, if any, are also applied to VMs which bridge using the VNIC template.

## 9.16 Configuring Multiple Host-Only Network Interfaces on Oracle Solaris Hosts

By default Oracle VM VirtualBox provides you with one host-only network interface. Adding more host-only network interfaces on Oracle Solaris hosts requires manual configuration. Here is how to add another host-only network interface.

Begin by stopping all running VMs. Then, unplumb the existing “vboxnet0” interface by execute the following command as root:

```
ifconfig vboxnet0 unplumb
```

If you have several vboxnet interfaces, you will need to unplumb all of them. Once all vboxnet interfaces are unplumbed, remove the driver by executing the following command as root:

```
rem_drv vboxnet
```

Edit the file `/platform/i86pc/kernel/drv/vboxnet.conf` and add a line for the new interface we want to add as shown below:

```
name="vboxnet" parent="pseudo" instance=1;
name="vboxnet" parent="pseudo" instance=2;
```

Add as many of these lines as required with each line having a unique instance number.

Next, reload the vboxnet driver by executing the following command as root:

```
add_drv vboxnet
```

On Oracle Solaris 11.1 and newer hosts you may want to rename the default vanity interface name. To check what name has been assigned, execute:

```
dladm show-phys
LINK              MEDIA               STATE      SPEED  DUPLEX    DEVICE
net0              Ethernet            up         100    full      e1000g0
net2              Ethernet            up         1000   full      vboxnet1
net1              Ethernet            up         1000   full      vboxnet0
```

In the above example, we can rename "net2" to "vboxnet1" before proceeding to plumb the interface. This can be done by executing as root:

```
dladm rename-link net2 vboxnet1
```

Now plumb all the interfaces using `ifconfig vboxnetX plumb`, where 'X' would be 1 in this case. Once the interface is plumbed, it may be configured like any other network interface. Refer to the `ifconfig` documentation for further details.

To make the settings for the newly added interfaces persistent across reboots, you will need to edit the files `/etc/inet/netmasks`, and if you are using NWAM `/etc/nwam/llp` and add the appropriate entries to set the netmask and static IP for each of those interfaces. The Oracle VM VirtualBox installer only updates these configuration files for the one "vboxnet0" interface it creates by default.

## 9.17 Configuring the Oracle VM VirtualBox CoreDumper on Oracle Solaris Hosts

Oracle VM VirtualBox is capable of producing its own core files for extensive debugging when things go wrong. Currently this is only available on Oracle Solaris hosts.

The Oracle VM VirtualBox CoreDumper can be enabled using the following command:

```
VBoxManage setextradata "VM name" VBoxInternal2/CoreDumpEnabled 1
```

You can specify which directory to use for core dumps with this command, as follows:

```
VBoxManage setextradata "VM name" VBoxInternal2/CoreDumpDir <path-to-directory>
```

Make sure the directory you specify is on a volume with sufficient free space and that the Oracle VM VirtualBox process has sufficient permissions to write files to this directory. If you skip this command and do not specify any core dump directory, the current directory of the Oracle VM VirtualBox executable will be used. This would most likely fail when writing cores as they are protected with root permissions. It is recommended you explicitly set a core dump directory.

You must specify when the Oracle VM VirtualBox CoreDumper should be triggered. This is done using the following commands:

```
VBoxManage setextradata "VM name" VBoxInternal2/CoreDumpReplaceSystemDump 1
VBoxManage setextradata "VM name" VBoxInternal2/CoreDumpLive 1
```

At least one of the above two commands will have to be provided if you have enabled the Oracle VM VirtualBox CoreDumper.

Setting `CoreDumpReplaceSystemDump` sets up the VM to override the host's core dumping mechanism and in the event of any crash only the Oracle VM VirtualBox CoreDumper would produce the core file.

Setting `CoreDumpLive` sets up the VM to produce cores whenever the VM process receives a `SIGUSR2` signal. After producing the core file, the VM will not be terminated and will continue to run. You can thus take cores of the VM process using the following command:

```
kill -s SIGUSR2 <VM-process-id>
```

Core files produced by the Oracle VM VirtualBox CoreDumper are of the form `core.vb.<ProcessName>.<Process` for example `core.vb.VBoxHeadless.11321`.

## 9.18 Oracle VM VirtualBox and Oracle Solaris Kernel Zones

Oracle Solaris kernel zones on x86-based systems make use of hardware-assisted virtualization features like Oracle VM VirtualBox does. However, for kernel zones and Oracle VM VirtualBox to share this hardware resource, they need to cooperate.

By default, due to performance reasons, Oracle VM VirtualBox acquires the hardware-assisted virtualization resource (VT-x/AMD-V) globally on the host machine and uses it until the last Oracle VM VirtualBox VM that requires it is powered off. This prevents other software from using VT-x/AMD-V during the time Oracle VM VirtualBox has taken control of it.

Oracle VM VirtualBox can be instructed to relinquish use of hardware-assisted virtualization features when not executing guest code, thereby allowing kernel zones to make use of them. To do this, shutdown all Oracle VM VirtualBox VMs and execute the following command:

```
VBoxManage setproperty hwvirtexclusive off
```

This command needs to be executed only once as the setting is stored as part of the global Oracle VM VirtualBox settings which will continue to persist across host-reboots and Oracle VM VirtualBox upgrades.

## 9.19 Locking Down the Oracle VM VirtualBox GUI

### 9.19.1 Customizing the VirtualBox Manager

There are several advanced customization settings for locking down the VirtualBox Manager. Locking down means removing some features that the user should not see.

```
VBoxManage setextradata global GUI/Customizations OPTION[,OPTION...]
```

where `OPTION` is one of the following keywords:

**`noSelector`**

Do not allow users to start the VirtualBox Manager. Trying to do so will show a window containing a proper error message.

**`noMenuBar`**

VM windows will not contain a menu bar.

**`noStatusBar`**

VM windows will not contain a status bar.

To disable any of these VirtualBox Manager customizations use the following command:

```
VBoxManage setextradata global GUI/Customizations
```

### 9.19.2 VM Selector Customization

The following per-machine VM extradata settings can be used to change the behavior of the VM selector window in respect of certain VMs:

```
VBoxManage setextradata "VM name" SETTING true
```

where `SETTING` can be:

**GUI/HideDetails**

Do not show the VM configuration of a certain VM. The details window will remain just empty if this VM is selected.

**GUI/PreventReconfiguration**

Do not allow the user to open the **Settings** dialog for a certain VM.

**GUI/PreventSnapshotOperations**

Prevent snapshot operations for a VM from the GUI, either at runtime or when the VM is powered off.

**GUI/HideFromManager**

Hide a certain VM in the VM selector window.

**GUI/PreventApplicationUpdate**

Disable the automatic update check and hide the corresponding menu item.

Please note that these settings would not prevent the user from reconfiguring the VM by using `VBoxManage modifyvm`.

### 9.19.3 Configure VM Selector Menu Entries

You can disable, or blacklist, certain entries in the global settings page of the VM selector:

```
VBoxManage setextradata global GUI/RestrictedGlobalSettingsPages OPTION[,OPTION...]
```

where `OPTION` is one of the following keywords:

**General**

Do not show the **General** settings pane.

**Input**

Do not show the **Input** settings pane.

**Update**

Do not show the **Update** settings pane.

**Language**

Do not show the **Language** settings pane.

**Display**

Do not show the **Display** settings pane.

**Network**

Do not show the **Network** settings pane.

**Extensions**

Do not show the **Extensions** settings pane.

**Proxy**

Do not show the **Proxy** settings pane.

This is a global setting. Any combination of the above is allowed. To restore the default behavior, use

```
VBoxManage setextradata global GUI/RestrictedGlobalSettingsPages
```

### 9.19.4 Configure VM Window Menu Entries

You can disable, or blacklist, certain menu actions in the VM window:

```
VBoxManage setextradata "VM name" GUI/RestrictedRuntimeMenus OPTION[,OPTION...]
```

where `OPTION` is one of the following keywords:

**All**

Do not show any menu in the VM window.

**Machine**

Do not show the **Machine** menu in the VM window.

**View**

Do not show the **View** menu in the VM window.

**Devices**

Do not show the **Devices** menu in the VM window.

**Help**

Do not show the **Help** menu in the VM window.

**Debug**

Do not show the **Debug** menu in the VM window. The Debug menu is only visible if the GUI was started with special command line parameters or environment variable settings.

This is a per-VM setting. Any combination of the above is allowed. To restore the default behavior, use

```
VBoxManage setextradata "VM name" GUI/RestrictedRuntimeMenus
```

You can also disable, or blacklist, certain menu actions of certain menus. Use the following command to disable certain actions of the **Application** menu. This is only available on Mac OS X hosts.

```
VBoxManage setextradata "VM name" GUI/RestrictedRuntimeApplicationMenuActions OPTION[,OPTION...]
```

where `OPTION` is one of the following keywords:

**`All`**

Do not show any menu item in this menu.

**`About`**

Do not show the **About** menu item in this menu.

This is a per-VM setting. Any combination of the above is allowed. To restore the default behavior, use

```
VBoxManage setextradata "VM name" GUI/RestrictedRuntimeMenus
```

Use the following command to disable certain actions of the **Machine** menu:

```
VBoxManage setextradata "VM name" GUI/RestrictedRuntimeApplicationMenuActions OPTION[,OPTION...]
```

where `OPTION` is one of the following keywords:

**`All`**

Do not show any menu item in this menu.

**`SettingsDialog`**

Do not show the **Settings** menu item in this menu.

**`TakeSnapshot`**

Do not show the **Take Snapshot** menu item in this menu.

**`TakeScreenshot`**

Do not show the **Take Screenshot** menu item in this menu.

**`InformationDialog`**

Do not show the **Session Information** menu item in this menu.

**`MouseIntegration`**

Do not show the **Disable Mouse Integration** menu item in this menu.

**`TypeCAD`**

Do not show the **Insert Ctrl+Alt+Del** menu item in this menu.

**TypeCABS**

Do not show the **Insert Ctrl+Alt+Backspace** menu item in this menu. Available on X11 hosts only.

**Pause**

Do not show the **Pause** menu item in this menu.

**Reset**

Do not show the **Reset** menu item in this menu.

**SaveState**

Do not show the **Save the machine state** menu item in this menu.

**Shutdown**

Do not show the **ACPI Shutdown** menu item in this menu.

**PowerOff**

Do not show the **Power Off the machine** menu item in this menu.

This is a per-VM setting. Any combination of the above is allowed. To restore the default behavior, use

```
VBoxManage setextradata "VM name" GUI/RestrictedRuntimeApplicationMenuActions
```

Use the following command to disable certain actions of the **View** menu:

```
VBoxManage setextradata "VM name" GUI/RestrictedRuntimeViewMenuActions OPTION[,OPTION...]
```

where `OPTION` is one of the following keywords:

**All**

Do not show any menu item in this menu.

**Fullscreen**

Do not show the **Switch to Fullscreen** menu item in this menu.

**Seamless**

Do not show the **Switch to Seamless Mode** menu item in this menu.

**Scale**

Do not show the **Switch to Scaled Mode** menu item in this menu.

**GuestAutoresize**

Do not show the **Auto-resize Guest Display** menu item in this menu.

**AdjustWindow**

Do not show the **Adjust Window Size** menu item in this menu.

**Multiscreen**

Do not show the **Multiscreen** menu item in this menu. Only visible in full screen/seamless mode.

This is a per-VM setting. Any combination of the above is allowed. To restore the default behavior, use

```
VBoxManage setextradata "VM name" GUI/RestrictedRuntimeViewMenuActions
```

Use the following command to disable certain actions of the **View** menu:

```
VBoxManage setextradata "VM name" GUI/RestrictedRuntimeDevicesMenuActions OPTION[,OPTION...]
```

where `OPTION` is one of the following keywords to disable actions in the **Devices** menu:

**All**

Do not show any menu item in this menu.

**OpticalDevices**

Do not show the **CD/DVD Devices** menu item in this menu.

**FloppyDevices**

Do not show the **Floppy Devices** menu item in this menu.

**USBDevices**

Do not show the **USB Devices** menu item in this menu.

**SharedClipboard**

Do not show the **Shared Clipboard** menu item in this menu.

**DragAndDrop**

Do not show the **Drag and Drop** menu item in this menu.

**NetworkSettings**

Do not show the **Network Settings...** menu item in this menu.

**SharedFoldersSettings**

Do not show the **Shared Folders Settings...** menu item in this menu.

**VRDEServer**

Do not show the **Remove Display** menu item in this menu.

**InstallGuestTools**

Do not show the **Insert Guest Additions CD image...** menu item in this menu.

This is a per-VM setting. Any combination of the above is allowed. To restore the default behavior, use

```
VBoxManage setextradata "VM name" GUI/RestrictedRuntimeDevicesMenuActions
```

Use the following command to disable certain actions of the **View** menu:

```
VBoxManage setextradata "VM name" GUI/RestrictedRuntimeDebuggerMenuActions OPTION[,OPTION...]
```

where `OPTION` is one of the following keywords to disable actions in the *Debug* menu, which is normally completely disabled:

**All**

Do not show any menu item in this menu.

**Statistics**

Do not show the **Statistics...** menu item in this menu.

**CommandLine**

Do not show the **Command Line...** menu item in this menu.

**Logging**

Do not show the **Logging...** menu item in this menu.

**LogDialog**

Do not show the **Show Log...** menu item in this menu.

This is a per-VM setting. Any combination of the above is allowed. To restore the default behavior, use

```
VBoxManage setextradata "VM name" GUI/RestrictedRuntimeDebuggerMenuActions
```

Use the following command to disable certain actions of the **View** menu:

```
VBoxManage setextradata "VM name" GUI/RestrictedRuntimeHelpMenuActions OPTION[,OPTION...]
```

where `OPTION` is one of the following keywords to disable actions in the **Help** menu, which is normally completely disabled:

**All**

Do not show any menu item in this menu.

**Contents**

Do not show the **Contents...** menu item in this menu.

**WebSite**

Do not show the **VirtualBox Web Site...** menu item in this menu.

**ResetWarnings**

Do not show the **Reset All Warnings** menu item in this menu.

**NetworkAccessManager**

Do not show the **Network Operations Manager** menu item in this menu.

**About**

Do not show the **About** menu item in this menu. Only for non-Mac OS X hosts.

**Contents**

Do not show the **Contents...** menu item in this menu.

**Contents**

Do not show the **Contents...** menu item in this menu.

This is a per-VM setting. Any combination of the above is allowed. To restore the default behavior, use

```
VBoxManage setextradata "VM name" GUI/RestrictedRuntimeHelpMenuActions
```

### 9.19.5 Configure VM Window Status Bar Entries

You can disable, or blacklist, certain status bar items:

```
VBoxManage setextradata "VM name" GUI/RestrictedStatusBarIndicators OPTION[,OPTION...]
```

where `OPTION` is one of the following keywords:

**HardDisks**

Do not show the hard disk icon in the VM window status bar. By default the hard disk icon is only shown if the VM configuration contains one or more hard disks.

**OpticalDisks**

Do not show the CD icon in the VM window status bar. By default the CD icon is only shown if the VM configuration contains one or more CD drives.

**FloppyDisks**

Do not show the floppy icon in the VM window status bar. By default the floppy icon is only shown if the VM configuration contains one or more floppy drives.

**Network**

Do not show the network icon in the VM window status bar. By default the network icon is only shown if the VM configuration contains one or more active network adapters.

**USB**

Do not show the USB icon in the status bar.

**SharedFolders**

Do not show the shared folders icon in the status bar.

**Capture**

Do not show the capture icon in the status bar.

**Features**

Do not show the CPU features icon in the status bar.

**Mouse**

Do not show the mouse icon in the status bar.

**Keyboard**

Do not show the keyboard icon in the status bar.

This is a per-VM setting. Any combination of the above is allowed. If all options are specified, no icons are displayed in the status bar of the VM window. To restore the default behavior, use

```
VBoxManage setextradata "VM name" GUI/RestrictedStatusBarIndicators
```

### 9.19.6 Configure VM Window Visual Modes

You can disable, or blacklist, certain VM visual modes:

```
VBoxManage setextradata "VM name" GUI/RestrictedVisualStates OPTION[,OPTION...]
```

where `OPTION` is one of the following keywords:

**Fullscreen**

Do not allow to switch the VM into full screen mode.

**Seamless**

Do not allow to switch the VM into seamless mode.

**Scale**

Do not allow to switch the VM into scale mode.

This is a per-VM setting. Any combination of the above is allowed. To restore the default behavior, use

```
VBoxManage setextradata "VM name" GUI/RestrictedVisualStates
```

### 9.19.7 Host Key Customization

To disable all Host key combinations, open the preferences and change the Host key to None. This might be useful when using Oracle VM VirtualBox in a kiosk mode.

To redefine or disable certain Host key actions, use the following command:

```
VBoxManage setextradata global GUI/Input/MachineShortcuts "FullscreenMode=F,...."
```

The following table shows the possible Host key actions, together with their default Host key shortcut. Setting an action to None will disable that Host key action.

| Action | Default Key | Action |
|---|---|---|
| TakeSnapshot | T | Take a snapshot |
| TakeScreenshot | E | Take a screenshot |
| MouseIntegration | I | Toggle mouse integration |
| TypeCAD | Del | Inject Ctrl+Alt+Del |
| TypeCABS | Backspace | Inject Ctrl+Alt+Backspace |
| Pause | P | Pause the VM |
| Reset | R | Hard reset the guest |
| SaveState | | Save the VM state and terminate the VM |
| Shutdown | H | Press the virtual ACPI power button |
| PowerOff | | Power off the VM without saving the state |
| Close | Q | Show the Close VM dialog |
| FullscreenMode | F | Switch the VM into full screen mode |
| SeamlessMode | L | Switch the VM into seamless mode |
| ScaleMode | C | Switch the VM into scaled mode |
| GuestAutoResize | G | Automatically resize the guest window |
| WindowAdjust | A | Immediately resize the guest window |
| PopupMenu | Home | Show the popup menu in full screen mode and seamless mode |
| SettingsDialog | S | Open the VM Settings dialog |
| InformationDialog | N | Show the VM Session Information window |
| NetworkAdaptersDialog | | Show the VM Network Adapters dialog |
| SharedFoldersDialog | | Show the VM Shared Folders dialog |
| InstallGuestAdditions | D | Mount the ISO containing the Guest Additions |

To disable full screen mode and seamless mode, use the following command:

```
VBoxManage setextradata global GUI/Input/MachineShortcuts "FullscreenMode=None,SeamlessMode=None"
```

### 9.19.8 Action when Terminating the VM

You can disallow, or blacklist, certain actions when terminating a VM. To disallow specific actions, use the following command:

```
VBoxManage setextradata "VM name" GUI/RestrictedCloseActions OPTION[,OPTION...]
```

where `OPTION` is one of the following keywords:

**`SaveState`**

Do not allow the user to save the VM state when terminating the VM.

**`Shutdown`**

Do not allow the user to shutdown the VM by sending the ACPI power-off event to the guest.

**`PowerOff`**

Do not allow the user to power off the VM.

**`PowerOffRestoringSnapshot`**

Do not allow the user to return to the last snapshot when powering off the VM.

**`Detach`**

Do not allow the user to detach from the VM process if the VM was started in separate mode.

This is a per-VM setting. Any combination of the above is allowed. If all options are specified, the VM cannot be shut down at all.

### 9.19.9 Default Action when Terminating the VM

You can define a specific action for terminating a VM. In contrast to the setting decribed in the previous section, this setting allows only one action when the user terminates the VM. No exit menu is shown. Use the following command:

```
VBoxManage setextradata "VM name" GUI/DefaultCloseAction ACTION
```

where `ACTION` is one of the following keywords:

**`SaveState`**

Save the VM state before terminating the VM process.

**`Shutdown`**

The VM is shut down by sending the ACPI power-off event to the guest.

**PowerOff**

The VM is powered off.

**PowerOffRestoringSnapshot**

The VM is powered off and the saved state returns to the last snapshot.

**Detach**

Terminate the frontend but leave the VM process running.

This is a per-VM setting. Any combination of the above is allowed. If all options are specified, the VM cannot be shut down at all.

### 9.19.10 Action for Handling a Guru Meditation

A VM runs into a Guru Meditation if there is a problem which cannot be fixed by other means than terminating the process. The default is to show a message window which instructs the user to open a bug report.

This behavior can be configured as follows:

```
VBoxManage setextradata "VM name" GUI/GuruMeditationHandler MODE
```

where `MODE` is one of the following keywords:

**Default**

A message window is shown. After the user confirmed, the VM is terminated.

**PowerOff**

The VM is immediately powered-off without showing any message window. The VM logfile will show information about what happened.

**Ignore**

The VM is left in stuck mode. Execution is stopped but no message window is shown. The VM has to be powered off manually.

This is a per-VM setting.

### 9.19.11 Configuring Automatic Mouse Capturing

By default, the mouse is captured if the user clicks on the guest window and the guest expects relative mouse coordinates at this time. This happens if the pointing device is configured as PS/2 mouse and the guest has not yet started the Oracle VM VirtualBox Guest Additions. For instance, the guest is booting or the Guest Additions are not installed, or if the pointing device is configured as a USB tablet but the guest has no USB driver loaded yet. Once the Guest Additions become active or the USB guest driver is started, the mouse capture is automatically released.

The default behavior is sometimes not desired. Therefore it can be configured as follows:

```
VBoxManage setextradata "VM name" GUI/MouseCapturePolicy MODE
```

where `MODE` is one of the following keywords:

**Default**

The default behavior as described above.

**HostComboOnly**

The mouse is only captured if the Host Key is toggled.

**Disabled**

The mouse is never captured, also not by toggling the Host Key

This is a per-VM setting.

### 9.19.12 Requesting Legacy Full-Screen Mode

Oracle VM VirtualBox uses special window manager facilities to switch a multi-screen machine to full-screen on a multi-monitor host system. However, not all window managers provide these facilities correctly. Oracle VM VirtualBox can be configured to use a legacy method of switching to full-screen mode instead, by using the command:

```
VBoxManage setextradata global GUI/Fullscreen/LegacyMode true
```

You can go back to the default method by using the following command:

```
VBoxManage setextradata global GUI/Fullscreen/LegacyMode
```

This is a global setting.

## 9.20 Starting the Oracle VM VirtualBox Web Service Automatically

The Oracle VM VirtualBox web service, `vboxwebsrv`, is used for controlling Oracle VM VirtualBox remotely. It is documented in detail in the Oracle VM VirtualBox Software Development Kit (SDK). See chapter 11, *Oracle VM VirtualBox Programming Interfaces*, page 276. As the client base using this interface is growing, we added start scripts for the various operation systems we support. The following sections describe how to use them. The Oracle VM VirtualBox web service is never started automatically as a result of a standard installation.

### 9.20.1 Linux: Starting the Web Service With init

On Linux, the web service can be automatically started during host boot by adding appropriate parameters to the file `/etc/default/virtualbox`. There is one mandatory parameter, `VBOXWEB_USER`, which must be set to the user which will later start the VMs. The parameters in the following table all start with the `VBOXWEB_` prefix string. For example: `VBOXWEB_HOST` and `VBOXWEB_PORT`.

| Parameter | Description | Default |
|---|---|---|
| USER | The user which the web service runs as | |
| HOST | The host to bind the web service to | local-host |
| PORT | The port to bind the web service to | 18083 |
| SSL_KEYFILE | Server key and certificate file, in PEM format | |
| SSL_PASSWORDFILE | File name for password to server key | |
| SSL_CACERT | CA certificate file, in PEM format | |
| SSL_CAPATH | CA certificate path | |
| SSL_DHFILE | DH file name or DH key length in bits | |
| SSL_RANDFILE | File containing seed for random number generator | |
| TIMEOUT | Session timeout in seconds, 0 disables timeouts | 300 |
| CHECK_INTERVAL | Frequency of timeout checks in seconds | 5 |
| THREADS | Maximum number of worker threads to run in parallel | 100 |
| KEEPALIVE | Maximum number of requests before a socket will be closed | 100 |
| ROTATE | Number of log files, 0 disables log rotation | 10 |
| LOGSIZE | Maximum log file size to trigger rotation, in bytes | 1MB |
| LOGINTERVAL | Maximum time interval to trigger log rotation, in seconds | 1 day |

Setting the parameter SSL_KEYFILE enables the SSL/TLS support. Using encryption is strongly encouraged, as otherwise everything, including passwords, is transferred in clear text.

### 9.20.2 Oracle Solaris: Starting the Web Service With SMF

On Oracle Solaris hosts, the Oracle VM VirtualBox web service daemon is integrated into the SMF framework. You can change the parameters, but do not have to if the defaults below already match your needs:

```
svccfg -s svc:/application/virtualbox/webservice:default setprop config/host=localhost
svccfg -s svc:/application/virtualbox/webservice:default setprop config/port=18083
svccfg -s svc:/application/virtualbox/webservice:default setprop config/user=root
```

chapter 9.20.1, *Linux: Starting the Web Service With init*, page 242 showing the parameter names and defaults also applies for Oracle Solaris. The parameter names must be changed to lowercase and a prefix of config/ has to be added. For example: config/user or config/ssl_keyfile. If you make any change, do not forget to run the following command to put the changes into effect immediately:

```
svcadm refresh svc:/application/virtualbox/webservice:default
```

If you forget the above command then the previous settings are used when enabling the service. Check the current property settings as follows:

```
svcprop -p config svc:/application/virtualbox/webservice:default
```

When everything is configured correctly you can start the Oracle VM VirtualBox web service with the following command:

```
svcadm enable svc:/application/virtualbox/webservice:default
```

For more information about SMF, please refer to the Oracle Solaris documentation.

### 9.20.3 Mac OS X: Starting the Web Service With launchd

On Mac OS X, launchd is used to start the Oracle VM VirtualBox webservice. An example configuration file can be found in `$HOME/Library/LaunchAgents/org.virtualbox.vboxwebsrv.plist`. It can be enabled by changing the `Disabled` key from `true` to `false`. To manually start the service use the following command:

```
launchctl load ~/Library/LaunchAgents/org.virtualbox.vboxwebsrv.plist
```

For additional information on how launchd services could be configured see:
https://developer.apple.com/library/mac/documentation/MacOSX/Conceptual/BPSystemStartup/Chapters/CreatingLaunchdJobs.html.

## 9.21 Oracle VM VirtualBox Watchdog

The memory ballooning service, formerly known as `VBoxBalloonCtrl`, was renamed to VBoxWatchdog. This service now incorporates the following host services that are meant to be run in a server environment:

- **Memory ballooning control.** This service automatically takes care of a VM's configured memory balloon. See chapter 4.10.1, *Memory Ballooning*, page 80. This service is useful for server environments where VMs may dynamically require more or less memory during runtime.

  The service periodically checks a VM's current memory balloon and its free guest RAM and automatically adjusts the current memory balloon by inflating or deflating it accordingly. This handling only applies to running VMs having recent Guest Additions installed.

- **Host isolation detection.** This service provides a way to detect whether the host cannot reach the specific Oracle VM VirtualBox server instance anymore and take appropriate actions, such as shutting down, saving the current state or even powering down certain VMs.

All configuration values can be either specified using the command line or global extradata, whereas command line values always have a higher priority when set. Some of the configuration values also be specified on a per-VM basis. So the overall lookup order is: command line, per-VM basis extradata if available, global extradata.

### 9.21.1 Memory Ballooning Control

The memory ballooning control inflates and deflates the memory balloon of VMs based on the VMs free memory and the desired maximum balloon size.

To set up the memory ballooning control the maximum ballooning size a VM can reach needs to be set. This can be specified using the command line, as follows:

```
--balloon-max <Size in MB>
```

Using a per-VM basis extradata value, as follows:

```
VBoxManage setextradata <VM-Name> VBoxInternal2/Watchdog/BalloonCtrl/BalloonSizeMax <Size in MB>
```

Using a global extradata value, as follows:

```
VBoxManage setextradata global VBoxInternal2/Watchdog/BalloonCtrl/BalloonSizeMax <Size in MB>
```

**Note:** If no maximum ballooning size is specified by at least one of the parameters above, no ballooning will be performed at all.

Setting the ballooning increment in MB can be either done using command line, as follows:

```
--balloon-inc <Size in MB>
```

Using a global extradata value, as follows:

```
VBoxManage setextradata global VBoxInternal2/Watchdog/BalloonCtrl/BalloonIncrementMB <Size in MB>
```

The default ballooning increment is 256 MB if not specified.
The same options apply for a ballooning decrement. Using the command line, as follows:

```
--balloon-dec <Size in MB>
```

Using a global extradata value, as follows:

```
VBoxManage setextradata global VBoxInternal2/Watchdog/BalloonCtrl/BalloonDecrementMB <Size in MB>
```

The default ballooning decrement is 128 MB if not specified.
The lower limit in MB for a balloon can be defined using the command line, as follows:

```
--balloon-lower-limit <Size in MB>
```

Using a global extradata value, as follows:

```
VBoxManage setextradata global VBoxInternal2/Watchdog/BalloonCtrl/BalloonLowerLimitMB <Size in MB>
```

The default lower limit is 128 MB if not specified.

### 9.21.2 Host Isolation Detection

To detect whether a host is being isolated, that is, the host cannot reach the Oracle VM VirtualBox server instance anymore, the host needs to set an alternating value to a global extradata value within a time period. If this value is not set within that time period a timeout occurred and the so-called host isolation response will be performed to the VMs handled. Which VMs are handled can be controlled by defining VM groups and assigning VMs to those groups. By default no groups are set, meaning that all VMs on the server will be handled when no host response is received within 30 seconds.
Set the groups handled by the host isolation detection using the following command line:

```
--apimon-groups=<string[,stringN]>
```

Using a global extradata value, as follows:

```
VBoxManage setextradata global VBoxInternal2/Watchdog/APIMonitor/Groups <string[,stringN]>
```

Set the host isolation timeout using the following command line:

```
--apimon-isln-timeout=<ms>
```

Using a global extradata value, as follows:

```
VBoxManage setextradata global VBoxInternal2/Watchdog/APIMonitor/IsolationTimeoutMS <ms>
```

Set the actual host isolation response using the following command line:

```
--apimon-isln-response=<cmd>
```

Using a global extradata value, as follows:

```
VBoxManage setextradata global VBoxInternal2/Watchdog/APIMonitor/IsolationResponse <cmd>
```

The following response commands are available:

- `none`. This has no effect.
- `pause`. Pauses the execution of a VM.
- `poweroff`. Shuts down the VM by pressing the virtual power button. The VM will not have the chance of saving any data or veto the shutdown process.
- `save`. Saves the current machine state and powers off the VM afterwards. If saving the machine state fails the VM will be paused.
- `shutdown`. Shuts down the VM in a gentle way by sending an `ACPI` shutdown event to the VM's operating system. The OS then has the chance of doing a clean shutdown.

### 9.21.3 More Information

For more advanced options and parameters like verbose logging check the built-in command line help accessible with `--help`.

### 9.21.4 Linux: Starting the Watchdog Service With init

On Linux, the watchdog service can be automatically started during host boot by adding appropriate parameters to the file `/etc/default/virtualbox`. There is one mandatory parameter, `VBOXWATCHDOG_USER`, which must be set to the user which will later start the VMs. For backward compatibility you can also specify `VBOXBALLOONCTRL_USER`.

The parameters in the following table all start with the `VBOXWATCHDOG_` prefix string. For example: `VBOXWATCHDOG_BALLOON_INTERVAL` and `VBOXWATCHDOG_LOGSIZE`. Legacy parameters such as `VBOXBALLOONCTRL_INTERVAL` can still be used.

| Parameter | Description | Default |
|---|---|---|
| `USER` | The user which the watchdog service runs as | |
| `ROTATE` | Number of log files, 0 disables log rotation | 10 |
| `LOGSIZE` | Maximum log file size to trigger rotation, in bytes | 1MB |
| `LOGINTERVAL` | Maximum time interval to trigger log rotation, in seconds | 1 day |
| `BALLOON_INTERVAL` | Interval for checking the balloon size, in milliseconds | 30000 |
| `BALLOON_INCREMENT` | Balloon size increment, in megabytes | 256 |
| `BALLOON_DECREMENT` | Balloon size decrement, in megabytes | 128 |
| `BALLOON_LOWERLIMIT` | Balloon size lower limit, in megabytes | 64 |
| `BALLOON_SAFETYMARGIN` | Free memory required for decreasing the balloon size, in megabytes | 1024 |

### 9.21.5 Oracle Solaris: Starting the Watchdog Service With SMF

On Oracle Solaris hosts, the Oracle VM VirtualBox watchdog service daemon is integrated into the SMF framework. You can change the parameters, but do not have to if the defaults already match your needs:

```
svccfg -s svc:/application/virtualbox/balloonctrl:default setprop \
  config/balloon_interval=10000
svccfg -s svc:/application/virtualbox/balloonctrl:default setprop \
config/balloon_safetymargin=134217728
```

chapter 9.21.4, *Linux: Starting the Watchdog Service With init*, page 246 also applies for Oracle Solaris. The parameter names must be changed to lowercase and a prefix of `config/` has to be added. For example: `config/user` or `config/balloon_safetymargin`. If you made any change, do not forget to run the following command to put the changes into effect immediately:

```
svcadm refresh svc:/application/virtualbox/balloonctrl:default
```

If you forget the above command then the previous settings will be used when enabling the service. Check the current property settings with the following command:

```
svcprop -p config svc:/application/virtualbox/balloonctrl:default
```

When everything is configured correctly you can start the Oracle VM VirtualBox watchdog service with the following command:

```
svcadm enable svc:/application/virtualbox/balloonctrl:default
```

For more information about SMF, please refer to the Oracle Solaris documentation.

## 9.22 Other Extension Packs

Another extension pack called VNC is available. This extension pack is open source and replaces the previous integration of the VNC remote access protocol. This is experimental code, and is initially available in the Oracle VM VirtualBox source code package only. It is to a large portion code contributed by users, and is not supported in any way by Oracle.

The keyboard handling is severely limited, and only the US keyboard layout works. Other keyboard layouts will have at least some keys which produce the wrong results, often with quite surprising effects, and for layouts which have significant differences to the US keyboard layout it is most likely unusable.

It is possible to install both the Oracle VM VirtualBox Extension Pack and VNC, but only one VRDE module can be active at any time. The following command switches to the VNC VRDE module in VNC:

```
VBoxManage setproperty vrdeextpack VNC
```

Configuring the remote access works very similarly to VRDP, see chapter 7.1, *Remote Display (VRDP Support)*, page 110, with some limitations. VNC does not support specifying several port numbers, and the authentication is done differently. VNC can only deal with password authentication, and there is no option to use password hashes. This leaves no other choice than having a clear-text password in the VM configuration, which can be set with the following command:

```
VBoxManage modifyvm "VM name" --vrdeproperty VNCPassword=secret
```

The user is responsible for keeping this password secret, and it should be removed when a VM configuration is passed to another person, for whatever purpose. Some VNC servers claim to have encrypted passwords in the configuration. This is not true encryption, it is only concealing the passwords, which is only as secure as using clear-text passwords.

The following command switches back to VRDP, if installed:

```
VBoxManage setproperty vrdeextpack "Oracle VM VirtualBox Extension Pack"
```

## 9.23 Starting Virtual Machines During System Boot

You can start VMs automatically during system boot on Linux, Oracle Solaris, and Mac OS X platforms for all users.

### 9.23.1 Linux: Starting the Autostart Service With init

On Linux, the autostart service is activated by setting two variables in `/etc/default/virtualbox`. The first one is `VBOXAUTOSTART_DB` which contains an absolute path to the autostart database directory. The directory should have write access for every user who should be able to start virtual machines automatically. Furthermore the directory should have the sticky bit set. The second variable is `VBOXAUTOSTART_CONFIG` which points the service to the autostart configuration file which is used during boot to determine whether to allow individual users to start a VM automatically and configure startup delays. The configuration file can be placed in `/etc/vbox` and contains several options. One is `default_policy` which controls whether the autostart service allows or denies to start a VM for users which are not in the exception list. The exception list starts with `exception_list` and contains a comma separated list with usernames. Furthermore a separate startup delay can be configured for every user to avoid overloading the host. A sample configuration is given below:

```
# Default policy is to deny starting a VM, the other option is "allow".
default_policy = deny

# Bob is allowed to start virtual machines but starting them
# will be delayed for 10 seconds
bob = {
    allow = true
    startup_delay = 10
}

# Alice is not allowed to start virtual machines, useful to exclude certain users
# if the default policy is set to allow.
alice = {
    allow = false
}
```

Every user who wants to enable autostart for individual machines has to set the path to the autostart database directory with the following command:

```
VBoxManage setproperty autostartdbpath <Autostart directory>
```

### 9.23.2 Oracle Solaris: Starting the Autostart Service With SMF

On Oracle Solaris hosts, the Oracle VM VirtualBox autostart daemon is integrated into the SMF framework. To enable it you have to point the service to an existing configuration file which has the same format as on Linux, see chapter 9.23.1, *Linux: Starting the Autostart Service With init*, page 248. For example:

```
svccfg -s svc:/application/virtualbox/autostart:default setprop \
  config/config=/etc/vbox/autostart.cfg
```

When everything is configured correctly you can start the Oracle VM VirtualBox autostart service with the following command:

```
svcadm enable svc:/application/virtualbox/autostart:default
```

For more information about SMF, please refer to the Oracle Solaris documentation.

### 9.23.3 Mac OS X: Starting the Autostart Service With launchd

On Mac OS X, launchd is used to start the Oracle VM VirtualBox autostart service. An example configuration file can be found in `/Applications/VirtualBox.app/Contents/MacOS/org.virtualbox.vboxauto` To enable the service copy the file to `/Library/LaunchDaemons` and change the `Disabled` key from `true` to `false`. Furthermore replace the second parameter to an existing configuration file which has the same format as on Linux, see chapter 9.23.1, *Linux: Starting the Autostart Service With init*, page 248.

To manually start the service use the following command:

```
launchctl load /Library/LaunchDaemons/org.virtualbox.vboxautostart.plist
```

For additional information on how launchd services can be configured see:
`http://developer.apple.com/mac/library/documentation/MacOSX/Conceptual/BPSystemStartup/BPSystemStartup.html`.

## 9.24 Oracle VM VirtualBox Expert Storage Management

In case the snapshot model of Oracle VM VirtualBox is not sufficient it is possible to enable a special mode which makes it possible to reconfigure storage attachments while the VM is paused. The user has to make sure that the disk data stays consistent to the guest because unlike with hotplugging the guest is not informed about detached or newly attached media.

The expert storage management mode can be enabled per VM executing:

```
VBoxManage setextradata "VM name" "VBoxInternal2/SilentReconfigureWhilePaused" 1
```

Storage attachments can be reconfigured while the VM is paused afterwards using:

```
VBoxManage storageattach ...
```

## 9.25 Handling of Host Power Management Events

Some host power management events are handled by Oracle VM VirtualBox. The actual behavior depends on the platform:

- **Host Suspends.** This event is generated when the host is about to suspend, that is, the host saves the state to some non-volatile storage and powers off.

  This event is currently only handled on Windows hosts and Mac OS X hosts. When this event is generated, Oracle VM VirtualBox will pause all running VMs.

- **Host Resumes.** This event is generated when the host woke up from the suspended state.

  This event is currently only handled on Windows hosts and Mac OS X hosts. When this event is generated, Oracle VM VirtualBox will resume all VMs which are where paused before.

- **Battery Low.** The battery level reached a critical level, usually less than 5 percent charged.

  This event is currently only handled on Windows hosts and Mac OS X hosts. When this event is generated, Oracle VM VirtualBox will save the state and terminate all VMs in preparation of a potential host powerdown.

  The behavior can be configured. By executing the following command, no VM is saved:

  ```
  VBoxManage setextradata global "VBoxInternal2/SavestateOnBatteryLow" 0
  ```

This is a global setting as well as a per-VM setting. The per-VM value has higher precedence than the global value. The following command will save the state of all VMs but will not save the state of VM “foo”:

```
VBoxManage setextradata global "VBoxInternal2/SavestateOnBatteryLow" 1
VBoxManage setextradata "foo" "VBoxInternal2/SavestateOnBatteryLow" 0
```

The first line is actually not required as by default the savestate action is performed.

## 9.26 Passing Through SSE4.1/SSE4.2 Instructions

To provide SSE 4.1/SSE 4.2 support to guests, the host CPU has to implement these instruction sets. The instruction sets are exposed to guests by default, but it is possible to disable the instructions for certain guests by using the following commands:

```
VBoxManage setextradata "VM name" VBoxInternal/CPUM/IsaExts/SSE4.1 0
VBoxManage setextradata "VM name" VBoxInternal/CPUM/IsaExts/SSE4.2 0
```

These are per-VM settings which are enabled by default.

## 9.27 Support for Keyboard Indicator Synchronization

This feature makes the host keyboard indicators (LEDs) match those of the VM’s emulated keyboard when the machine window is active. It is currently implemented for Mac OS X and Windows hosts. This feature is enabled by default on supported host OSes. You can disable this feature by running the following command:

```
VBoxManage setextradata "VM name" GUI/HidLedsSync 0
```

This is a per-VM setting, which is enabled by default.

## 9.28 Capturing USB Traffic for Selected Devices

You can capture USB traffic for single USB devices or on the root hub level, which captures the traffic of all USB devices attached to the root hub. Oracle VM VirtualBox stores the traffic in a format which is compatible with Wireshark. To capture the traffic of a specific USB device it must be attached to the VM with `VBoxManage` using the following command:

```
VBoxManage controlvm "VM name" usbattach "device uuid|address" --capturefile "filename"
```

In order to enable capturing on the root hub use the following command while the VM is not running:

```
VBoxManage setextradata "VM name" \
  VBoxInternal/Devices/usb-ehci/0/LUN#0/Config/CaptureFilename "filename"
```

The command above enables capturing on the root hub attached to the EHCI controller. To enable it for the OHCI or XHCI controller replace `usb-ehci` with `usb-ohci` or `usb-xhci` respectively.

## 9.29 Configuring the Heartbeat Service

Oracle VM VirtualBox ships a simple heartbeat service. Once the Guest Additions are active, the guest sends frequent heartbeat pings to the host. If the guest stops sending the heartbeat pings without properly terminating the service, the VM process will log this event in the VBox.log file. In the future it might be possible to configure dedicated actions but for now there is only a warning in the log file.

There are two parameters to configure. The *heartbeat interval* defines the time between two heartbeat pings. The default value is 2 seconds, that is, the heartbeat service of the Oracle VM VirtualBox Guest Additions will send a heartbeat ping every two seconds. The value in nanoseconds can be configured like this:

```
VBoxManage setextradata "VM name"\
  VBoxInternal/Devices/VMMDev/0/Config/HeartbeatInterval 2000000000
```

The *heartbeat timeout* defines the time the host waits starting from the last heartbeat ping before it defines the guest as unresponsive. The default value is 2 times the heartbeat interval (4 seconds) and can be configured as following, in nanoseconds:

```
VBoxManage setextradata "VM name" \
  VBoxInternal/Devices/VMMDev/0/Config/HeartbeatTimeout 4000000000
```

If the heartbeat timeout expires, there will be a log message like *VMMDev: HeartBeatCheck-Timer: Guest seems to be unresponsive. Last heartbeat received 5 seconds ago.* If another heartbeat ping arrives after this warning, there will be a log message like *VMMDev: GuestHeartBeat: Guest is alive.*

## 9.30 Encryption of Disk Images

Oracle VM VirtualBox enables you to transparently encrypt the data stored in hard disk images for the guest. It does not depend on a specific image format to be used. Images which have the data encrypted are not portable between Oracle VM VirtualBox and other virtualization software.

Oracle VM VirtualBox uses the AES algorithm in XTS mode and supports 128-bit or 256-bit data encryption keys (DEK). The DEK is stored encrypted in the medium properties and is decrypted during VM startup by entering a password which was chosen when the image was encrypted.

Since the DEK is stored as part of the VM configuration file, it is important that it is kept safe. Losing the DEK means that the data stored in the disk images is lost irrecoverably. Having complete and up to date backups of all data related to the VM is the responsibility of the user.

### 9.30.1 Limitations of Disk Encryption

There are some limitations the user needs to be aware of when using this feature:

- This feature is part of the Oracle VM VirtualBox Extension Pack, which needs to be installed. Otherwise disk encryption is unavailable.
- Since encryption works only on the stored user data, it is currently not possible to check for metadata integrity of the disk image. Attackers might destroy data by removing or changing blocks of data in the image or change metadata items such as the disk size.
- Exporting appliances which contain encrypted disk images is not possible because the OVF specification does not support this. All images are therefore decrypted during export.

- The DEK is kept in memory while the VM is running to be able to decrypt data read and encrypt data written by the guest. While this should be obvious the user needs to be aware of this because an attacker might be able to extract the key on a compromised host and decrypt the data.
- When encrypting or decrypting the images, the password is passed in clear text using the Oracle VM VirtualBox API. This needs to be kept in mind, especially when using third party API clients which make use of the webservice where the password might be transmitted over the network. The use of HTTPS is mandatory in such a case.
- Encrypting images with differencing images is only possible if there are no snapshots or a linear chain of snapshots. This limitation may be addressed in a future Oracle VM VirtualBox version.

### 9.30.2 Encrypting Disk Images

Encrypting disk images can be done either using the GUI or `VBoxManage`. While the GUI is easier to use, it works on a per VM basis and encrypts all disk images attached to the specific VM. With `VBoxManage` one can encrypt individual images, including all differencing images. To encrypt an unencrypted medium with `VBoxManage`, use:

```
VBoxManage encryptmedium "uuid|filename" \
  --newpassword "file|-" --cipher "cipher id" --newpasswordid "id"
```

To supply the encryption password point `VBoxManage` to the file where the password is stored or specify `-` to let VBoxManage ask you for the password on the command line.

The cipher parameter specifies the cipher to use for encryption and can be either `AES-XTS128-PLAIN64` or `AES-XTS256-PLAIN64`. The specified password identifier can be freely chosen by the user and is used for correct identification when supplying multiple passwords during VM startup.

If the user uses the same password when encrypting multiple images and also the same password identifier, the user needs to supply the password only once during VM startup.

### 9.30.3 Starting a VM with Encrypted Images

When a VM is started using the GUI, a dialog will open where the user needs to enter all passwords for all encrypted images attached to the VM. If another frontend like VBoxHeadless is used, the VM will be paused as soon as the guest tries to access an encrypted disk. The user needs to provide the passwords through `VBoxManage` using the following command:

```
VBoxManage controlvm "uuid|vmname" addencpassword "id" "password" [--removeonsuspend "yes|no"]
```

The `id` parameter must be the same as the password identifier supplied when encrypting the images. `password` is the password used when encrypting the images. The user can optionally specify `--removeonsuspend "yes|no"` to specify whether to remove the password from VM memory when the VM is suspended. Before the VM can be resumed, the user needs to supply the passwords again. This is useful when a VM is suspended by a host suspend event and the user does not want the password to remain in memory.

### 9.30.4 Decrypting Encrypted Images

In some circumstances it might be required to decrypt previously encrypted images. This can be done in the GUI for a complete VM or using `VBoxManage` with the following command:

```
VBoxManage encryptmedium "uuid|filename" --oldpassword "file|-"
```

The only required parameter is the password the image was encrypted with. The options are the same as for encrypting images.

## 9.31 Paravirtualized Debugging

In this section we cover debugging of guest operating systems using interfaces supported by paravirtualization providers.

> **Note:** Paravirtualized debugging significantly alter guest operating system behaviour and should only be used by expert users for debugging and diagnostics.

These debug options are specified as a string of key-value pairs separated by commas. An empty string disables paravirtualized debugging.

### 9.31.1 Hyper-V Debug Options

All of the options listed below are optional, and thus the default value specified will be used when the corresponding key-value pair is not specified.

- Key: **`enabled`**

  Value: `0` or `1`

  Default: `0`

  Specify `1` to enable the Hyper-V debug interface. If this key-value pair is not specified or the value is not `1`, the Hyper-V debug interface is disabled regardless of other key-value pairs being present.

- Key: **`address`**

  Value: IPv4 address

  Default: 127.0.0.1

  Specify the IPv4 address where the remote debugger is connected.

- Key: **`port`**

  Value: UDP port number

  Default: 50000

  Specify the UDP port number where the remote debugger is connected.

- Key: **`vendor`**

  Value: Hyper-V vendor signature reported by CPUID to the guest

  Default: When debugging is enabled: `Microsoft Hv`, otherwise: `VBoxVBoxVBox`

  Specify the Hyper-V vendor signature which is exposed to the guest by CPUID. For debugging Microsoft Windows guests, it is required the hypervisor reports the Microsoft vendor.

- Key: **`hypercallinterface`**

  Value: `0` or `1`

  Default: `0`

  Specify whether hypercalls should be suggested for initiating debug data transfers between host and guest rather than MSRs when requested by the guest.

- Key: **`vsinterface`**

  Value: `0` or `1`

  Default: When debugging is enabled, `1`, otherwise `0`

  Specify whether to expose the VS#1 virtualization service interface to the guest. This interface is required for debugging Microsoft Windows 10 32-bit guests, but is optional for other Windows versions.

#### 9.31.1.1 Setting up Windows Guests for Debugging with the Hyper-V Paravirtualization Provider

Windows supports debugging over a serial cable, USB, IEEE 1394 Firewire, and Ethernet. USB and IEEE 1394 are not applicable for virtual machines, and Ethernet requires Windows 8 or later. While a serial connection is universally usable, it is slow.

Debugging using the Hyper-V debug transport, supported on Windows Vista and later, offers significant benefits. It provides excellent performance due to direct host-to-guest transfers, it is easy to set up and requires minimal support from the hypervisor. It can be used with the debugger running on the same host as the VM or with the debugger and VM on separate machines connected over a network.

**Prerequisites**

- A VM configured for Hyper-V paravirtualization running a Windows Vista or newer Windows guest. You can check the effective paravirtualization provider for your VM with the output of the following `VBoxManage` command:

  ```
  VBoxManage showvminfo "VM name"
  ```

- A sufficiently up-to-date version of the Microsoft WinDbg debugger required to debug the version of Windows in your VM.

- While Windows 8 and newer Windows guests ship with Hyper-V debug support, Windows 7 and Vista do not. To use Hyper-V debugging with a Windows 7 or Vista guest, copy the file `kdvm.dll` from a Windows 8.0 installation. This file is typically located in `C:\Windows\System32`. Copy it to the same location in your Windows 7/Vista guest. Make sure you copy the 32-bit or 64-bit version of the DLL which matches your guest OS.

  > **Note:** Only Windows 8.0 ships `kdvm.dll`. Windows 8.1 and newer Windows versions do not.

**VM and Guest Configuration**

1. Power off the VM.
2. Enable the debug options with the following `VBoxManage` command:

   ```
   VBoxManage modifyvm "VM name" --paravirtdebug "enabled=1"
   ```

   The above command assumes your debugger will connect to your host machine on UDP port 50000. However, if you need to run the debugger on a remote machine you may specify the remote address and port here. For example:

   ```
   VBoxManage modifyvm "VM name" --paravirtdebug "enabled=1,address=192.168.32.1,port=55000"
   ```

   See chapter 9.31.1, *Hyper-V Debug Options*, page 253 for the complete set of options.
3. Start the VM.
4. In the guest, start an elevated command prompt and execute the following commands:
   - For a Windows 8 or newer Windows guest:

     ```
     bcdedit /dbgsettings net hostip:5.5.5.5 port:50000 key:1.2.3.4
     ```

   - For a Windows 7 or Vista guest:

```
bcdedit /set loadoptions host_ip=5.5.5.5,host_port=50000,encryption_key=1.2.3.4
bcdedit /set dbgtransport kdvm.dll
```

The IP address and port in the `bcdedit` command are ignored when using the Hyper-V debug transport. Any valid IP and a port number greater than 49151 and lower than 65536 can be entered.

The encryption key in the `bcdedit` command is relevant and must be valid. The key "1.2.3.4" used in the above example is valid and may be used if security is not a concern. If you do not specify any encryption key, `bcdedit` will generate one for you and you will need to copy this key to later enter in Microsoft WinDbg on the remote end. This encryption key is used to encrypt the debug data exchanged between Windows and the debugger.

- Run one or more of the following commands to enable debugging for the appropriate phase or component of your Windows guest:

```
bcdedit /set debug on
bcdedit /set bootdebug on
bcdedit /set {bootmgr} bootdebug on
```

Please note that the `bootdebug` options are only effective on Windows 8 or newer when using the Hyper-V debug transport. Refer to Microsoft Windows documentation for detailed explanation of `bcdedit` options.

5. Start Microsoft WinDbg on your host machine or remote host.

   From the **File** menu, select **Kernel Debug**. On the **NET** tab, specify the UDP port number you used in the `paravirtdebug` options. If you did not specify any, leave it as 50000. Ensure that the UDP port is not blocked by a firewall or other security software.

   In the **Key** field, enter `1.2.3.4` or the encryption key from the `bcdedit` command in your Windows guest.

   Click **OK** to start listening for connections. Microsoft WinDbg typically shows a Waiting to Reconnect message during this phase.

   Alternatively, to directly start a debug session, run WinDbg from the command line as follows :

```
windbg.exe -k net:port=50000,key=1.2.3.4
```

   See the WinDbg documentation for the complete command line syntax.

6. Reboot your Windows guest and it should then connect as a debuggee with Microsoft WinDbg.

## 9.32 PC Speaker Passthrough

As an experimental feature, primarily due to being limited to Linux host only and unknown Linux distribution coverage, Oracle VM VirtualBox supports passing through the PC speaker to the host. The PC speaker, sometimes called the system speaker, is a way to produce audible feedback such as beeps without the need for regular audio and sound card support.

The PC speaker passthrough feature in Oracle VM VirtualBox handles beeps only. Advanced PC speaker use by the VM, such as PCM audio, will not work, resulting in undefined host behavior.

Producing beeps on Linux is a very complex topic. Oracle VM VirtualBox offers a collection of options, in an attempt to make this work deterministically and reliably on as many Linux distributions and system configurations as possible. These are summarized in the following table.

| Code | Device | Notes |
|---|---|---|
| 1 | `/dev/input/by-path/platform-pcspkr-event-spkr` | Direct input device use. |
| 2 | `/dev/tty` | Uses the terminal association of the VM process. VM needs to be started on a virtual console. |
| 3 | `/dev/tty0` or `/dev/vc/0` | Can only be used by user `root` or users with `cap_sys_tty_config` capability. |
| 9 | A user-specified console or evdev device path. | As for codes 1 to 3, but with a custom device path. |
| 70 | `/dev/tty` | Standard beep only. Loses frequency and length. See code 2. |
| 79 | A user-specified terminal device path. | As for code 70, but with a custom device path. |
| 100 | All of the above. | Tries all the available codes. |

To enable PC speaker passthrough use the following command:

```
VBoxManage setextradata "VM name" "VBoxInternal/Devices/i8254/0/Config/PassthroughSpeaker" N
```

Replace `N` with the code representing the case you want to use. Changing this setting will take effect when the VM is started next. It is safe to enable PC speaker passthrough on all host OSes. It will only have an effect on Linux.

The VM log file, `VBox.log`, will contain lines with the prefix `PIT: speaker:` showing the PC speaker passthrough setup activities. It gives hints which device it picked or why it failed.

Enabling PC speaker passthrough for the VM is usually the simple part. The real difficulty is making sure that Oracle VM VirtualBox can access the necessary device, because in a typical Linux install most of them can only be accessed by user `root`. You should follow the preferred way to persistently change this, such as by referring to your distribution's documentation. Since there are countless Linux distribution variants, we can only give the general hints that there is often a way to give the X11 session user access to additional devices, or you need to find a working solution using a udev configuration file. If everything fails you might try setting the permissions using a script which is run late enough in the host system startup.

Sometimes additional rules are applied by the kernel to limit access. For example, that the VM process must have the same controlling terminal as the device configured to be used for beeping, something which is often very difficult to achieve for GUI applications such as Oracle VM VirtualBox. The table above contains some hints, but in general refer to the Linux documentation.

If you have trouble getting any beeps even if the device permissions are set up and VBox.log confirms that it uses evdev or console for the PC speaker control, check if your system has a PC speaker. Some systems do not have one. Other complications can arise from Linux rerouting the PC speaker output to a sound card. Check if the beeps are audible if you connect speakers to your sound card. Today almost all systems have one. Finally, check if the audio mixer control has a channel named "beep", which could be hidden in the mixer settings, and that it is not muted.

## 9.33 Accessing USB devices Exposed Over the Network with USB/IP

Oracle VM VirtualBox supports passing through USB devices which are exposed over the network using the USB over IP protocol without the need to configure the client side provided by the kernel and usbip tools. Furthermore, this feature works with Oracle VM VirtualBox running on any supported host, rather than just Linux alone, as is the case with the official client.

To enable support for passing through USB/IP devices, the device server exporting the devices must be added with the following command:

```
VBoxManage usbdevsource add "Unique name" --backend "USBIP" --address "Device server[:port]"
```

USB devices exported on the device server are then accessible through the GUI or `VBoxManage`, like any USB devices attached locally. This can be used multiple times to access different device servers.

To remove a device server, the following command can be used:

```
VBoxManage usbdevsource remove "Unique name"
```

### 9.33.1 Setting up USB/IP Support on a Linux System

This section gives a brief overview on how to set up a Linux based system to act as a USB device server. The system on the server requires that the `usbip-core.ko` and `usbip-host.ko` kernel drivers are available, and that the USB/IP tools package is installed. The particular installation method for the necessary tools depends on which distribution is used. For example, for Debian based systems, the following command should be used to install the required tools:

```
apt-get install usbip-utils
```

To check whether the necessary tools are already installed use the following command:

```
$ usbip list -l
```

This should produce output similar to that shown in the example below:

```
 - busid 4-2 (0bda:0301)
   Realtek Semiconductor Corp. : multicard reader (0bda:0301)

 - busid 5-1 (046d:c52b)
   Logitech, Inc. : Unifying Receiver (046d:c52b)
```

If everything is installed, the USB/IP server needs to be started as `root` using the following command:

```
usbipd -D
```

Refer to the documentation for the installed distribution to determine how to start the service when the system boots.

By default, no device on the server is exported. This must be done manually for each device. To export a device use the following command:

```
usbip bind -b "bus identifier"
```

To export the multicard reader in the previous example:

```
usbip bind -b 4-2
```

### 9.33.2 Security Considerations

The communication between the server and client is unencrypted and there is no authorization required to access exported devices. An attacker might sniff sensitive data or gain control over a device. To mitigate this risk, the device should be exposed over a local network to which only trusted clients have access. To access the device remotely over a public network, a VPN solution should be used to provide the required level of security protection.

## 9.34 Using Hyper-V with Oracle VM VirtualBox

Oracle VM VirtualBox can be used on a Windows host where Hyper-V is running. This is an experimental feature.

No configuration is required. Oracle VM VirtualBox detects Hyper-V automatically and uses Hyper-V as the virtualization engine for the host system. The CPU icon in the VM window status bar indicates that Hyper-V is being used.

> **Note:** When using this feature, you might experience significant Oracle VM VirtualBox performance degradation on some host systems.

## 9.35 Nested Virtualization

Oracle VM VirtualBox supports *nested virtualization* on host systems that run AMD CPUs. This feature enables the passthrough of hardware virtualization functions to the guest VM. That means that you can install a hypervisor, such as Oracle VM VirtualBox, Oracle VM Server or KVM, on an Oracle VM VirtualBox guest. You can then create and run VMs within the guest VM.

You can enable the nested virtualization feature in one of the following ways:

- From the VirtualBox Manager, select the **Enable Nested VT-x/AMD-V** check box on the **Processor** tab. To disable the feature, deselect the check box.
- Use the `--nested-hw-virt` option of the `VBoxManage modifyvm` command to enable or disable nested virtualization. See chapter 8.8, *VBoxManage modifyvm*, page 135.

## 9.36 VISO file format / RTIsoMaker

ISO image maker.

### Synopsis

```
RTIsoMaker [options] [@commands.rsp] <filespec...>
```

### Description

Construct a virtual ISO 9660 / Joliet / UDF / HFS hybrid image and either write it to a file (RTIsoMaker) or serve it as a virtual image (VISO).

#### VISO file format

A VISO file is a virtual ISO image, i.e. constructed in memory from a bunch of files on the host. A VISO is just the recipe describing how to go about this using a syntax vaguely similar to mkisofs and genisoimage.

One requirement is that the VISO file must start with one of the `--iprt-iso-maker-file-marker` options. Which of the options you use will dictate the quoting and escaping rules used when reading the file. The option takes the image UUID as an argument.

The VISO files are treated as UTF-8 and must not contain any byte order marker (BOM). There is currently no way to comment out lines in a VISO file.

### File specifications and –name-setup

All non-options that does not start with '@' are taken to indicate a file, directory, or similar that is should be added to the ISO image. Directories are added recursively and content is subject to filtering options.

Since there can be up to six different namespaces on an ISO, it is handy to be able to control the names used in each and be able to exclude an object from one or more namespaces. The `--name-setup` option specifies the file specification format to use forthwith.

The default setup is:

```
--name-setup iso+joliet+udf+hfs
```

Which means you specify one on-ISO name for all namespaces followed by '=' and the source file system name. Only specifying the source file system will add the file/dir/whatever to the root of the ISO image.

Lets look at the following two examples:

```
/docs/readme.txt=/home/user/Documents/product-x-readme.txt
/home/user/Documents/product-x-readme.txt
```

In the first case the file `'/home/user/Documents/product-x-readme.txt'` is added to the ISO image as `'/docs/readme.txt'` in all enabled namespaces. In the primary ISO 9660 namespace, the filename will by default be converted to upper case because it's required by the spec.

In the second case the file is added to the root under the name `'product-x-readme.txt'` in all namespaces. Though, in the primary ISO 9660 namespace the name will be transformed to apply with the current ISO level, probably uppercased, possibly truncated too.

Given `--name-setup iso,joliet,udf` you can specify the name individually for each of the three namespace, if you like. If you omit any, they will use last name given. Any names left blank (==) will be considered omitted.

A different name in each namespace:

```
/ISO.TXT=/Joliet.TxT=/UDF.txt=/tmp/iso/real.txt
```

Specific name in the ISO 9660 namespace, same in the rest:

```
/ISO.TXT=/OtherNamespaces.TxT=/tmp/iso/real.txt
```

Omit the file from the ISO 9660 namespace:

```
=/OtherNamespaces.TxT=/tmp/iso/real.txt
```

Omit the file from the joliet namespace:

```
/ISO.TXT==/UDF.TxT=/tmp/iso/real.txt
```

Use the same filename as the source everywhere:

```
/tmp/iso/real.txt
```

Using for instance `--name-setup udf` you can add a files/dirs/whatever to select namespace(s) without the more complicated empty name syntax above.

When adding directories, you can only control the naming and omitting of the directory itself, not any recursively added files and directories below it.

## Options

### General

**`-o <output-file>`**
**`--output=<output-file>`**
The output filename. This option is not supported in VISO mode.

**`--name-setup <spec>`**
Configures active namespaces and how file specifications are to be interpreted. The specification is a comma separated list. Each element in the list is a sub-list separated by space, `'+'` or `'|'` giving the namespaces that elements controls. Namespaces are divied into two major and minor ones, you cannot specifying a minor before the major it belongs to.

Major namespaces and aliases in parentheses:

- iso (primary, iso9660, iso-9660, primary-iso, iso-primary)
- joliet
- udf
- hfs (hfs-plus)

Minor namespaces:

- rock: rock ridge on previous major namespace (iso / joliet)
- iso-rock: rock ridge extensions on primary ISO 9660 namespace
- joliet-rock: rock ridge on joliet namespace (just for fun)
- trans-tbl: translation table file on previous major namespace
- iso-trans-tbl
- joliet-trans-tbl
- udf-trans-tbl
- hfs-trans-tbl

**--push-iso=<*iso-file*>**
**--push-iso-no-joliet=<*iso-file*>**
**--push-iso-no-rock-<*iso-file*>**
**--push-iso-no-rock-no-joliet=<*iso-file*>**
Open the specified ISO file and use it as source file system until the corresponding `--pop` options is encountered. The variations are for selecting which namespace on the ISO to (not) access. These options are handy for copying files/directories/stuff from an ISO without having to extract them first or using the `:iprtvfs:` syntax.

**--pop**
Pops a `--push-iso` of the source file system stack.

**--import-iso=<*iso-file*>**
Imports everything on the given ISO file, including boot configuration and system area (first 16 sectors) content. You can use `--name-setup` to omit namespaces.

### Namespaces

**--iso-level=<*0|1|2|3*>**
Sets the ISO level:

- 0: Disable primary ISO namespace.
- 1: ISO level 1: Filenames 8.3 format and limited to 4GB - 1.
- 2: ISO level 2: 31 char long names and limited to 4GB - 1.
- 3: ISO level 3: 31 char long names and support for >=4GB files. (default)
- 4: Fictive level used by other tools. Not yet implemented.

**--rock-ridge**
**--limited-rock-ridge**
**--no-rock-ridge**
Enables or disables rock ridge support for the primary ISO 9660 namespace. The `--limited-rock-ridge` option omits a couple of bits in the root directory that would make Linux pick rock ridge over joliet.

Default: `--limited-rock-ridge`

**-J**
**--joliet**
**--no-joliet**
Enables or disable the joliet namespace. This option must precede any file specifications.

Default: `--joliet`

**--joliet-ucs-level=<*1|2|3*>**
**--ucs-level=<*1|2|3*>**
Set the Joliet UCS support level. This is currently only flagged in the image but not enforced on the actual path names.

Default level: 3

### File Attributes

**--rational-attribs**
Enables rational file attribute handling (default):

- Owner ID is set to zero
- Group ID is set to zero
- Mode is set to 0444 for non-executable files.
- Mode is set to 0555 for executable files.
- Mode is set to 0555 for directories, preserving stick bits.

**--strict-attribs**
Counters `--rational-attribs` and causes attributes to be recorded exactly as they appear in the source.

**--file-mode=<*mode*>**
**--no-file-mode**
Controls the forced file mode mask for rock ridge, UDF and HFS.

**--dir-mode=<*mode*>**
**--no-dir-mode**
Controls the forced directory mode mask for rock ridge, UDF and HFS.

**--new-dir-mode=<*mode*>**
Controls the default mode mask (rock ridge, UDF, HFS) for directories that are created implicitly. The `--dir-mode` option overrides this.

**--chmod=<*mode*>:<*on-iso-file*>**
Explictily sets the rock ridge, UDF and HFS file mode for a file/dir/whatever that has already been added to the ISO. The mode can be octal, `ra+x`, `a+r`, or `a+rx`. (Support for more complicated mode specifications may be implemented at a later point.)

Note that only namespaces in the current –name-setup are affected.

**--chown=<*owner-id*>:<*on-iso-file*>**
Explictily sets the rock ridge, UDF and HFS file owner ID (numeric) for a file/dir/whatever that has already been added to the ISO.

Note that only namespaces in the current –name-setup are affected.

**–chgrp=*group-id*:*on-iso-file***
Explictily sets the rock ridge, UDF and HFS file group ID (numeric) for a file/dir/whatever that has already been added to the ISO.

Note that only namespaces in the current –name-setup are affected.

### Booting

**--eltorito-new-entry**
**--eltorito-alt-boot**
Starts a new El Torito boot entry.

**--eltorito-add-image=<*filespec*>**
File specification of a file that should be added to the image and used as the El Torito boot image of the current boot entry.

**-b <*on-iso-file*>**
**--eltorito-boot=<*on-iso-file*>**
Specifies a file on the ISO as the El Torito boot image for the current boot entry.

**--eltorito-floppy-12**
**--eltorito-floppy-144**
**--eltorito-floppy-288**
**--no-emulation-boot**
**--hard-disk-boot**
Sets the boot image emulation type of the current El Torito boot entry.

**--boot-load-seg=<*seg*>**
Specify the image load segment for the current El Torito boot entry.

Default: 0x7c0

**--boot-load-size=<*sectors*>**
Specify the image load size in emulated sectors for the current El Torito boot entry.

Default: 4 (sectors of 512 bytes)

**--no-boot**
Indicates that the current El Torito boot entry isn't bootable. (The BIOS will allegedly configure the emulation, but not attempt booting.)

**--boot-info-table**
Write a isolinux/syslinux boot info table into the boot image for the current El Torito boot entry.

**--eltorito-platform-id=<*id*>**
Set the El Torito platform ID of the current entry, a new entry of the verification entry depending on when it's used. The ID must be one of: `x86`, `PPC`, `Mac`, `efi`

**-c <*namespec*>**
**--boot-catalog=<*namespec*>**
Enters the El Torito boot catalog into the namespaces as a file. The *namespec* uses the same format as a 'filespec', but omits the final source file system name component.

**-G <*file*>**
**--generic-boot=<*file*>**
Specifies a file that should be loaded at offset 0 in the ISO image. The file must not be larger than 32KB. When creating a hybrid image, parts of this may be regenerated by partition tables and such.

### String properties (applied to active namespaces only)

**--abstract=<*file-id*>**
The name of the abstract file in the root dir.

**-A <*text*|*_file-id*>**
**--application-id=<*text*|*_file-id*>**
Application ID string or root file name. The latter must be prefixed with an underscore.

**--biblio=<*file-id*>**
The name of the bibliographic file in the root dir.

**--copyright=<*file-id*>**
The name of the copyright file in the root dir.

**-P <*text*|*_file-id*>**
**--publisher=<*text*|*_file-id*>**
Publisher ID string or root file name. The latter must be prefixed with an underscore.

**-p <*text*|*_file-id*>**
**--preparer=<*text*|*_file-id*>**
Data preparer ID string or root file name. The latter must be prefixed with an underscore.

**--sysid=<*text*>**
System ID string.

**--volid=<*text*>**
**--volume-id=<*text*>**
Volume ID string (label). (It is possible to set different labels for primary ISO 9660, joliet, UDF and HFS by changing the active namespaces using the `--name-setup` option between `--volume-id` occurences.)

**--volset=<*text*>**
Volume set ID string.

### Compatibility:

**--graft-points**
Alias for –name-setup iso+joliet+udf+hfs.

**-l**
**--long-names**
Allow 31 charater filenames. Just ensure ISO level >= 2 here.

**-R**
**--rock**
Same as `--rock-ridge` and `--strict-attribs`.

**-r**
**--rational-rock**
Same as `--rock-ridge` and `--rational-attribs`.

### VISO Specific:

**--iprt-iso-maker-file-marker=<*UUID*>**
**--iprt-iso-maker-file-marker-bourne=<*UUID*>**
**--iprt-iso-maker-file-marker-bourne-sh=<*UUID*>**
Used as first option in a VISO file to specify the file UUID and that it is formatted using bourne-shell argument quoting & escaping style.

**--iprt-iso-maker-file-marker-ms=<*UUID*>**
**--iprt-iso-maker-file-marker-ms-sh=<*UUID*>**
Used as first option in a VISO file to specify the file UUID and that it is formatted using microsoft CRT argument quoting & escaping style.

### Testing (not applicable to VISO):

**--output-buffer-size=<*bytes*>**
Selects a specific output buffer size for testing virtual image reads.

**--random-output-buffer-size**
Enables randomized buffer size for each virtual image read, using the current output buffer size (`--output-buffer-size`) as maximum.

**--random-order-verification=<*size*>**
: Enables verification pass of the image that compares blocks of the given size in random order from the virtual and output images.

# 10 Technical Background

This chapter provides additional information for readers who are familiar with computer architecture and technology and wish to find out more about how Oracle VM VirtualBox works *under the hood*. The contents of this chapter are not required reading in order to use Oracle VM VirtualBox successfully.

## 10.1 Where Oracle VM VirtualBox Stores its Files

In Oracle VM VirtualBox, a virtual machine and its settings are described in a virtual machine settings file in XML format. In addition, most virtual machine have one or more virtual hard disks, which are typically represented by disk images, such as those in VDI format. Where all these files are stored depends on which version of Oracle VM VirtualBox created the machine.

### 10.1.1 Machines Created by Oracle VM VirtualBox Version 4.0 or Later

By default, each virtual machine has one directory on your host computer where all the files of that machine are stored: the XML settings file, with a `.vbox` file extension, and its disk images.

By default, this *machine folder* is placed in a common folder called `VirtualBox VMs`, which Oracle VM VirtualBox creates in the current system user's home directory. The location of this home directory depends on the conventions of the host operating system, as follows:

- On Windows, this is the location returned by the `SHGetFolderPath` function of the Windows system library Shell32.dll, asking for the user profile. On very old Windows versions which do not have this function or where it unexpectedly returns an error, there is a fallback based on environment variables. First, `%USERPROFILE%` is checked. If it does not exist then an attempt with `%HOMEDRIVE%%HOMEPATH%` is made. A typical location is `C:\Users\username`.
- On Linux, Mac OS X, and Oracle Solaris, this is generally taken from the environment variable `$HOME`, except for the user `root` where it is taken from the account database. This is a workaround for the frequent trouble caused by users using Oracle VM VirtualBox in combination with the tool `sudo` which by default does not reset the environment variable `$HOME`. A typical location on Linux and Oracle Solaris is `/home/username` and on Mac OS X `/Users/username`.

For simplicity, we will abbreviate the location of the home directory as `$HOME`. Using that convention, the common folder for all virtual machines is `$HOME/VirtualBox VMs`.

As an example, when you create a virtual machine called "Example VM", Oracle VM VirtualBox creates the following:

- A machine folder `$HOME/VirtualBox VMs/Example VM/`
- In the machine folder, a settings file: `Example VM.vbox`
- In the machine folder, a virtual disk image: `Example VM.vdi`.

This is the default layout if you use the **Create New Virtual Machine** wizard described in chapter 1.8, *Creating Your First Virtual Machine*, page 8. Once you start working with the VM,

additional files are added. Log files are in a subfolder called `Logs`, and if you have taken snapshots, they are in a `Snapshots` subfolder. For each VM, you can change the location of its snapshots folder in the VM settings.

You can change the default machine folder by selecting **Preferences** from the **File** menu in the Oracle VM VirtualBox main window. Then, in the displayed window, click on the **General** tab. Alternatively, use `VBoxManage setproperty machinefolder`. See chapter 8.31, *VBoxManage setproperty*, page 172.

### 10.1.2 Machines Created by Oracle VM VirtualBox Versions Before 4.0

If you have upgraded to Oracle VM VirtualBox 4.0 from an earlier version of Oracle VM VirtualBox, you probably have settings files and disks in the earlier file system layout.

Before version 4.0, Oracle VM VirtualBox separated the machine settings files from virtual disk images. The machine settings files had an `.xml` file extension and resided in a folder called `Machines` under the global Oracle VM VirtualBox configuration directory. See chapter 10.1.3, *Global Configuration Data*, page 266. On Linux, for example, this was the hidden directory `$HOME/.VirtualBox/Machines`. The default hard disks folder was called `HardDisks` and was also located in the `.VirtualBox` folder. Both locations could be changed by the user in the global preferences. The concept of a default hard disk folder was abandoned with Oracle VM VirtualBox 4.0, since disk images now reside in each machine's folder by default.

The old layout had the following severe disadvantages:

- It was very difficult to move a virtual machine from one host to another because the files involved did not reside in the same folder. In addition, the virtual media of all machines were registered with a global registry in the central Oracle VM VirtualBox settings file, `$HOME/.VirtualBox/VirtualBox.xml`.

  To move a machine to another host, it was therefore not enough to move the XML settings file and the disk images, which were in different locations, but the hard disk entries from the global media registry XML had to be meticulously copied as well. This was close to impossible if the machine had snapshots and therefore differencing images.

- Storing virtual disk images, which can grow very large, under the hidden `.VirtualBox` directory, at least on Linux and Oracle Solaris hosts, made many users wonder where their disk space had gone.

Whereas new VMs created with Oracle VM VirtualBox 4.0 or later will conform to the new layout, for maximum compatibility, old VMs are *not* converted to the new layout. Otherwise machine settings would be irrevocably broken if a user downgraded from 4.0 back to an older version of Oracle VM VirtualBox.

### 10.1.3 Global Configuration Data

In addition to the files of the virtual machines, Oracle VM VirtualBox maintains global configuration data in the following directory:

- **Linux and Oracle Solaris:** `$HOME/.config/VirtualBox`.

  `$HOME/.VirtualBox` is used if it exists, for compatibility with legacy versions before Oracle VM VirtualBox 4.3.

- **Windows:** `$HOME/.VirtualBox`.

- **Mac OS X:** `$HOME/Library/VirtualBox`.

Oracle VM VirtualBox creates this configuration directory automatically, if necessary. Optionally, you can specify an alternate configuration directory by setting the `VBOX_USER_HOME` environment variable, or additionally on Linux or Oracle Solaris by using the standard `XDG_CONFIG_HOME` variable. Since the global `VirtualBox.xml` settings file points to all other configuration files, this enables switching between several Oracle VM VirtualBox configurations.

Most importantly, in this directory, Oracle VM VirtualBox stores its global settings file, another XML file called `VirtualBox.xml`. This includes global configuration options and the list of registered virtual machines with pointers to their XML settings files. Neither the location of this file nor its directory has changed with Oracle VM VirtualBox 4.0.

Before Oracle VM VirtualBox 4.0, all virtual media, such as disk image files, were also contained in a global registry in this settings file. For compatibility, this media registry still exists if you upgrade Oracle VM VirtualBox and there are media from machines which were created with a version before 4.0. If you have no such machines, then there will be no global media registry. With Oracle VM VirtualBox 4.0, each machine XML file has its own media registry.

Also before Oracle VM VirtualBox 4.0, the default `Machines` folder and the default `HardDisks` folder resided under the Oracle VM VirtualBox configuration directory, such as `$HOME/.VirtualBox/Machines` on Linux. If you are upgrading from an Oracle VM VirtualBox version before 4.0, files in these directories are not automatically moved in order not to break backwards compatibility.

### 10.1.4 Summary of 4.0 Configuration Changes

The following table gives a brief overview of the configuration changes between legacy versions and version 4.0 or later.

| **Setting** | **Before 4.0** | **4.0 or above** |
|---|---|---|
| Default machines folder | `$HOME/.VirtualBox/Machines` | `$HOME/VirtualBox VMs` |
| Default disk image location | `$HOME/.VirtualBox/HardDisks` | In each machine's folder |
| Machine settings file extension | `.xml` | `.vbox` |
| Media registry | Global `VirtualBox.xml` file | Each machine settings file |
| Media registration | Explicit open/close required | Automatic on attach |

### 10.1.5 Oracle VM VirtualBox XML Files

Oracle VM VirtualBox uses XML for both the machine settings files and the global configuration file, `VirtualBox.xml`.

All Oracle VM VirtualBox XML files are versioned. When a new settings file is created, for example because a new virtual machine is created, Oracle VM VirtualBox automatically uses the settings format of the current Oracle VM VirtualBox version. These files may not be readable if you downgrade to an earlier version of Oracle VM VirtualBox. However, when Oracle VM VirtualBox encounters a settings file from an earlier version, such as after upgrading Oracle VM VirtualBox, it attempts to preserve the settings format as much as possible. It will only silently upgrade the settings format if the current settings cannot be expressed in the old format, for example because you enabled a feature that was not present in an earlier version of Oracle VM VirtualBox.

As an example, before Oracle VM VirtualBox 3.1, it was only possible to enable or disable a single DVD drive in a virtual machine. If it was enabled, then it would always be visible as

the secondary master of the IDE controller. With Oracle VM VirtualBox 3.1, DVD drives can be attached to arbitrary slots of arbitrary controllers, so they could be the secondary slave of an IDE controller or in a SATA slot. If you have a machine settings file from an earlier version and upgrade Oracle VM VirtualBox to 3.1 and then move the DVD drive from its default position, this cannot be expressed in the old settings format; the XML machine file would get written in the new format, and a backup file of the old format would be kept.

In such cases, Oracle VM VirtualBox backs up the old settings file in the virtual machine's configuration directory. If you need to go back to the earlier version of Oracle VM VirtualBox, then you will need to manually copy these backup files back.

We intentionally do not document the specifications of the Oracle VM VirtualBox XML files, as we must reserve the right to modify them in the future. We therefore strongly suggest that you do not edit these files manually. Oracle VM VirtualBox provides complete access to its configuration data through its the `VBoxManage` command line tool, see chapter 8, *VBoxManage*, page 120 and its API, see chapter 11, *Oracle VM VirtualBox Programming Interfaces*, page 276.

## 10.2 Oracle VM VirtualBox Executables and Components

Oracle VM VirtualBox was designed to be modular and flexible. When the Oracle VM VirtualBox graphical user interface (GUI) is opened and a VM is started, at least the following three processes are running:

- `VBoxSVC`, the Oracle VM VirtualBox service process which always runs in the background. This process is started automatically by the first Oracle VM VirtualBox client process and exits a short time after the last client exits. The first Oracle VM VirtualBox service can be the GUI, `VBoxManage`, `VBoxHeadless`, the web service amongst others. The service is responsible for bookkeeping, maintaining the state of all VMs, and for providing communication between Oracle VM VirtualBox components. This communication is implemented using COM/XPCOM.

> **Note:** When we refer to *clients* here, we mean the local clients of a particular `VBoxSVC` server process, not clients in a network. Oracle VM VirtualBox employs its own client/server design to allow its processes to cooperate, but all these processes run under the same user account on the host operating system, and this is totally transparent to the user.

- The GUI process, `VirtualBoxVM`, a client application based on the cross-platform Qt library. When started without the `--startvm` option, this application acts as the VirtualBox Manager, displaying the VMs and their settings. It then communicates settings and state changes to `VBoxSVC` and also reflects changes effected through other means, such as the `VBoxManage` command.
- If the `VirtualBoxVM` client application is started with the `--startvm` argument, it loads the VMM library which includes the actual hypervisor and then runs a virtual machine and provides the input and output for the guest.

Any Oracle VM VirtualBox front-end, or client, will communicate with the service process and can both control and reflect the current state. For example, either the VM selector or the VM window or VBoxManage can be used to pause the running VM, and other components will always reflect the changed state.

The Oracle VM VirtualBox GUI application is only one of several available front ends, or clients. The complete list shipped with Oracle VM VirtualBox is as follows:

- `VirtualBoxVM`: The Qt front end implementing the VirtualBox Manager and running VMs.

- `VBoxManage`: A less user-friendly but more powerful alternative. See chapter 8, *VBoxManage*, page 120.
- `VBoxHeadless`: A VM front end which does not directly provide any video output and keyboard or mouse input, but enables redirection through the VirtualBox Remote Desktop Extension. See chapter 7.1.2, *VBoxHeadless, the Remote Desktop Server*, page 112.
- `vboxwebsrv`: The Oracle VM VirtualBox web service process which enables control of an Oracle VM VirtualBox host remotely. This is described in detail in the Oracle VM VirtualBox Software Development Kit (SDK) reference. See chapter 11, *Oracle VM VirtualBox Programming Interfaces*, page 276.
- The Oracle VM VirtualBox Python shell: A Python alternative to `VBoxManage`. This is also described in the SDK reference.

Internally, Oracle VM VirtualBox consists of many more or less separate components. You may encounter these when analyzing Oracle VM VirtualBox internal error messages or log files. These include the following:

- IPRT: A portable runtime library which abstracts file access, threading, and string manipulation. Whenever Oracle VM VirtualBox accesses host operating features, it does so through this library for cross-platform portability.
- VMM (Virtual Machine Monitor): The heart of the hypervisor.
- EM (Execution Manager): Controls execution of guest code.
- REM (Recompiled Execution Monitor): Provides software emulation of CPU instructions.
- TRPM (Trap Manager): Intercepts and processes guest traps and exceptions.
- HM (Hardware Acceleration Manager): Provides support for VT-x and AMD-V.
- GIM (Guest Interface Manager): Provides support for various paravirtualization interfaces to the guest.
- PDM (Pluggable Device Manager): An abstract interface between the VMM and emulated devices which separates device implementations from VMM internals and makes it easy to add new emulated devices. Through PDM, third-party developers can add new virtual devices to Oracle VM VirtualBox without having to change Oracle VM VirtualBox itself.
- PGM (Page Manager): A component that controls guest paging.
- PATM (Patch Manager): Patches guest code to improve and speed up software virtualization.
- TM (Time Manager): Handles timers and all aspects of time inside guests.
- CFGM (Configuration Manager): Provides a tree structure which holds configuration settings for the VM and all emulated devices.
- SSM (Saved State Manager): Saves and loads VM state.
- VUSB (Virtual USB): A USB layer which separates emulated USB controllers from the controllers on the host and from USB devices. This component also enables remote USB.
- DBGF (Debug Facility): A built-in VM debugger.

- Oracle VM VirtualBox emulates a number of devices to provide the hardware environment that various guests need. Most of these are standard devices found in many PC compatible machines and widely supported by guest operating systems. For network and storage devices in particular, there are several options for the emulated devices to access the underlying hardware. These devices are managed by PDM.

- Guest Additions for various guest operating systems. This is code that is installed from within a virtual machine. See chapter 4, *Guest Additions*, page 62.

- The "Main" component is special. It ties all the above bits together and is the only public API that Oracle VM VirtualBox provides. All the client processes listed above use only this API and never access the hypervisor components directly. As a result, third-party applications that use the Oracle VM VirtualBox Main API can rely on the fact that it is always well-tested and that all capabilities of Oracle VM VirtualBox are fully exposed. It is this API that is described in the Oracle VM VirtualBox SDK. See chapter 11, *Oracle VM VirtualBox Programming Interfaces*, page 276.

## 10.3 Hardware vs. Software Virtualization

Oracle VM VirtualBox enables software in the virtual machine to run directly on the processor of the host, but an array of complex techniques is employed to intercept operations that would interfere with your host. Whenever the guest attempts to do something that could be harmful to your computer and its data, Oracle VM VirtualBox steps in and takes action. In particular, for lots of hardware that the guest believes to be accessing, Oracle VM VirtualBox simulates a certain "virtual" environment according to how you have configured a virtual machine. For example, when the guest attempts to access a hard disk, Oracle VM VirtualBox redirects these requests to whatever you have configured to be the virtual machine's virtual hard disk. This is normally an image file on your host.

Unfortunately, the x86 platform was never designed to be virtualized. Detecting situations in which Oracle VM VirtualBox needs to take control over the guest code that is executing, as described above, is difficult. There are two ways in which to achieve this:

- Since 2006, Intel and AMD processors have had support for so-called *hardware virtualization*. This means that these processors can help Oracle VM VirtualBox to intercept potentially dangerous operations that a guest operating system may be attempting and also makes it easier to present virtual hardware to a virtual machine.

  These hardware features differ between Intel and AMD processors. Intel named its technology >VT-x. AMD calls theirs AMD-V. The Intel and AMD support for virtualization is very different in detail, but not very different in principle.

  **Note:** On many systems, the hardware virtualization features first need to be enabled in the BIOS before Oracle VM VirtualBox can use them.

- As opposed to other virtualization software, for many usage scenarios, Oracle VM VirtualBox does not *require* hardware virtualization features to be present. Through sophisticated techniques, Oracle VM VirtualBox virtualizes many guest operating systems entirely in *software*. This means that you can run virtual machines even on older processors which do not support hardware virtualization.

Even though Oracle VM VirtualBox does not always require hardware virtualization, enabling it is *required* in the following scenarios:

- Certain rare guest operating systems like OS/2 make use of very esoteric processor instructions that are not supported with our software virtualization. For virtual machines that are configured to contain such an operating system, hardware virtualization is enabled automatically.
- Oracle VM VirtualBox's 64-bit guest support, added with version 2.0, and multiprocessing (SMP), added with version 3.0, both require hardware virtualization to be enabled. This is not much of a limitation since the vast majority of today's 64-bit and multicore CPUs ship with hardware virtualization anyway. The exceptions to this rule are older Intel Celeron and AMD Opteron CPUs, for example.

**Warning:** Do not run other hypervisors, either open source or commercial virtualization products, together with Oracle VM VirtualBox. While several hypervisors can normally be *installed* in parallel, do not attempt to *run* several virtual machines from competing hypervisors at the same time. Oracle VM VirtualBox cannot track what another hypervisor is currently attempting to do on the same host, and especially if several products attempt to use hardware virtualization features such as VT-x, this can crash the entire host. Also, within Oracle VM VirtualBox, you can mix software and hardware virtualization when running multiple VMs. In certain cases a small performance penalty will be unavoidable when mixing VT-x and software virtualization VMs. We recommend not mixing virtualization modes if maximum performance and low overhead are essential. This does *not* apply to AMD-V.

## 10.4 Paravirtualization Providers

Oracle VM VirtualBox enables the exposure of a paravirtualization interface, to facilitate accurate and efficient execution of software within a virtual machine. These interfaces require the guest operating system to recognize their presence and make use of them in order to leverage the benefits of communicating with the Oracle VM VirtualBox hypervisor.

Most modern mainstream guest operating systems, including Windows and Linux, ship with support for one or more paravirtualization interfaces. Hence, there is typically no need to install additional software in the guest to take advantage of this feature.

Exposing a paravirtualization provider to the guest operating system does not rely on the choice of host platforms. For example, the *Hyper-V* paravirtualization provider can be used for VMs to run on any host platform supported by Oracle VM VirtualBox and not just Windows.

Oracle VM VirtualBox provides the following interfaces:

- **Minimal**: Announces the presence of a virtualized environment. Additionally, reports the TSC and APIC frequency to the guest operating system. This provider is mandatory for running any Mac OS X guests.
- **KVM**: Presents a Linux KVM hypervisor interface which is recognized by Linux kernels version 2.6.25 or later. Oracle VM VirtualBox's implementation currently supports paravirtualized clocks and SMP spinlocks. This provider is recommended for Linux guests.
- **Hyper-V**: Presents a Microsoft Hyper-V hypervisor interface which is recognized by Windows 7 and newer operating systems. Oracle VM VirtualBox's implementation currently supports paravirtualized clocks, APIC frequency reporting, guest debugging, guest crash reporting and relaxed timer checks. This provider is recommended for Windows guests.

## 10.5 Details About Software Virtualization

Implementing virtualization on x86 CPUs with no hardware virtualization support is an extraordinarily complex task because the CPU architecture was not designed to be virtualized. The problems can usually be solved, but at the cost of reduced performance. Thus, there is a constant clash between virtualization performance and accuracy.

The x86 instruction set was originally designed in the 1970s and underwent significant changes with the addition of protected mode in the 1980s with the 286 CPU architecture and then again with the Intel 386 and its 32-bit architecture. Whereas the 386 did have limited virtualization support for real mode operation with V86 mode, as used by the "DOS Box" of Windows 3.x and OS/2 2.x, no support was provided for virtualizing the entire architecture.

In theory, software virtualization is not overly complex. There are four privilege levels, called *rings*, provided by the hardware. Typically only two rings are used: ring 0 for kernel mode and ring 3 for user mode. Additionally, one needs to differentiate between *host context* and *guest context*.

In host context, everything is as if no hypervisor was active. This might be the active mode if another application on your host has been scheduled CPU time. In that case, there is a host ring 3 mode and a host ring 0 mode. The hypervisor is not involved.

In guest context, however, a virtual machine is active. So long as the guest code is running in ring 3, this is not much of a problem since a hypervisor can set up the page tables properly and run that code natively on the processor. The problems mostly lie in how to intercept what the guest's kernel does.

There are several possible solutions to these problems. One approach is full software emulation, usually involving recompilation. That is, all code to be run by the guest is analyzed, transformed into a form which will not allow the guest to either modify or see the true state of the CPU, and only then executed. This process is obviously highly complex and costly in terms of performance. Oracle VM VirtualBox contains a recompiler based on QEMU which can be used for pure software emulation, but the recompiler is only activated in special situations, described below.

Another possible solution is paravirtualization, in which only specially modified guest OSes are allowed to run. This way, most of the hardware access is abstracted and any functions which would normally access the hardware or privileged CPU state are passed on to the hypervisor instead. Paravirtualization can achieve good functionality and performance on standard x86 CPUs, but it can only work if the guest OS can actually be modified, which is obviously not always the case.

Oracle VM VirtualBox chooses a different approach. When starting a virtual machine, through its ring-0 support kernel driver, Oracle VM VirtualBox has set up the host system so that it can run most of the guest code natively, but it has inserted itself at the "bottom" of the picture. It can then assume control when needed. If a privileged instruction is executed, the guest traps, in particular because an I/O register was accessed and a device needs to be virtualized, or external interrupts occur. Oracle VM VirtualBox may then handle this and either route a request to a virtual device or possibly delegate handling such things to the guest or host OS. In guest context, Oracle VM VirtualBox can therefore be in one of three states:

- Guest ring 3 code is run unmodified, at full speed, as much as possible. The number of faults will generally be low, unless the guest allows port I/O from ring 3. This is something we cannot do as we do not want the guest to be able to access real ports. This is also referred to as *raw mode*, as the guest ring-3 code runs unmodified.

- For guest code in ring 0, Oracle VM VirtualBox employs a clever trick. It actually reconfigures the guest so that its ring-0 code is run in ring 1 instead, which is normally not used in x86 operating systems). As a result, when guest ring-0 code, actually running n ring 1, such as a guest device driver attempts to write to an I/O register or execute a privileged instruction, the Oracle VM VirtualBox hypervisor in the "real" ring 0 can take over.

- The hypervisor (VMM) can be active. Every time a fault occurs, Oracle VM VirtualBox looks at the offending instruction and can relegate it to a virtual device or the host OS or the guest OS or run it in the recompiler.

  In particular, the recompiler is used when guest code disables interrupts and Oracle VM VirtualBox cannot figure out when they will be switched back on. In these situations, Oracle VM VirtualBox actually analyzes the guest code using its own disassembler. Also, certain privileged instructions such as LIDT need to be handled specially. Finally, any real-mode or protected-mode code, such as BIOS code, a DOS guest, or any operating system startup, is run in the recompiler entirely.

Unfortunately this only works to a degree. Among others, the following situations require special handling:

- Running ring 0 code in ring 1 causes a lot of additional instruction faults, as ring 1 is not allowed to execute any privileged instructions, of which guest's ring-0 contains plenty. With each of these faults, the VMM must step in and emulate the code to achieve the desired behavior. While this works, emulating thousands of these faults is very expensive and severely hurts the performance of the virtualized guest.

- There are certain flaws in the implementation of ring 1 in the x86 architecture that were never fixed. Certain instructions that *should* trap in ring 1 do not. This affects, for example, the LGDT/SGDT, LIDT/SIDT, or POPF/PUSHF instruction pairs. Whereas the "load" operation is privileged and can therefore be trapped, the "store" instruction always succeed. If the guest is allowed to execute these, it will see the true state of the CPU, not the virtualized state. The CPUID instruction also has the same problem.

- A hypervisor typically needs to reserve some portion of the guest's address space, both linear address space and selectors, for its own use. This is not entirely transparent to the guest OS and may cause clashes.

- The SYSENTER instruction, used for system calls, executed by an application running in a guest OS always transitions to ring 0. But that is where the hypervisor runs, not the guest OS. In this case, the hypervisor must trap and emulate the instruction even when it is not desirable.

- The CPU segment registers contain a "hidden" descriptor cache which is not software-accessible. The hypervisor cannot read, save, or restore this state, but the guest OS may use it.

- Some resources must, and can, be trapped by the hypervisor, but the access is so frequent that this creates a significant performance overhead. An example is the TPR (Task Priority) register in 32-bit mode. Accesses to this register must be trapped by the hypervisor. But certain guest operating systems, notably Windows and Oracle Solaris, write this register very often, which adversely affects virtualization performance.

To fix these performance and security issues, Oracle VM VirtualBox contains a Code Scanning and Analysis Manager (CSAM), which disassembles guest code, and the Patch Manager (PATM), which can replace it at runtime.

Before executing ring 0 code, CSAM scans it recursively to discover problematic instructions. PATM then performs *in-situ* patching. It replaces the instruction with a jump to hypervisor memory where an integrated code generator has placed a more suitable implementation. In reality, this is a very complex task as there are lots of odd situations to be discovered and handled correctly. So, with its current complexity, one could argue that PATM is an advanced *in-situ* recompiler.

In addition, every time a fault occurs, Oracle VM VirtualBox analyzes the offending code to determine if it is possible to patch it in order to prevent it from causing more faults in the

future. This approach works well in practice and dramatically improves software virtualization performance.

## 10.6 Details About Hardware Virtualization

With Intel VT-x, there are two distinct modes of CPU operation: VMX root mode and non-root mode.

- In root mode, the CPU operates much like older generations of processors without VT-x support. There are four privilege levels, called rings, and the same instruction set is supported, with the addition of several virtualization specific instruction. Root mode is what a host operating system without virtualization uses, and it is also used by a hypervisor when virtualization is active.
- In non-root mode, CPU operation is significantly different. There are still four privilege rings and the same instruction set, but a new structure called VMCS (Virtual Machine Control Structure) now controls the CPU operation and determines how certain instructions behave. Non-root mode is where guest systems run.

Switching from root mode to non-root mode is called “VM entry”, the switch back is “VM exit”. The VMCS includes a guest and host state area which is saved/restored at VM entry and exit. Most importantly, the VMCS controls which guest operations will cause VM exits.

The VMCS provides fairly fine-grained control over what the guests can and cannot do. For example, a hypervisor can allow a guest to write certain bits in shadowed control registers, but not others. This enables efficient virtualization in cases where guests can be allowed to write control bits without disrupting the hypervisor, while preventing them from altering control bits over which the hypervisor needs to retain full control. The VMCS also provides control over interrupt delivery and exceptions.

Whenever an instruction or event causes a VM exit, the VMCS contains information about the exit reason, often with accompanying detail. For example, if a write to the CR0 register causes an exit, the offending instruction is recorded, along with the fact that a write access to a control register caused the exit, and information about source and destination register. Thus the hypervisor can efficiently handle the condition without needing advanced techniques such as CSAM and PATM described above.

VT-x inherently avoids several of the problems which software virtualization faces. The guest has its own completely separate address space not shared with the hypervisor, which eliminates potential clashes. Additionally, guest OS kernel code runs at privilege ring 0 in VMX non-root mode, obviating the problems by running ring 0 code at less privileged levels. For example the SYSENTER instruction can transition to ring 0 without causing problems. Naturally, even at ring 0 in VMX non-root mode, any I/O access by guest code still causes a VM exit, allowing for device emulation.

The biggest difference between VT-x and AMD-V is that AMD-V provides a more complete virtualization environment. VT-x requires the VMX non-root code to run with paging enabled, which precludes hardware virtualization of real-mode code and non-paged protected-mode software. This typically only includes firmware and OS loaders, but nevertheless complicates VT-x hypervisor implementation. AMD-V does not have this restriction.

Of course hardware virtualization is not perfect. Compared to software virtualization, the overhead of VM exits is relatively high. This causes problems for devices whose emulation requires high number of traps. One example is the VGA device in 16-color modes, where not only every I/O port access but also every access to the framebuffer memory must be trapped.

## 10.7 Nested Paging and VPIDs

In addition to normal hardware virtualization, your processor may also support the following additional sophisticated techniques:

- Nested paging implements some memory management in hardware, which can greatly accelerate hardware virtualization since these tasks no longer need to be performed by the virtualization software.

  With nested paging, the hardware provides another level of indirection when translating linear to physical addresses. Page tables function as before, but linear addresses are now translated to "guest physical" addresses first and not physical addresses directly. A new set of paging registers now exists under the traditional paging mechanism and translates from guest physical addresses to host physical addresses, which are used to access memory.

  Nested paging eliminates the overhead caused by VM exits and page table accesses. In essence, with nested page tables the guest can handle paging without intervention from the hypervisor. Nested paging thus significantly improves virtualization performance.

  On AMD processors, nested paging has been available starting with the Barcelona (K10) architecture. They now call it rapid virtualization indexing (RVI). Intel added support for nested paging, which they call extended page tables (EPT), with their Core i7 (Nehalem) processors.

  If nested paging is enabled, the Oracle VM VirtualBox hypervisor can also use *large pages* to reduce TLB usage and overhead. This can yield a performance improvement of up to 5%. To enable this feature for a VM, you use the `VBoxManage modifyvm --largepages` command. See chapter 8.8, *VBoxManage modifyvm*, page 135.

  If you have an Intel CPU with EPT, please consult chapter 13.4.1, *CVE-2018-3646*, page 299 for security concerns regarding EPT.

- On Intel CPUs, a hardware feature called Virtual Processor Identifiers (VPIDs) can greatly accelerate context switching by reducing the need for expensive flushing of the processor's Translation Lookaside Buffers (TLBs).

  To enable these features for a VM, you use the `VBoxManage modifyvm --vtxvpid` and `--largepages` commands. See chapter 8.8, *VBoxManage modifyvm*, page 135.

# 11 Oracle VM VirtualBox Programming Interfaces

Oracle VM VirtualBox comes with comprehensive support for third-party developers. The so-called *Main API* of Oracle VM VirtualBox exposes the entire feature set of the virtualization engine. It is completely documented and available to anyone who wishes to control Oracle VM VirtualBox programmatically.

The Main API is made available to C++ clients through COM on Windows hosts or XPCOM on other hosts. Bridges also exist for SOAP, Java and Python.

All programming information such as documentation, reference information, header and other interface files, as well as samples have been split out to a separate **Software Development Kit (SDK)**. The SDK is available for download from `http://www.virtualbox.org`. In particular, the SDK comes with a Programming Guide and Reference manual in PDF format. This manual contains, among other things, the information that was previously in this chapter of the User Manual.

# 12 Troubleshooting

This chapter provides answers to commonly asked questions. In order to improve your user experience with Oracle VM VirtualBox, it is recommended to read this section to learn more about common pitfalls and get recommendations on how to use the product.

## 12.1 Procedures and Tools

### 12.1.1 Categorizing and Isolating Problems

More often than not, a virtualized guest behaves like a physical system. Any problems that a physical machine would encounter, a virtual machine will encounter as well. If, for example, Internet connectivity is lost due to external issues, virtual machines will be affected just as much as physical ones.

If a true Oracle VM VirtualBox problem is encountered, it helps to categorize and isolate the problem first. Here are some of the questions that should be answered before reporting a problem:

- Is the problem specific to a certain guest OS? Or a specific release of a guest OS? Especially with Linux guest related problems, the issue may be specific to a certain distribution and version of Linux.
- Is the problem specific to a certain host OS? Problems are usually not host OS specific, because most of the Oracle VM VirtualBox code base is shared across all supported platforms, but especially in the areas of networking and USB support, there are significant differences between host platforms. Some GUI related issues are also host specific.
- Is the problem specific to certain host hardware? This category of issues is typically related to the host CPU. Because of significant differences between VT-x and AMD-V, problems may be specific to one or the other technology. The exact CPU model may also make a difference, even for software virtualization, because different CPUs support different features, which may affect certain aspects of guest CPU operation.
- Is the problem specific to a certain virtualization mode? Some problems may only occur in software virtualization mode, others may be specific to hardware virtualization.
- Is the problem specific to guest SMP? That is, is it related to the number of virtual CPUs (VCPUs) in the guest? Using more than one CPU usually significantly affects the internal operation of a guest OS.
- Is the problem specific to the Guest Additions? In some cases, this is obvious, such as a shared folders problem. In other cases such as display problems, it may be less obvious. If the problem is Guest Additions specific, is it also specific to a certain version of the Guest Additions?
- Is the problem specific to a certain environment? Some problems are related to a particular environment external to the VM. This usually involves network setup. Certain configurations of external servers such as DHCP or PXE may expose problems which do not occur with other, similar servers.

- Is the problem a regression? Knowing that an issue is a regression usually makes it significantly easier to find the solution. In this case, it is crucial to know which version is affected and which is not.

### 12.1.2 Collecting Debugging Information

For problem determination, it is often important to collect debugging information which can be analyzed by Oracle VM VirtualBox support. This section contains information about what kind of information can be obtained.

Every time Oracle VM VirtualBox starts up a VM, a so-called *release log file* is created, containing lots of information about the VM configuration and runtime events. The log file is called `VBox.log` and resides in the VM log file folder. Typically this will be a directory as follows:

```
$HOME/VirtualBox VMs/{machinename}/Logs
```

When starting a VM, the configuration file of the last run will be renamed to `.1`, up to `.3`. Sometimes when there is a problem, it is useful to have a look at the logs. Also when requesting support for Oracle VM VirtualBox, supplying the corresponding log file is mandatory.

For convenience, for each virtual machine, the VirtualBox Manager window can show these logs in a window. To access it, select a virtual machine from the list on the left and select **Show Log** from the **Machine** menu.

The release log file, VBox.log, contains a wealth of diagnostic information, such as Host OS type and version, Oracle VM VirtualBox version and build (32-bit or 64-bit). It also includes a complete dump of the guest's configuration (CFGM), detailed information about the host CPU type and supported features, whether hardware virtualization is enabled, information about VT-x/AMD-V setup, state transitions (such as creating, running, paused, stopping), guest BIOS messages, Guest Additions messages, device-specific log entries and, at the end of execution, final guest state and condensed statistics.

In case of crashes, it is very important to collect *crash dumps*. This is true for both host and guest crashes. For information about enabling core dumps on Linux, Oracle Solaris, and OS X systems, refer to the following core dump article on the Oracle VM VirtualBox website:

`http://www.virtualbox.org/wiki/Core_dump`.

You can also use `VBoxManage debugvm` to create a dump of a complete virtual machine. See chapter 8.42, *VBoxManage debugvm*, page 195.

For network related problems, it is often helpful to capture a trace of network traffic. If the traffic is routed through an adapter on the host, it is possible to use Wireshark or a similar tool to capture the traffic there. However, this often also includes a lot of traffic unrelated to the VM.

Oracle VM VirtualBox provides an ability to capture network traffic only on a specific VM's network adapter. Refer to the following network tracing article on the Oracle VM VirtualBox website for information on enabling this capture:

`http://www.virtualbox.org/wiki/Network_tips`.

The trace files created by Oracle VM VirtualBox are in `.pcap` format and can be easily analyzed with Wireshark.

### 12.1.3 The Built-In VM Debugger

Oracle VM VirtualBox includes a built-in VM debugger, which advanced users may find useful. This debugger enables you to examine and, to some extent, control the VM state.

> **Warning:** Use the VM debugger at your own risk. There is no support for it, and the following documentation is only made available for advanced users with a very high level of familiarity with the x86/AMD64 machine instruction set, as well as detailed knowledge of the PC architecture. A degree of familiarity with the internals of the guest OS in question may also be very helpful.

The VM debugger is available in all regular production versions of Oracle VM VirtualBox, but it is disabled by default because the average user will have little use for it. There are two ways to access the debugger:

- Using a debugger console window displayed alongside the VM
- Using the `telnet` protocol on port 5000

The debugger can be enabled in the following ways:

- Start the VM directly using `VirtualBox --startvm`, with an additional `--dbg`, `--debug`, or `--debug-command-line` argument. See the `VirtualBox` command usage help for details.
- Set the `VBOX_GUI_DBG_ENABLED` or `VBOX_GUI_DBG_AUTO_SHOW` environment variable to `true` before launching the Oracle VM VirtualBox process. Setting these variables, only their presence is checked, is effective even when the first Oracle VM VirtualBox process is the VM selector window. VMs subsequently launched from the selector will have the debugger enabled.
- Set the `GUI/Dbg/Enabled` extra data item to `true` before launching the VM. This can be set globally or on a per VM basis.

A new **Debug** menu entry is added to the Oracle VM VirtualBox application. This menu enables the user to open the debugger console.

The VM debugger command syntax is loosely modeled on Microsoft and IBM debuggers used on DOS, OS/2, and Windows. Users familiar with symdeb, CodeView, or the OS/2 kernel debugger will find the Oracle VM VirtualBox VM debugger familiar.

The most important command is `help`. This will print brief usage help for all debugger commands. The set of commands supported by the VM debugger changes frequently and the `help` command is always up-to-date.

A brief summary of frequently used commands is as follows:

- `stop`: Stops the VM execution and enables single stepping
- `g`: Continue VM execution
- `t`: Single step an instruction
- `rg/rh/r`: Print the guest/hypervisor/current registers
- `kg/kh/k`: Print the guest/hypervisor/current call stack
- `da/db/dw/dd/dq`: Print memory contents as ASCII/bytes/words/dwords/qwords
- `u`: Unassemble memory
- `dg`: Print the guest's GDT
- `di`: Print the guest's IDT
- `dl`: Print the guest's LDT
- `dt`: Print the guest's TSS
- `dp*`: Print the guest's page table structures
- `bp/br`: Set a normal/recompiler breakpoint
- `bl`: List breakpoints

- `bc`: Clear a breakpoint
- `writecore`: Write a VM core file to disk. See chapter 12.1.4, *VM Core Format*, page 280

See the built-in `help` for other available commands.

The VM debugger supports symbolic debugging, although symbols for guest code are often not available. For Oracle Solaris guests, the `detect` command automatically determines the guest OS version and locates kernel symbols in guest's memory. Symbolic debugging is then available. For Linux guests, the `detect` commands also determines the guest OS version, but there are no symbols in the guest's memory. Kernel symbols are available in the file `/proc/kallsyms` on Linux guests. This file must be copied to the host, for example using `scp`. The `loadmap` debugger command can be used to make the symbol information available to the VM debugger. Note that the `kallsyms` file contains the symbols for the currently loaded modules. If the guest's configuration changes, the symbols will change as well and must be updated.

For all guests, a simple way to verify that the correct symbols are loaded is the `k` command. The guest is normally idling and it should be clear from the symbolic information that the guest operating system's idle loop is being executed.

Another group of debugger commands is the set of `info` commands. Running `info help` provides complete usage information. The information commands provide ad-hoc data pertinent to various emulated devices and aspects of the VMM. There is no general guideline for using the `info` commands, the right command to use depends entirely on the problem being investigated. Some of the `info` commands are as follows:

- `cfgm`: Print a branch of the configuration tree
- `cpuid`: Display the guest CPUID leaves
- `ioport`: Print registered I/O port ranges
- `mmio`: Print registered MMIO ranges
- `mode` – print the current paging mode
- `pit`: Print the i8254 PIT state
- `pic`: Print the i8259A PIC state
- `ohci/ehci/xhci`: Print a subset of the OHCI/EHCI/xHCI USB controller state
- `pcnet0`: Print the PCnet state
- `vgatext`: Print the contents of the VGA framebuffer formatted as standard text mode
- `timers`: Print all VM timers

The output of the `info` commands generally requires in-depth knowledge of the emulated device or Oracle VM VirtualBox VMM internals. However, when used properly, the information provided can be invaluable.

### 12.1.4 VM Core Format

Oracle VM VirtualBox uses the 64-bit ELF format for its VM core files created by `VBoxManage debugvm`, see chapter 8.42, *VBoxManage debugvm*, page 195. The VM core file contain the memory and CPU dumps of the VM and can be useful for debugging your guest OS. The 64-bit ELF object format specification can be obtained at:

`http://downloads.openwatcom.org/ftp/devel/docs/elf-64-gen.pdf`.

The overall layout of the VM core format is as follows:

```
[ ELF 64 Header]
[ Program Header, type PT_NOTE ]
  → offset to COREDESCRIPTOR
[ Program Header, type PT_LOAD ] - one for each contiguous physical memory range
  → Memory offset of range
  → File offset
[ Note Header, type NT_VBOXCORE ]
[ COREDESCRIPTOR ]
  → Magic
  → VM core file version
  → VBox version
  → Number of vCPUs etc.
[ Note Header, type NT_VBOXCPU ] - one for each vCPU
[ vCPU 1 Note Header ]
  [ DBGFCORECPU - vCPU 1 dump ]
[ Additional Notes + Data ] - currently unused
[ Memory dump ]
```

The memory descriptors contain physical addresses relative to the guest and not virtual addresses. Regions of memory such as MMIO regions are not included in the core file.

The relevant data structures and definitions can be found in the Oracle VM VirtualBox sources under the following header files: `include/VBox/dbgfcorefmt.h`, `include/iprt/x86.h` and `src/VBox/Runtime/include/internal/ldrELFCommon.h`.

The VM core file can be inspected using `elfdump` and GNU `readelf` or other similar utilities.

## 12.2 General Troubleshooting

### 12.2.1 Guest Shows IDE/SATA Errors for File-Based Images on Slow Host File System

Occasionally, some host file systems provide very poor writing performance and as a consequence cause the guest to time out IDE/SATA commands. This is normal behavior and should normally cause no real problems, as the guest should repeat commands that have timed out. However, guests such as some Linux versions have severe problems if a write to an image file takes longer than about 15 seconds. Some file systems however require more than a minute to complete a single write, if the host cache contains a large amount of data that needs to be written.

The symptom for this problem is that the guest can no longer access its files during large write or copying operations, usually leading to an immediate hang of the guest.

In order to work around this problem, the true fix is to use a faster file system that does not exhibit such unacceptable write performance, it is possible to flush the image file after a certain amount of data has been written. This interval is normally infinite, but can be configured individually for each disk of a VM.

For IDE disks use the following command:

```
VBoxManage setextradata "VM name"
"VBoxInternal/Devices/piix3ide/0/LUN#[x]/Config/FlushInterval" [b]
```

For SATA disks use the following command:

```
VBoxManage setextradata "VM name"
"VBoxInternal/Devices/ahci/0/LUN#[x]/Config/FlushInterval" [b]
```

The value [x] that selects the disk for IDE is 0 for the master device on the first channel, 1 for the slave device on the first channel, 2 for the master device on the second channel or 3 for the slave device on the second channel. For SATA use values between 0 and 29. Only disks support this configuration option; it must not be set for CD/DVD drives.

The unit of the interval [b] is the number of bytes written since the last flush. The value for it must be selected so that the occasional long write delays do not occur. Since the proper flush interval depends on the performance of the host and the host filesystem, finding the optimal value

that makes the problem disappear requires some experimentation. Values between 1000000 and 10000000 (1 to 10 megabytes) are a good starting point. Decreasing the interval both decreases the probability of the problem and the write performance of the guest. Setting the value unnecessarily low will cost performance without providing any benefits. An interval of 1 will cause a flush for each write operation and should solve the problem in any case, but has a severe write performance penalty.

Providing a value of 0 for [b] is treated as an infinite flush interval, effectively disabling this workaround. Removing the extra data key by specifying no value for [b] has the same effect.

### 12.2.2 Responding to Guest IDE/SATA Flush Requests

If desired, the virtual disk images can be flushed when the guest issues the IDE FLUSH CACHE command. Normally these requests are ignored for improved performance. The parameters below are only accepted for disk drives. They must not be set for DVD drives.

To enable flushing for IDE disks, issue the following command:

```
VBoxManage setextradata "VM name" "VBoxInternal/Devices/piix3ide/0/LUN#[x]/Config/IgnoreFlush" 0
```

The value [x] that selects the disk is 0 for the master device on the first channel, 1 for the slave device on the first channel, 2 for the master device on the second channel or 3 for the master device on the second channel.

To enable flushing for SATA disks, issue the following command:

```
VBoxManage setextradata "VM name" "VBoxInternal/Devices/ahci/0/LUN#[x]/Config/IgnoreFlush" 0
```

The value [x] that selects the disk can be a value between 0 and 29.

Note that this does not affect the flushes performed according to the configuration described in chapter 12.2.1, *Guest Shows IDE/SATA Errors for File-Based Images on Slow Host File System*, page 281. Restoring the default of ignoring flush commands is possible by setting the value to 1 or by removing the key.

### 12.2.3 Performance Variation with Frequency Boosting

Many newer multi-core processors support some form of frequency boosting, which means that if only one core is utilized, it can run possibly 50% faster or even more than the rated CPU frequency. This causes measured performance to vary somewhat as a function of the momentary overall system load. The exact behavior depends strongly on the specific processor model.

As a consequence, benchmarking on systems which utilize frequency boosting may produce unstable and non-repeatable results. This is especially true if benchmark runs are short, of the order of seconds. To obtain stable results, benchmarks must be run over longer periods of time and with a constant system load apart from the VM being tested.

### 12.2.4 Frequency Scaling Effect on CPU Usage

On some hardware platforms and operating systems, CPU frequency scaling may cause CPU usage reporting to be highly misleading. This happens in situations when the host CPU load is significant but not heavy, such as 15-30% of the maximum.

Most operating systems determine CPU usage in terms of time spent, measuring for example how many nanoseconds the systems or a process was active within one second. However, in order to save energy, modern systems can significantly scale down CPU speed when the system is not fully loaded. Naturally, when the CPU is running at for example one half of its maximum speed, the same number of instructions will take roughly twice as long to execute compared to running at full speed.

Depending on the specific hardware and host OS, this effect can very significantly skew the CPU usage reported by the OS. The reported CPU usage can be several times higher than what

it would have been had the CPU been running at full speed. The effect can be observed both on the host OS and in a guest OS.

### 12.2.5 Inaccurate Windows CPU Usage Reporting

CPU usage reporting tools which come with Windows, such as Task Manager or Resource Monitor, do not take the time spent processing hardware interrupts into account. If the interrupt load is heavy, with thousands of interrupts per second, CPU usage may be significantly underreported.

This problem affects Windows as both host and guest OS. Sysinternals tools, such as Process Explorer, do not suffer from this problem.

### 12.2.6 Poor Performance Caused by Host Power Management

On some hardware platforms and operating systems, virtualization performance is negatively affected by host CPU power management. The symptoms may be choppy audio in the guest or erratic guest clock behavior.

Some of the problems may be caused by firmware and/or host operating system bugs. Therefore, updating the firmware and applying operating systems fixes is recommended.

For optimal virtualization performance, the C1E power state support in the system's BIOS should be disabled, if such a setting is available. Not all systems support the C1E power state. On Intel systems, the `Intel C State` setting should be disabled. Disabling other power management settings may also improve performance. However, a balance between performance and power consumption must always be considered.

### 12.2.7 GUI: 2D Video Acceleration Option is Grayed Out

To use 2D Video Acceleration within Oracle VM VirtualBox, your host's video card should support certain OpenGL extensions. On startup, Oracle VM VirtualBox checks for those extensions, and, if the test fails, this option is silently grayed out.

To find out why it has failed, you can manually execute the following command:

```
VBoxTestOGL --log "log_file_name" --test 2D
```

It will list the required OpenGL extensions one by one and will show you which one failed the test. This usually means that you are running an outdated or misconfigured OpenGL driver on your host. It can also mean that your video chip is lacking required functionality.

## 12.3 Windows Guests

### 12.3.1 No USB 3.0 Support in Windows 7 Guests

If a Windows 7 or Windows Server 2008 R2 guest is configured for USB 3.0 (xHCI) support, the guest OS will not have any USB support at all. This happens because Windows 7 predates USB 3.0 and therefore does not ship with any xHCI drivers. Microsoft also does not offer any vendor-provided xHCI drivers through Windows Update.

To solve this problem, it is necessary to download and install the Intel xHCI driver in the guest. Intel offers the driver as the USB 3.0 eXtensible Host Controller (xHCI) driver for Intel 7 Series/C216 chipsets.

Note that the driver only supports Windows 7 and Windows Server 2008 R2. The driver package includes support for both 32-bit and 64-bit OS variants.

### 12.3.2 Windows Bluescreens After Changing VM Configuration

Changing certain virtual machine settings can cause Windows guests to fail during start up with a bluescreen. This may happen if you change VM settings after installing Windows, or if you copy a disk image with an already installed Windows to a newly created VM which has settings that differ from the original machine.

This applies in particular to the following settings:

- The ACPI and I/O APIC settings should never be changed after installing Windows. Depending on the presence of these hardware features, the Windows installation program chooses special kernel and device driver versions and will fail to startup should these hardware features be removed. Enabling them for a Windows VM which was installed without them does not cause any harm. However, Windows will not use these features in this case.
- Changing the storage controller hardware will cause bootup failures as well. This might also apply to you if you copy a disk image from an older version of Oracle VM VirtualBox to a virtual machine created with a newer Oracle VM VirtualBox version. The default subtype of IDE controller hardware was changed from PIIX3 to PIIX4 with Oracle VM VirtualBox 2.2. Make sure these settings are identical.

### 12.3.3 Windows 0x101 Bluescreens with SMP Enabled (IPI Timeout)

If a VM is configured to have more than one processor (symmetrical multiprocessing, SMP), some configurations of Windows guests crash with an 0x101 error message, indicating a timeout for interprocessor interrupts (IPIs). These interrupts synchronize memory management between processors.

According to Microsoft, this is due to a race condition in Windows. A hotfix is available. See http://support.microsoft.com/kb/955076.

If this does not help, please reduce the number of virtual processors to 1.

### 12.3.4 Windows 2000 Installation Failures

When installing Windows 2000 guests, you might run into one of the following issues:

- Installation reboots, usually during component registration.
- Installation fills the whole hard disk with empty log files.
- Installation complains about a failure installing `msgina.dll`.

These problems are all caused by a bug in the hard disk driver of Windows 2000. After issuing a hard disk request, there is a race condition in the Windows driver code which leads to corruption if the operation completes too fast. For example, the hardware interrupt from the IDE controller arrives too soon. With physical hardware, there is a guaranteed delay in most systems so the problem is usually hidden there. However, it should be possible to also reproduce it on physical hardware. In a virtual environment, it is possible for the operation to be done immediately, especially on very fast systems with multiple CPUs, and the interrupt is signaled sooner than on a physical system. The solution is to introduce an artificial delay before delivering such interrupts. This delay can be configured for a VM using the following command:

```
VBoxManage setextradata "VM name" "VBoxInternal/Devices/piix3ide/0/Config/IRQDelay" 1
```

This sets the delay to one millisecond. In case this does not help, increase it to a value between 1 and 5 milliseconds. Please note that this slows down disk performance. After installation, you should be able to remove the key, or set it to 0.

### 12.3.5 How to Record Bluescreen Information from Windows Guests

When Windows guests run into a kernel crash, they display the infamous bluescreen. Depending on how Windows is configured, the information will remain on the screen until the machine is restarted or it will reboot automatically. During installation, Windows is usually configured to reboot automatically. With automatic reboots, there is no chance to record the bluescreen information which might be important for problem determination.

Oracle VM VirtualBox provides a method of halting a guest when it wants to perform a reset. In order to enable this feature, use the following command:

```
VBoxManage setextradata "VM name" "VBoxInternal/PDM/HaltOnReset" 1
```

### 12.3.6 PCnet Driver Failure in 32-bit Windows Server 2003 Guests

Certain editions of Windows 2000 and 2003 servers support more than 4 GB RAM on 32-bit systems. The AMD PCnet network driver shipped with Windows Server 2003 fails to load if the 32-bit guest OS uses paging extensions, which will occur with more than approximately 3.5 GB RAM assigned to the VM.

This problem is known to occur with version 4.38.0.0 of the PCnet driver. The issue was fixed in version 4.51.0.0 of the driver, which is available as a separate download. An alternative solution may be changing the emulated NIC type to Intel PRO/1000 MT Desktop (82540EM), or reducing the RAM assigned to the VM to approximately 3.5 GB or less.

### 12.3.7 No Networking in Windows Vista Guests

With Windows Vista, Microsoft dropped support for the AMD PCNet card that Oracle VM VirtualBox used to provide as the default virtual network card before version 1.6.0. For Windows Vista guests, Oracle VM VirtualBox now uses an Intel E1000 card by default.

If, for some reason, you still want to use the AMD card, you need to download the PCNet driver from the AMD website. This driver is available for 32-bit Windows only. You can transfer it into the virtual machine using a shared folder. See chapter 4.3, *Shared Folders*, page 69.

### 12.3.8 Windows Guests may Cause a High CPU Load

Several background applications of Windows guests, especially virus scanners, are known to increases the CPU load notably even if the guest appears to be idle. We recommend to deactivate virus scanners within virtualized guests if possible.

### 12.3.9 Long Delays When Accessing Shared Folders

The performance for accesses to shared folders from a Windows guest might be decreased due to delays during the resolution of the Oracle VM VirtualBox shared folders name service. To fix these delays, add the following entries to the file `\windows\system32\drivers\etc\lmhosts` of the Windows guest:

```
255.255.255.255         VBOXSVR #PRE
255.255.255.255         VBOXSRV #PRE
```

After doing this change, a reboot of the guest is required.

### 12.3.10 USB Tablet Coordinates Wrong in Windows 98 Guests

If a Windows 98 VM is configured to use the emulated USB tablet (absolute pointing device), the coordinate translation may be incorrect and the pointer is restricted to the upper left quarter of the guest's screen.

The USB HID (Human Interface Device) drivers in Windows 98 are very old and do not handle tablets the same way as more recent operating systems do. For example, Windows 2000 and later, Mac OS X, and Oracle Solaris. To work around the problem, use the following command:

```
VBoxManage setextradata "VM name" "VBoxInternal/USB/HidMouse/0/Config/CoordShift" 0
```

To restore the default behavior, remove the key or set its value to 1.

### 12.3.11 Windows Guests are Removed From an Active Directory Domain After Restoring a Snapshot

If a Windows guest is a member of an Active Directory domain and the snapshot feature of Oracle VM VirtualBox is used, it could happen it loses this status after you restore an older snapshot.

The reason is the automatic machine password changing performed by Windows at regular intervals for security purposes. You can disable this feature by following the instruction of the following article from Microsoft: http://support.microsoft.com/kb/154501

### 12.3.12 Restoring d3d8.dll and d3d9.dll

Oracle VM VirtualBox Guest Additions for Windows prior to 4.1.8 did not properly back up the original d3d8.dll and d3d9.dll system files when selecting and installing the experimental Direct3D support. This process replaces both system files with files from the Guest Additions so that Direct3D calls can be handled correctly. Although this issue was fixed with Oracle VM VirtualBox 4.1.8, there is no way the Windows Guest Additions installer can repair these files.

Corruption of these files has no implications if 3D acceleration is enabled and basic Direct3D support is installed. That is, without WDDM on Windows Vista or later, or on older Windows systems like Windows XP. With the basic Direct3D support all Direct3D 8.0 and Direct3D 9.0 applications will utilize Oracle VM VirtualBox Direct3D files directly and thus will run as expected.

For WDDM Direct3D support however, the originally shipped d3d8.dll and d3d9.dll files are required in order to run Direct3D 8.0 and Direct3D 9.0 applications. As a result of the above mentioned system files corruption these applications will not work anymore. See below for a step-by-step guide for restoring the original d3d8.dll and d3d9.dll system files in case the Oracle VM VirtualBox Guest Additions installer warned about those incorrect files or when having trouble running Direct3D applications.

> **Note:** Starting at Windows 7 the 3D desktop, called Aero, uses DirectX 10 for rendering so that corrupted d3d8.dll and d3d9.dll system files will have no effect on the actual rendering.

This is why such a detected file corruption is not considered as fatal for the basic Direct3D installation on all supported Windows guests, and for WDDM Direct3D installation on Windows 7 and later guests.

**Extracting d3d8 and d3d9.dll from a Windows XP installation CD:**

1. Download and install the latest version of 7-Zip File Manager.
2. Browse into the installation CD. For example E:\i386, or E:\amd64 for the 64-bit version.
3. Locate the entries d3d8.dl_ and d3d9.dl_. Double-click on the file names and extract d3d8.dll and d3d9.dll.
4. Reboot Windows in Safe mode.
5. Copy the extracted d3d8.dll and d3d9.dll files to C:\Windows\system32 and C:\Windows\system32\dllcache.

6. Reboot Windows.

**Extracting d3d8 and d3d9.dll from a Windows XP Service pack:**

1. Download and install the latest version of 7-Zip File Manager.
2. Choose Open Inside, to open WindowsXP-KB936929-SP3-x86.exe as an archive and browse the i386 directory.
3. Locate the entries d3d8.dl_ and d3d9.dl_. Double-click on the file names and extract d3d8.dll and d3d9.dll.
4. Reboot Windows in Safe mode.
5. Copy the extracted d3d8.dll and d3d9.dll files to C:\Windows\system32 and C:\Windows\system32\dllcache.
6. Reboot Windows.

Extracting d3d8 and d3d9.dll from a Vista/Windows7 installation CD or Service Pack ISO:

1. Download and install the latest version of 7-Zip File Manager.
2. Browse into the installation CD. For example E:\sources.
3. Locate file install.wim and double-click the file. After the 7-Zip utility unzips the file, you will see a few numbered folders. Each numeric subfolder represents a different version of Windows such as Starter or Home Basic.
4. Open one of the numeric folders and browse to the Windows\System32 directory, or C:\Windows\SysWOW64 for the 64-bit version. Locate and extract the d3d8.dll and d3d9.dll files.
5. Copy extracted the d3d8.dll and d3d9.dll files to C:\Windows\system32 or C:\Windows\SysWOW64. Files from system32 should go to system32, from SysWOW64 to SysWOW64.
6. Reboot Windows.

### 12.3.13 Windows 3.x Limited to 64 MB RAM

Windows 3.x guests are typically limited to 64 MB RAM, even if a VM is assigned much more memory. While Windows 3.1 is theoretically capable of using up to 512 MB RAM, it only uses memory available through the XMS interface. Versions of HIMEM.SYS, the Microsoft XMS manager, shipped with MS-DOS and Microsoft Windows 3.x can only use up to 64 MB on standard PCs.

This is a HIMEM.SYS limitation documented by Microsoft in Knowledge base article KB 116256. Windows 3.1 memory limits are described in detail in Microsoft Knowledge base article KB 84388.

It is possible for Windows 3.x guests to utilize more than 64 MB RAM if a different XMS provider is used. That could be a newer HIMEM.SYS version, such as that shipped with Windows 98, or a more capable third-party memory manager, such as QEMM.

## 12.4 Linux and X11 Guests

### 12.4.1 Linux Guests May Cause a High CPU load

Some Linux guests may cause a high CPU load even if the guest system appears to be idle. This can be caused by a high timer frequency of the guest kernel. Some Linux distributions, for example Fedora, ship a Linux kernel configured for a timer frequency of 1000Hz. We recommend to recompile the guest kernel and to select a timer frequency of 100Hz.

Linux kernels shipped with Red Hat Enterprise Linux (RHEL) as of release 4.7 and 5.1 as well as kernels of related Linux distributions, such as CentOS and Oracle Linux, support a kernel parameter *divider=N*. Hence, such kernels support a lower timer frequency without recompilation. We suggest you add the kernel parameter *divider=10* to select a guest kernel timer frequency of 100Hz.

### 12.4.2 AMD Barcelona CPUs

Most Linux-based guests will fail with AMD Phenoms or Barcelona-level Opterons due to a bug in the Linux kernel. Enable the I/O-APIC to work around the problem. See chapter 3.5, *System Settings*, page 47.

### 12.4.3 Buggy Linux 2.6 Kernel Versions

The following bugs in Linux kernels prevent them from executing correctly in Oracle VM VirtualBox, causing VM boot crashes:

- The Linux kernel version 2.6.18, and some 2.6.17 versions, introduced a race condition that can cause boot crashes in Oracle VM VirtualBox. Please use a kernel version 2.6.19 or later.
- With hardware virtualization and the I/O APIC enabled, kernels before 2.6.24-rc6 may panic on boot with the following message:

  ```
  Kernel panic - not syncing: IO-APIC + timer doesn't work!  Boot with
  apic=debug and send a report.  Then try booting with the 'noapic' option
  ```

  If you see this message, either disable hardware virtualization or the I/O APIC as described in chapter 3.5, *System Settings*, page 47, or upgrade the guest to a newer kernel.

  See `http://www.mail-archive.com/git-commits-head@vger.kernel.org/msg30813.html` for details about the kernel fix.

### 12.4.4 Shared Clipboard, Auto-Resizing, and Seamless Desktop in X11 Guests

Guest desktop services in guests running the X11 window system such as Oracle Solaris and Linux, are provided by a guest service called `VBoxClient`, which runs under the ID of the user who started the desktop session and is automatically started using the following command lines when your X11 user session is started if you are using a common desktop environment such as Gnome or KDE.

```
VBoxClient --clipboard
VBoxClient --display
VBoxClient --seamless
```

If a particular desktop service is not working correctly, it is worth checking whether the process which should provide it is running.

The `VBoxClient` processes create files in the user's home directory with names of the form `.vboxclient-*.pid` when they are running in order to prevent a given service from being started twice. It can happen due to misconfiguration that these files are created owned by root and not deleted when the services are stopped, which will prevent them from being started in future sessions. If the services cannot be started, you may wish to check whether these files still exist.

## 12.5 Oracle Solaris Guests

### 12.5.1 Older Oracle Solaris 10 Releases Crash in 64-bit Mode

Oracle Solaris 10 releases up to and including Oracle Solaris 10 8/07 incorrectly detect newer Intel processors produced since 2007. This problem leads to the 64-bit Oracle Solaris kernel crashing, and usually causing a triple fault, almost immediately during startup, in both virtualized and physical environments.

The recommended solution is upgrading to at least Oracle Solaris 10 5/08. Alternative solutions include forcing Oracle Solaris to always boot the 32-bit kernel or applying a patch for bug 6574102 while Oracle Solaris is using the 32-bit kernel.

### 12.5.2 Certain Oracle Solaris 10 Releases May Take a Long Time to Boot with SMP

When using more than one CPU, Oracle Solaris 10 5/08, Oracle Solaris 10 10/08, and Oracle Solaris 10 5/09 may take a long time to boot and may print warnings on the system console regarding failures to read from disk. This is a bug in Oracle Solaris 10 which affects specific physical and virtual configurations. It is caused by trying to read microcode updates from the boot disk when the disk interrupt is reassigned to a not yet fully initialized secondary CPU. Disk reads will time out and fail, triggering delays of about 45 seconds and warnings.

The recommended solution is upgrading to at least Oracle Solaris 10 10/09 which includes a fix for this problem. Alternative solutions include restricting the number of virtual CPUs to one or possibly using a different storage controller.

### 12.5.3 Solaris 8 5/01 and Earlier May Crash on Startup

Solaris 2.6, 7 and 8 releases up to and including Solaris 8 4/01 ("S8U4") incorrectly set up Machine Check Exception (MCE) MSRs on Pentium 4 and some later Intel CPUs. The problem leads to the Solaris kernel crashing, and usually causing a triple fault, almost immediately during startup, in both virtualized and physical environments. Solaris 9 and later releases are not affected by this problem, and neither is Solaris 2.5.1 and earlier.

The recommended solution is upgrading to at least Solaris 8 7/01 ("S8U5"). Alternative solutions include applying a patch for bugs 4408508 and 4414557 on an unaffected system.

## 12.6 FreeBSD Guests

### 12.6.1 FreeBSD 10.0 May Hang with xHCI

If xHCI (USB 3.0) emulation is enabled for FreeBSD 10.0 guests, the guest OS will hang. This is caused by the guest OS incorrectly handling systems where Message Signaled Interrupts (MSIs) are not used with the xHCI device.

The problem does not exist in earlier FreeBSD releases and was fixed in FreeBSD 10.1.

## 12.7 Windows Hosts

### 12.7.1 VBoxSVC Out-of-Process COM Server Issues

Oracle VM VirtualBox makes use of the Microsoft Component Object Model (COM) for interprocess and intraprocess communication. This enables Oracle VM VirtualBox to share a common configuration among different virtual machine processes and provide several user interface options based on a common architecture. All global status information and configuration is maintained by the process `VBoxSVC.exe`, which is an out-of-process COM server. Whenever an Oracle VM VirtualBox process is started, it requests access to the COM server and Windows automatically starts the process. Note that it should never be started by the end user.

When the last process disconnects from the COM server, it will terminate itself after some seconds. The Oracle VM VirtualBox configuration (XML files) is maintained and owned by the COM server and the files are locked whenever the server runs.

In some cases, such as when a virtual machine is terminated unexpectedly, the COM server will not notice that the client is disconnected and stay active for a longer period of 10 minutes or so, keeping the configuration files locked. In other rare cases the COM server might experience an internal error and subsequently other processes fail to initialize it. In these situations, it is recommended to use the Windows task manager to kill the process `VBoxSVC.exe`.

### 12.7.2 CD/DVD Changes Not Recognized

In case you have assigned a physical CD/DVD drive to a guest and the guest does not notice when the medium changes, make sure that the Windows media change notification (MCN) feature is not turned off. This is represented by the following key in the Windows registry:

```
HKEY_LOCAL_MACHINE\System\CurrentControlSet\Services\Cdrom\Autorun
```

Certain applications may disable this key against Microsoft's advice. If it is set to 0, change it to 1 and reboot your system. Oracle VM VirtualBox relies on Windows notifying it of media changes.

### 12.7.3 Sluggish Response When Using Microsoft RDP Client

If connecting to a Virtual Machine using the Microsoft RDP client, called a Remote Desktop Connection, there can be large delays between input such as moving the mouse over a menu and output. This is because this RDP client collects input for a certain time before sending it to the RDP server.

The interval can be decreased by setting a Windows registry key to smaller values than the default of 100. The key does not exist initially and must be of type DWORD. The unit for its values is milliseconds. Values around 20 are suitable for low-bandwidth connections between the RDP client and server. Values around 4 can be used for a gigabit Ethernet connection. Generally values below 10 achieve a performance that is very close to that of the local input devices and screen of the host on which the Virtual Machine is running.

Depending whether the setting should be changed for an individual user or for the system, set either of the following.

```
HKEY_CURRENT_USER\Software\Microsoft\Terminal Server Client\Min Send Interval
```

```
HKEY_LOCAL_MACHINE\Software\Microsoft\Terminal Server Client\Min Send Interval
```

### 12.7.4 Running an iSCSI Initiator and Target on a Single System

Deadlocks can occur on a Windows host when attempting to access an iSCSI target running in a guest virtual machine with an iSCSI initiator, such as a Microsoft iSCSI Initiator, that is running

on the host. This is caused by a flaw in the Windows cache manager component, and causes sluggish host system response for several minutes, followed by a "Delayed Write Failed" error message in the system tray or in a separate message window. The guest is blocked during that period and may show error messages or become unstable.

Setting the environment variable `VBOX_DISABLE_HOST_DISK_CACHE` to 1 will enable a workaround for this problem until Microsoft addresses the issue. For example, open a command prompt window and start Oracle VM VirtualBox like this:

```
set VBOX_DISABLE_HOST_DISK_CACHE=1
VirtualBox
```

While this will decrease guest disk performance, especially writes, it does not affect the performance of other applications running on the host.

### 12.7.5 Bridged Networking Adapters Missing

If no bridged adapters show up in the **Networking** section of the VM settings, this typically means that the bridged networking driver was not installed properly on your host. This could be due to the following reasons:

- The maximum allowed filter count was reached on the host. In this case, the MSI log would mention the `0x8004a029` error code returned on NetFlt network component install, as follows:

  ```
  VBoxNetCfgWinInstallComponent: Install failed, hr (0x8004a029)
  ```

  You can try to increase the maximum filter count in the Windows registry using the following key:

  ```
  HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\Network\MaxNumFilters
  ```

  The maximum number allowed is 14. After a reboot, try to reinstall Oracle VM VirtualBox.

- The INF cache is corrupt. In this case, the install log (`%windir%\inf\setupapi.log` on XP or `%windir%\inf\setupapi.dev.log` on Vista or later) would typically mention the failure to find a suitable driver package for either the `sun_VBoxNetFlt` or `sun_VBoxNetFltmp` components. The solution then is to uninstall Oracle VM VirtualBox, remove the INF cache (`%windir%\inf\INFCACHE.1`), reboot and try to reinstall Oracle VM VirtualBox.

### 12.7.6 Host-Only Networking Adapters Cannot be Created

If a host-only adapter cannot be created, either with the VirtualBox Manager or the `VBoxManage` command, then the INF cache is probably corrupt. In this case, the install log (`%windir%\inf\setupapi.log` on Windows XP or `%windir%\inf\setupapi.dev.log` on Windows Vista or later) would typically mention the failure to find a suitable driver package for the `sun_VBoxNetAdp` component. Again, as with the bridged networking problem described above, the solution is to uninstall Oracle VM VirtualBox, remove the INF cache (`%windir%\inf\INFCACHE.1`), reboot and try to reinstall Oracle VM VirtualBox.

## 12.8 Linux Hosts

### 12.8.1 Linux Kernel Module Refuses to Load

If the Oracle VM VirtualBox kernel module, `vboxdrv`, refuses to load you may see an "Error inserting vboxdrv: Invalid argument" message. As root, check the output of the `dmesg` command to find out why the load failed. Most probably the kernel disagrees with the version of `gcc` used to compile the module. Make sure that you use the same compiler as used to build the kernel.

### 12.8.2 Linux Host CD/DVD Drive Not Found

If you have configured a virtual machine to use the host's CD/DVD drive, but this does not appear to work, make sure that the current user has permission to access the corresponding Linux device file. This is `/dev/hdc`, `/dev/scd0`, `/dev/cdrom` or similar. On most distributions, the user must be added to a corresponding group, usually called `cdrom` or `cdrw`.

### 12.8.3 Linux Host CD/DVD Drive Not Found (Older Distributions)

On older Linux distributions, if your CD/DVD device has a different name, Oracle VM VirtualBox may be unable to find it. On older Linux hosts, Oracle VM VirtualBox performs the following steps to locate your CD/DVD drives:

1. Oracle VM VirtualBox checks if the environment variable `VBOX_CDROM` is defined. If so, Oracle VM VirtualBox omits all the following checks.
2. Oracle VM VirtualBox tests if `/dev/cdrom` works.
3. Oracle VM VirtualBox checks if any CD/DVD drives are currently mounted by checking `/etc/mtab`.
4. Oracle VM VirtualBox checks if any of the entries in `/etc/fstab` point to CD/DVD devices.

You can set the VBOX_CDROM environment variable to contain a list of your CD/DVD devices, separated by colons. For example:

```
export VBOX_CDROM='/dev/cdrom0:/dev/cdrom1'
```

On modern Linux distributions, Oracle VM VirtualBox uses the hardware abstraction layer (HAL) to locate CD and DVD hardware.

### 12.8.4 Linux Host Floppy Not Found

chapter 12.8.3, *Linux Host CD/DVD Drive Not Found (Older Distributions)*, page 292 appplies also to floppy disks, except that on older distributions Oracle VM VirtualBox tests for `/dev/fd*` devices by default. This can be overridden with the `VBOX_FLOPPY` environment variable.

### 12.8.5 Strange Guest IDE Error Messages When Writing to CD/DVD

If the experimental CD/DVD writer support is enabled with an incorrect Oracle VM VirtualBox, host or guest configuration, it is possible that any attempt to access the CD/DVD writer fails and simply results in guest kernel error messages for Linux guests or application error messages for Windows guests. Oracle VM VirtualBox performs the usual consistency checks when a VM is powered up. In particular, it aborts with an error message if the device for the CD/DVD writer is not writable by the user starting the VM. But Oracle VM VirtualBox cannot detect all misconfigurations. The necessary host and guest OS configuration is not specific for Oracle VM VirtualBox, but a few frequent problems are listed here which occurred in connection with Oracle VM VirtualBox.

Special care must be taken to use the correct device. The configured host CD/DVD device file name, in most cases `/dev/cdrom`, must point to the device that allows writing to the CD/DVD unit. For CD/DVD writer units connected to a SCSI controller or to a IDE controller that interfaces to the Linux SCSI subsystem, common for some SATA controllers, this must refer to the SCSI device node, such as `/dev/scd0`. Even for IDE CD/DVD writer units this must refer to the appropriate SCSI CD-ROM device node, such as `/dev/scd0`, if the `ide-scsi` kernel module is loaded. This module is required for CD/DVD writer support with all Linux 2.4 kernels and some early 2.6 kernels. Many Linux distributions load this module whenever a CD/DVD writer

is detected in the system, even if the kernel would support CD/DVD writers without the module. Oracle VM VirtualBox supports the use of IDE device files, such as `/dev/hdc`, provided the kernel supports this and the `ide-scsi` module is not loaded.

Similar rules, except that within the guest the CD/DVD writer is always an IDE device, apply to the guest configuration. Since this setup is very common, it is likely that the default configuration of the guest works as expected.

### 12.8.6 VBoxSVC IPC Issues

On Linux, Oracle VM VirtualBox makes use of a custom version of Mozilla XPCOM (cross platform component object model) for interprocess and intraprocess communication (IPC). The process `VBoxSVC` serves as a communication hub between different Oracle VM VirtualBox processes and maintains the global configuration, such as the XML database. When starting an Oracle VM VirtualBox component, the processes `VBoxSVC` and `VBoxXPCOMIPCD` are started automatically. They are only accessible from the user account they are running under. `VBoxSVC` owns the Oracle VM VirtualBox configuration database which normally resides in `~/.config/VirtualBox`, or the appropriate configuration directory for your operating system. While it is running, the configuration files are locked. Communication between the various Oracle VM VirtualBox components and `VBoxSVC` is performed through a local domain socket residing in `/tmp/.vbox-<username>-ipc`. In case there are communication problems, such as an Oracle VM VirtualBox application cannot communicate with `VBoxSVC`, terminate the daemons and remove the local domain socket directory.

### 12.8.7 USB Not Working

If USB is not working on your Linux host, make sure that the current user is a member of the `vboxusers` group. Please keep in mind that group membership does not take effect immediately but rather at the next login. If available, the `newgrp` command may avoid the need for a logout and login.

### 12.8.8 PAX/grsec Kernels

Linux kernels including the grsec patch, see `http://www.grsecurity.net/`, and derivates have to disable PAX_MPROTECT for the VBox binaries to be able to start a VM. The reason is that VBox has to create executable code on anonymous memory.

### 12.8.9 Linux Kernel vmalloc Pool Exhausted

When running a large number of VMs with a lot of RAM on a Linux system, say 20 VMs with 1 GB of RAM each, additional VMs might fail to start with a kernel error saying that the vmalloc pool is exhausted and should be extended. The error message also tells you to specify `vmalloc=256MB` in your kernel parameter list. If adding this parameter to your GRUB or LILO configuration makes the kernel fail to boot, with an error message such as "failed to mount the root partition", then you have probably run into a memory conflict of your kernel and initial RAM disk. This can be solved by adding the following parameter to your GRUB configuration:

```
uppermem 524288
```

## 12.9 Oracle Solaris Hosts

### 12.9.1 Cannot Start VM, Not Enough Contiguous Memory

The ZFS file system is known to use nearly all available RAM as cache if the default system settings are not changed. This may lead to a heavy fragmentation of the host memory preventing

Oracle VM VirtualBox VMs from being started. We recommend to limit the ZFS cache by adding the following line to /etc/system, where xxxx bytes is the amount of memory usable for the ZFS cache.

```
set zfs:zfs_arc_max = xxxx
```

### 12.9.2 VM Aborts With Out of Memory Errors on Oracle Solaris 10 Hosts

32-bit Oracle Solaris 10 hosts (bug 1225025) require swap space equal to, or greater than the host's physical memory size. For example, 8 GB physical memory would require at least 8 GB swap. This can be configured during an Oracle Solaris 10 install by choosing a Custom Install and changing the default partitions.

> **Note:** This restriction applies only to 32-bit Oracle Solaris hosts, 64-bit hosts are not affected.

For existing Oracle Solaris 10 installs, an additional swap image needs to be mounted and used as swap. Hence if you have 1 GB swap and 8 GB of physical memory, you require to add 7 GB more swap. This can be done as follows:

For ZFS, run the following as root user:

```
zfs create -V 8gb /_<ZFS volume>_/swap
swap -a /dev/zvol/dsk/_<ZFS volume>_/swap
```

To mount if after reboot, add the following line to /etc/vfstab:

```
/dev/zvol/dsk/_<ZFS volume>_/swap - - swap - no -
```

Alternatively, you could grow the existing swap using:

```
zfs set volsize=8G rpool/swap
```

And reboot the system for the changes to take effect.

For UFS (as root user):

```
mkfile 7g /path/to/swapfile.img
swap -a /path/to/swapfile.img
```

To mount it after reboot, add the following line to /etc/vfstab:

```
/path/to/swap.img - - swap - no -
```

# 13 Security Guide

## 13.1 General Security Principles

The following principles are fundamental to using any application securely.

- *Keep software up to date*. One of the principles of good security practise is to keep all software versions and patches up to date. Activate the Oracle VM VirtualBox update notification to get notified when a new Oracle VM VirtualBox release is available. When updating Oracle VM VirtualBox, do not forget to update the Guest Additions. Keep the host operating system as well as the guest operating system up to date.
- *Restrict network access to critical services*. Use proper means, for instance a firewall, to protect your computer and your guests from accesses from the outside. Choosing the proper networking mode for VMs helps to separate host networking from the guest and vice versa.
- *Follow the principle of least privilege*. The principle of least privilege states that users should be given the least amount of privilege necessary to perform their jobs. Always execute Oracle VM VirtualBox as a regular user. We strongly discourage anyone from executing Oracle VM VirtualBox with system privileges.

  Choose restrictive permissions when creating configuration files, for instance when creating /etc/default/virtualbox, see chapter 2.3.3.7, *Automatic Installation Options*, page 36. Mode 0600 is preferred.
- *Monitor system activity*. System security builds on three pillars: good security protocols, proper system configuration and system monitoring. Auditing and reviewing audit records address the third requirement. Each component within a system has some degree of monitoring capability. Follow audit advice in this document and regularly monitor audit records.
- *Keep up to date on latest security information*. Oracle continually improves its software and documentation. Check this note yearly for revisions.

## 13.2 Secure Installation and Configuration

### 13.2.1 Installation Overview

The Oracle VM VirtualBox base package should be downloaded only from a trusted source, for instance the official website `http://www.virtualbox.org`. The integrity of the package should be verified with the provided SHA256 checksum which can be found on the official website.

General Oracle VM VirtualBox installation instructions for the supported hosts can be found in chapter 2, *Installation Details*, page 29.

On Windows hosts, the installer can be used to disable USB support, support for bridged networking, support for host-only networking and the Python language binding. See chapter 2.1, *Installing on Windows Hosts*, page 29. All these features are enabled by default but disabling some of them could be appropriate if the corresponding functionality is not required by any virtual machine. The Python language bindings are only required if the Oracle VM VirtualBox API is to be used by external Python applications. In particular USB support and support for the two networking modes require the installation of Windows kernel drivers on the host. Therefore

disabling those selected features can not only be used to restrict the user to certain functionality but also to minimize the surface provided to a potential attacker.

The general case is to install the complete Oracle VM VirtualBox package. The installation must be done with system privileges. All Oracle VM VirtualBox binaries should be executed as a regular user and never as a privileged user.

The Oracle VM VirtualBox Extension Pack provides additional features and must be downloaded and installed separately, see chapter 1.6, *Installing Oracle VM VirtualBox and Extension Packs*, page 6. As for the base package, the SHA256 checksum of the extension pack should be verified. As the installation requires system privileges, Oracle VM VirtualBox will ask for the system password during the installation of the extension pack.

### 13.2.2 Post Installation Configuration

Normally there is no post installation configuration of Oracle VM VirtualBox components required. However, on Oracle Solaris and Linux hosts it is necessary to configure the proper permissions for users executing VMs and who should be able to access certain host resources. For instance, Linux users must be member of the *vboxusers* group to be able to pass USB devices to a guest. If a serial host interface should be accessed from a VM, the proper permissions must be granted to the user to be able to access that device. The same applies to other resources like raw partitions, DVD/CD drives, and sound devices.

## 13.3 Security Features

This section outlines the specific security mechanisms offered by Oracle VM VirtualBox.

### 13.3.1 The Security Model

One property of virtual machine monitors (VMMs) like Oracle VM VirtualBox is to encapsulate a guest by executing it in a protected environment, a virtual machine, running as a user process on the host operating system. The guest cannot communicate directly with the hardware or other computers but only through the VMM. The VMM provides emulated physical resources and devices to the guest which are accessed by the guest operating system to perform the required tasks. The VM settings control the resources provided to the guest, for example the amount of guest memory or the number of guest processors and the enabled features for that guest. For example remote control, certain screen settings and others. See chapter 3.4, *General Settings*, page 45.

### 13.3.2 Secure Configuration of Virtual Machines

Several aspects of a virtual machine configuration are subject to security considerations.

#### 13.3.2.1 Networking

The default networking mode for VMs is NAT which means that the VM acts like a computer behind a router, see chapter 6.3, *Network Address Translation (NAT)*, page 100. The guest is part of a private subnet belonging to this VM and the guest IP is not visible from the outside. This networking mode works without any additional setup and is sufficient for many purposes.

If bridged networking is used, the VM acts like a computer inside the same network as the host, see chapter 6.5, *Bridged Networking*, page 103. In this case, the guest has the same network access as the host and a firewall might be necessary to protect other computers on the subnet from a potential malicious guest as well as to protect the guest from a direct access from other computers. In some cases it is worth considering using a forwarding rule for a specific port in NAT mode instead of using bridged networking.

Some setups do not require a VM to be connected to the public network at all. Internal networking, see chapter 6.6, *Internal Networking*, page 104, or host-only networking, see chapter 6.7, *Host-Only Networking*, page 105, are often sufficient to connect VMs among each other or to connect VMs only with the host but not with the public network.

#### 13.3.2.2 VRDP Remote Desktop Authentication

When using the Oracle VM VirtualBox Extension Pack provided by Oracle for VRDP remote desktop support, you can optionally use various methods to configure RDP authentication. The "null" method is very insecure and should be avoided in a public network. See chapter 7.1.5, *RDP Authentication*, page 115.

#### 13.3.2.3 Clipboard

The shared clipboard enables users to share data between the host and the guest. Enabling the clipboard in Bidirectional mode enables the guest to read and write the host clipboard. The Host to Guest mode and the Guest to Host mode limit the access to one direction. If the guest is able to access the host clipboard it can also potentially access sensitive data from the host which is shared over the clipboard.

If the guest is able to read from and/or write to the host clipboard then a remote user connecting to the guest over the network will also gain this ability, which may not be desirable. As a consequence, the shared clipboard is disabled for new machines.

#### 13.3.2.4 Shared Folders

If any host folder is shared with the guest then a remote user connected to the guest over the network can access these files too as the folder sharing mechanism cannot be selectively disabled for remote users.

#### 13.3.2.5 3D Graphics Acceleration

Enabling 3D graphics using the Guest Additions exposes the host to additional security risks. See chapter 4.5.1, *Hardware 3D Acceleration (OpenGL and Direct3D 8/9)*, page 74.

#### 13.3.2.6 CD/DVD Passthrough

Enabling CD/DVD passthrough enables the guest to perform advanced operations on the CD/DVD drive, see chapter 5.9, *CD/DVD Support*, page 94. This could induce a security risk as a guest could overwrite data on a CD/DVD medium.

#### 13.3.2.7 USB Passthrough

Passing USB devices to the guest provides the guest full access to these devices, see chapter 3.11.1, *USB Settings*, page 56. For instance, in addition to reading and writing the content of the partitions of an external USB disk the guest will be also able to read and write the partition table and hardware data of that disk.

### 13.3.3 Configuring and Using Authentication

The following components of Oracle VM VirtualBox can use passwords for authentication:

- When using remote iSCSI storage and the storage server requires authentication, an initiator secret can optionally be supplied with the `VBoxManage storageattach` command. As long as no settings password is provided, by using the command line option

`--settingspwfile`, then this secret is stored *unencrypted* in the machine configuration and is therefore potentially readable on the host. See chapter 5.10, *iSCSI Servers*, page 95 and chapter 8.19, *VBoxManage storageattach*, page 160.

- When using the Oracle VM VirtualBox web service to control an Oracle VM VirtualBox host remotely, connections to the web service are authenticated in various ways. This is described in detail in the Oracle VM VirtualBox Software Development Kit (SDK) reference. See chapter 11, *Oracle VM VirtualBox Programming Interfaces*, page 276.

### 13.3.4 Potentially Insecure Operations

The following features of Oracle VM VirtualBox can present security problems:

- Enabling 3D graphics using the Guest Additions exposes the host to additional security risks. See chapter 4.5.1, *Hardware 3D Acceleration (OpenGL and Direct3D 8/9)*, page 74.
- When teleporting a machine, the data stream through which the machine's memory contents are transferred from one host to another is not encrypted. A third party with access to the network through which the data is transferred could therefore intercept that data. An SSH tunnel could be used to secure the connection between the two hosts. But when considering teleporting a VM over an untrusted network the first question to answer is how both VMs can securely access the same virtual disk image with a reasonable performance.
- When Page Fusion, see chapter 4.10.2, *Page Fusion*, page 81, is enabled, it is possible that a side-channel opens up that enables a malicious guest to determine the address space of another VM running on the same host layout. For example, where DLLs are typically loaded. This information leak in itself is harmless, however the malicious guest may use it to optimize attack against that VM through unrelated attack vectors. It is recommended to only enable Page Fusion if you do not think this is a concern in your setup.
- When using the Oracle VM VirtualBox web service to control an Oracle VM VirtualBox host remotely, connections to the web service, over which the API calls are transferred using SOAP XML, are not encrypted. They use plain HTTP by default. This is a potential security risk. For details about the web service, see chapter 11, *Oracle VM VirtualBox Programming Interfaces*, page 276.

  The web services are not started by default. See chapter 9.20, *Starting the Oracle VM VirtualBox Web Service Automatically*, page 242 to find out how to start this service and how to enable SSL/TLS support. It has to be started as a regular user and only the VMs of that user can be controlled. By default, the service binds to localhost preventing any remote connection.
- Traffic sent over a UDP Tunnel network attachment is not encrypted. You can either encrypt it on the host network level, with IPsec, or use encrypted protocols in the guest network, such as SSH. The security properties are similar to bridged Ethernet.
- Because of shortcomings in older Windows versions, using Oracle VM VirtualBox on Windows versions older than Vista with Service Pack 1 is not recommended.

### 13.3.5 Encryption

The following components of Oracle VM VirtualBox use encryption to protect sensitive data:

- When using the Oracle VM VirtualBox Extension Pack provided by Oracle for VRDP remote desktop support, RDP data can optionally be encrypted. See chapter 7.1.6, *RDP Encryption*, page 116. Only the Enhanced RDP Security method (RDP5.2) with TLS protocol provides a secure connection. Standard RDP Security (RDP4 and RDP5.1) is vulnerable to a man-in-the-middle attack.

## 13.4 Security Recommendations

This section contains security recommendations for specific issues. By default VirtualBox will configure the VMs to run in a secure manner, however this may not always be possible without additional user actions (e.g. host OS / firmware configuration changes).

### 13.4.1 CVE-2018-3646

This security issue affect a range of Intel CPUs with nested paging. AMD CPUs are expected not to be impacted (pending direct confirmation by AMD). Also the issue does not affect VMs running with hardware virtualization disabled or with nested paging disabled.

For more information about nested paging, see chapter 10.7, *Nested Paging and VPIDs*, page 275.

Mitigation options:

#### 13.4.1.1 Disable nested paging

By disabling nested paging (EPT), the VMM will construct page tables shadowing the ones in the guest. It is no possible for the guest to insert anything fishy into the page tables, since the VMM carefully validates each entry before shadowing it.

As a side effect of disabling nested paging, several CPU features will not be made available to the guest. Among these features are AVX, AVX2, XSAVE, AESNI, and POPCNT. Not all guests may be able to cope with dropping these features after installation. Also, for some guests, especially in SMP configurations, there could be stability issues arrising from disabling nested paging. Finally, some workloads may experience a performance degradation.

#### 13.4.1.2 Flushing the level 1 data cache

This aims at removing potentially sensitive data from the level 1 data cache when running guest code. However, it is made difficult by hyper-threading setups sharing the level 1 cache and thereby potentially letting the other thread in a pair refill the cache with data the user does not want the guest to see. In addition, flushing the level 1 data cache is usually not without performance side effects.

Up to date CPU microcode is a prerequisite for the cache flushing mitigations. Some host OSes may install these automatically, though it has traditionally been a task best performed by the system firmware. So, please check with your system / mainboard manufacturer for the latest firmware update.

We recommend disabling hyper threading on the host. This is traditionally done from the firmware setup, but some OSes also offers ways disable HT. In some cases it may be disabled by default, but please verify as the effectiveness of the mitigation depends on it.

The default action taken by VirtualBox is to flush the level 1 data cache when a thread is scheduled to execute guest code, rather than on each VM entry. This reduces the performance impact, while making the assumption that the host OS will not handle security sensitive data from interrupt handlers and similar without taking precautions.

A more aggressive flushing option is provided via the VBoxManage modifyvm option `--l1d-flush-on-vm-entry`. When enabled the level 1 data cache will be flushed on every VM entry. The performance impact is greater than with the default option, though this of course depends on the workload. Workloads producing a lot of VM exits (like networking, VGA access, and similiar) will probably be most impacted.

For users not concerned by this security issue, the default mitigation can be disabled using

```
VBoxManage modifyvm name --l1d-flush-on-sched off
```

### 13.4.2 CVE-2018-12126, CVE-2018-12127, CVE-2018-12130, CVE-2019-11091

These security issues affect a range of Intel CPUs starting with Nehalem. The CVE-2018-12130 also affects some Atom Silvermont, Atom Airmont, and Knights family CPUs, however the scope is so limited that the host OS should deal with it for us and VBox therefore not be affected (leaks only happens when entering and leaving C states).

Mitigation option:

#### 13.4.2.1 Buffer overwriting and disabling HT

First, up to date CPU microcode is a prerequisite for the buffer overwriting (clearing) mitigations. Some host OSes may install these automatically, though it has traditionally been a task best performed by the system firmware. So, please check with your system / mainboard manufacturer for the latest firmware update.

This mitigation aims at removing potentially sensitive data from the affected buffers before running guest code. Since this means additional work each time the guest is scheduled, there might be some performance side effects.

We recommend disabling hyper threading on host affected by CVE-2018-12126 and CVE-2018-12127 because the affected sets of buffers are normally shared between thread pairs and therefore cause leaks between the threads. This is traditionally done from the firmware setup, but some OSes also offers ways disable HT. In some cases it may be disabled by default, but please verify as the effectiveness of the mitigation depends on it.

The default action taken by VirtualBox is to clear the affected buffers when a thread is scheduled to execute guest code, rather than on each VM entry. This reduces the performance impact, while making the assumption that the host OS will not handle security sensitive data from interrupt handlers and similar without taking precautions.

A more aggressive flushing option is provided via the VBoxManage modifyvm option `--mds-clear-on-vm-entry`. When enabled the affected buffers will be cleared on every VM entry. The performance impact is greater than with the default option, though this of course depends on the workload. Workloads producing a lot of VM exits (like networking, VGA access, and similiar) will probably be most impacted.

For users not concerned by this security issue, the default mitigation can be disabled using

```
VBoxManage modifyvm name --mds-clear-on-sched off
```

# 14 Known Limitations

## 14.1 Experimental Features

Some Oracle VM VirtualBox features are labeled as experimental. Such features are provided on an "as-is" basis and are not formally supported. However, feedback and suggestions about such features are welcome. A comprehensive list of experimental features is as follows:

- Hardware 3D acceleration support for Windows, Linux, and Oracle Solaris guests
- Hardware 2D video playback acceleration support for Windows guests
- PCI pass-through (Linux hosts only)
- Mac OS X guests (Mac OS X hosts only)
- ICH9 chipset emulation
- EFI firmware
- Host CD/DVD drive pass-through
- Support of iSCSI using internal networking
- Using Oracle VM VirtualBox and Hyper-V on the same host

## 14.2 Known Issues

The following section describes known problems with this release of Oracle VM VirtualBox. Unless marked otherwise, these issues are planned to be fixed in later releases.

- The following **Guest SMP (multiprocessor) limitations** exist:
  - **Poor performance** with 32-bit guests on AMD CPUs. This affects mainly Windows and Oracle Solaris guests, but possibly also some Linux kernel revisions. Partially solved in 3.0.6 for 32-bit Windows NT, 2000, XP, and 2003 guests. Requires Guest Additions 3.0.6 or later to be installed.
  - **Poor performance** with 32-bit guests on certain Intel CPU models that do not include virtual APIC hardware optimization support. This affects mainly Windows and Oracle Solaris guests, but possibly also some Linux kernel revisions. Partially solved in 3.0.12 for 32-bit Windows NT, 2000, XP, and 2003 guests. Requires Guest Additions 3.0.12 or later to be installed.
- **NX (no execute, data execution prevention)** only works for guests running on 64-bit hosts or guests running on 32-bit hosts with PAE enabled and requires that hardware virtualization be enabled.
- For **basic Direct3D support in Windows guests** to work, the Guest Additions must be installed in Windows safe mode. Press F8 when the Windows guest is booting and select **Safe Mode**, then install the Guest Additions. Otherwise the Windows file protection mechanism will interfere with the replacement DLLs installed by Oracle VM VirtualBox and keep restoring the original Windows system DLLs.

**Note:** This does *not* apply to the WDDM Direct3D video driver available for Vista and Windows 7 guests shipped with Oracle VM VirtualBox 4.1.

- **Guest control.** On Windows guests, a process started using the guest control execute support will not be able to display a graphical user interface *unless* the user account under which it is running is currently logged in and has a desktop session.

  Also, to use accounts without or with an empty password, the guest's group policy must be changed. To do so, open the group policy editor on the command line by typing `gpedit.msc`, open the key `Computer Configuration\Windows Settings\Security Settings\Local Policies\Security Options` and change the value of `Accounts: Limit local account use of blank passwords to console logon only` to Disabled.

- **Compacting virtual disk images is limited to VDI files.** The `VBoxManage modifyhd --compact` command is currently only implemented for VDI files. At the moment the only way to optimize the size of a virtual disk images in other formats, such as VMDK or VHD, is to clone the image and then use the cloned image in the VM configuration.

- **OVF import/export:**
  - OVF localization, with multiple languages in a single OVF file, is not yet supported.
  - Some OVF sections like StartupSection, DeploymentOptionSection, and InstallSection are ignored.
  - OVF environment documents, including their property sections and appliance configuration with ISO images, are not yet supported.
  - Remote files using HTTP or other mechanisms are not yet supported.

- Neither **scale mode** nor **seamless mode** work correctly with guests using OpenGL 3D features, such as with compiz-enabled window managers.

- The RDP server in the Oracle VM VirtualBox extension pack supports only audio streams in format 22.05kHz stereo 16 bit. If the RDP client requests any other audio format there will be no audio.

- Preserving the aspect ratio in scale mode works only on Windows hosts and on Mac OS X hosts.

- On **Mac OS X hosts,** the following features are not yet implemented:
  - Numlock emulation
  - CPU frequency metric
  - Memory ballooning

- **Mac OS X guests:**
  - Mac OS X guests can only run on a certain host hardware. For details about license and host hardware limitations. See chapter 3.1.1, *Mac OS X Guests*, page 41 and check the Apple software license conditions.
  - Oracle VM VirtualBox does not provide Guest Additions for Mac OS X at this time.
  - The graphics resolution currently defaults to 1024x768 as Mac OS X falls back to the built-in EFI display support. See chapter 3.14.1, *Video Modes in EFI*, page 59 for more information on how to change EFI video modes.
  - Mac OS X guests only work with one CPU assigned to the VM. Support for SMP will be provided in a future release.

  - Depending on your system and version of Mac OS X, you might experience guest hangs after some time. This can be fixed by turning off energy saving. Set timeout to "Never" in the system preferences.
  - By default, the Oracle VM VirtualBox EFI enables debug output of the Mac OS X kernel to help you diagnose boot problems. Note that there is a lot of output and not all errors are fatal. They would also show when using a physical Apple Macintosh computer. You can turn off these messages by using the following command:

    ```
    VBoxManage setextradata "VM name" "VBoxInternal2/EfiBootArgs" "  "
    ```

    To revert to the previous behavior, use the following command:

    ```
    VBoxManage setextradata "VM name" "VBoxInternal2/EfiBootArgs" ""
    ```

  - It is currently not possible to start a Mac OS X guest in safe mode by specifying "-x" option in "VBoxInternal2/EfiBootArgs" extradata.

- **Oracle Solaris hosts:**
  - There is no support for USB devices connected to Oracle Solaris 10 hosts.
  - USB support on Oracle Solaris hosts requires Oracle Solaris 11 version snv_124 or later. Webcams and other isochronous devices are known to have poor performance.
  - Host Webcam passthrough is restricted to 640x480 frames at 20 frames per second due to limitations in the Oracle Solaris V4L2 API. This may be addressed in a future Oracle Solaris release.
  - No ACPI information, such as battery status or power source, is reported to the guest.
  - No support for using wireless adapters with bridged networking.
  - Crossbow-based bridged networking on Oracle Solaris 11 hosts does not work directly with aggregate links. However, you can use `dladm` to manually create a VNIC over the aggregate link and use that with a VM. This limitation does not exist in Oracle Solaris 11u1 build 17 and newer.

- **Guest Additions of version 4.1, 4.1.2 and 4.1.4 for Windows.** The Oracle VM VirtualBox WDDM Video driver may be installed and remain in the guest system when Guest additions uninstallation is performed. This is caused by a bug in Guest Additions uninstaller.

  **Note:** This does *not* apply to a Guest Additions update. Installing one version of Guest Additions on top of another works correctly.

  To solve this problem, uninstall the Oracle VM VirtualBox WDDM Video driver manually. Open Device Manager, and check whether the Display Adapter is named "Oracle VM VirtualBox Graphics Adapter ..". If not, there is nothing to be done. If it is, right-click the Oracle VM VirtualBox Graphics Adapter in Device Manager, select **Uninstall**, check **Delete the Driver Software for this Device** and click **OK**. Once uninstallation is done, start Device Manager, go to the **Action** menu and select **Scan for Hardware Change**s to ensure that the correct Windows default driver be picked up for the Graphics adapter.

- Neither *virtio* nor *Intel PRO/1000* drivers for **Windows XP guests** support segmentation offloading. Therefore Windows XP guests have slower transmission rates comparing to other guest types. Refer to MS Knowledge base article 842264 for additional information.

- **Guest Additions for OS/2.** Seamless windows and automatic guest resizing will probably never be implemented due to inherent limitations of the OS/2 graphics system.

- Some guest operating systems predating ATAPI CD-ROMs may exhibit long delays or entirely fail to boot in certain configurations. This is most likely to happen when an IDE/ATAPI CD-ROM exists alone on a primary or secondary IDE channel.

  Affected operating systems are MS OS/2 1.21: fails to boot with an error message referencing COUNTRY.SYS and MS OS/2 1.3: long boot delays. To avoid such problems, disable the emulated IDE/ATAPI CD-ROM. The guest OS cannot use this device, anyway.

# 15 Change Log

This section summarizes the changes between Oracle VM VirtualBox versions. Note that this change log is not exhaustive and not all changes are listed.

Oracle VM VirtualBox version numbers consist of three numbers separated by dots where the first and second number represent the major version and the third number the minor version. Minor version numbers of official releases are always even. An odd minor version number represents an internal development or test build. In addition, each build contains a revision number.

## 15.1 Version 6.0.8 (2019-05-13)

This is a maintenance release. The following items were fixed and/or added:

- Core: fix saved state resume failures (bugs #18265 and #18331)
- User interface: show full file location in New Medium window.
- User interface: fix mouse click pass-through problems in multi-screen virtual machines (6.0.6 regression, bug #18567)
- Graphics: fixed a crash when powering off a VM without graphics controller (bug #18570)
- API: partial fix for dealing with VM config conflicting with other VMs related to medium UUIDs, now correctly flags VM as inaccessible (bug #17908)
- Windows hosts: Support paths longer than 4096 characters on in shared folders
- Linux hosts: fix kernel module build breakage in non-default build set-ups (bug #18620, thank you Ambroz Bizjak)
- Linux hosts: fix kernel module build breakage in debug build set-ups (bug #18621, thank you Ambroz Bizjak)
- Windows guests: notice file size increases in shared folders which were missed in certain cases
- Linux guests: make shared folders work with Linux 3.16.35
- Linux guests: fix incorrectly read-only shared folders (bug #18345)

## 15.2 Version 6.0.6 (2019-04-17)

This is a maintenance release. The following items were fixed and/or added:

- Virtualization core: nested AMD virtualization fixes
- User interface: fixed copying directories in file manager
- User interface: fixed operation progress in file manager when copying content
- User interface: fixed operation progress when deleting snapshots

- User interface: fixed unattended installation of recent Ubuntu guests
- User interface: new virtual disk sector sizes should be divisible by 512 (bug #18177)
- User interface: various additional improvements
- Storage: fixed loading saved states for LsiLogic devices (6.0.0 regression; bug #18263)
- Storage: fixed fixed reading certain QCOW2 images and support version 3 of the format readonly
- Storage: Improved IDE PCI emulation to allow NetWare IDE drivers to use bus-mastering
- Graphics: Improved VMSVGA support to work with old X servers which previously showed only a badly scrambled screen
- Graphics: fixed invisible mouse cursor with VMSVGA emulation and without mouse integration (bug #18239)
- Graphics: make EFI work with VMSVGA emulation (bug #18282)
- Graphics: remember last guest screen size VMSVGA emulation (bug #18408)
- Graphics: fix RDP to guests using VMSVGA emulation (bug #18518)
- Graphics: various additional VMSVGA emulation fixes
- Audio: implemented audio device enumeration for the DirectSound backend
- Network: fixed unwanted padding bytes in Windows host adaptor network packets (bug #18202 and bug #18355)
- Serial: fixed possible crash on Windows when using a host device (6.0.0 regression; bug #18319)
- Serial: fixed loopback handling in the emulation causing garbage to be sent during boot with Linux guests (6.0.0 regression; bug #18319)
- Shared folders: fixed duplicate folders after restoring a saved state (bug #18373)
- Shared folders: fixed hangs inside guest (bug #18151)
- Drag and drop: fixed copying files from guest host (bug #18305)
- Recording: fixed modifying settings via VBoxManage (bug #18494)
- Fixed invalid Extra Data characters making machines inaccessible
- VBoxManage: crash fix (bug #18341)
- Fixed hangs during failed virtual machine start-up
- Windows host: fix unwanted wake-ups from standby, thank you Daniel Karsubka (bug #18549)
- Windows host: disallow Pause as a host key (bug #18482)
- Linux host and guest: support Linux 5.0 and 5.1, thank you Valdis Kletnieks (see also bug #18515)
- Linux host: support kernel 4.4.169 (bug #18315)
- Linux host: fix logging when building Linux kernel modules (bug #18226)

- Linux host: clarified building Linux host drivers with secure boot (bug #18312)
- FreeBSD host: compilation fixes
- Installers: reduced size of packages
- Web services: work with Java 11
- LibreSSL compilation fix, thank you Stefan Strogin
- Windows guests: fixed running applications which use complex display topologies with WDDM driver, fixed Skype for Business hangs (bug #17092)
- Windows guests: fixed an occasional guest crash with WDDM driver and VBoxSVGA adapter (bug #18369)
- Windows guests: shared folder file creation detection issue (bug #9276)
- Windows guests: fixed "mismatched pool allocation/free" error with checked builds of Windows and ReactOS (bug #18187, thank you Adam Stachowicz)
- Linux guests: shared folder performance and reliability improvements and missing features (bugs #17360, #819)
- Solaris guests: fix an error message from VBoxClient (bug #18428)

## 15.3 Version 6.0.4 (2019-01-28)

This is a maintenance release. The following items were fixed and/or added:

- Virtualization core: support Shanghai/Zhaoxin CPUs.
- User interface: handle command line arguments to VirtualBox correctly (bugs #18206 and #18197)
- User interface: improvements to machine manager window, virtual optical disk creator, storage selector window and log viewer window
- User interface: various small fixes and improvements
- User interface: fix incorrect restoring of main window position (bug #18367)
- Audio: implemented time scheduling for the AC'97 device emulation to keep audio and video in sync
- Graphics: basic support for VMSVGA graphics device in virtual machines using EFI
- Network: fix occasional NATNet crashes (bug #13899)
- Network: worked around problems in certain PCnet drivers on old operating systems
- Serial: fixed connecting to pseudo terminals on POSIX hosts (6.0.0 regression; bug #18319)
- Linux hosts and guests: fix for building kernel modules against Linux 5.0. Thank you Kyle Laker

## 15.4 Version 6.0.2 (2019-01-15)

This is a maintenance release. The following items were fixed and/or added:

- User interface: fixed creation of desktop shortcuts for starting virtual machines (bug #18207)
- User interface: allow the first run window to selecting host drives (bug #18230)
- User interface: fixed attaching empty host optical drives (bug #18223)
- User interface: implemented a new virtual optical disk creation window
- USB: modified Linux backends to reset USB devices (previously, most guest attempts to reset USB devices were ignored)
- PCnet: fixed a regression which caused some PCnet PCI guest drivers to not detect the emulated hardware (bug #18286)
- Linux hosts: Skip device enumeration if PulseAudio interface is not available
- Linux hosts: fixed conflict between Debian and Oracle build desktop files (bug #18264)
- Linux and MacOS hosts: VirtualBoxVM command not accessible (bug #18257)
- Windows guests: multiple monitor fixes with VBoxSVGA graphics
- Windows guests: black screen with VBoxSVGA graphics when 3D is disabled (bug #18205)
- Linux guests: fixed building drivers on SLES 12.4 (bug #18213)
- Linux guests: fixed building shared folder driver with older kernels (bug #18238)
- OS/2 shared folders: fixed write regression introduced in 6.0.0 GA

## 15.5 Version 6.0.0 (2018-12-18)

This is a major update. The following major new features were added:

- Implemented support for exporting a virtual machine to Oracle Cloud Infrastructure
- User interface: greatly improved HiDPI and scaling support, including better detection and per-machine configuration
- Major rework of user interface with simpler and more powerful application and virtual machine set-up
- User interface: a new file manager enabling user to control the guest file system and copy files between host and guest.
- Graphics: major update of 3D graphics support for Windows guests, and VMSVGA 3D graphics device emulation on Linux and Solaris guests
- Added support for surround speaker setups (as used by Windows 10 Build 1809)
- Added utility `vboximg-mount` on Apple hosts to access the content of guest disks on the host
- Added support for using Hyper-V as the fallback execution core on Windows host, to avoid inability to run VMs at the price of reduced performance

In addition, the following items were fixed and/or added:

- Execution core: fixed single-stepping in certain circumstances (bug #17316)
- User interface: video and audio recording can now be separately enabled
- Audio/Video recording fixes and improvements
- Audio: better support for attaching and detaching remote desktop connections
- Serial port emulation fixes
- Serial ports: allow changing the serial port attachment while a machine is running (bug #6115)
- Networking: Added a workaround for older guests which do not enable bus mastering for the virtio PCI device
- Networking: fixed wrong RCODE from DNS AAAA query with –natdnshostresolver1 (bug #18171)
- iSCSI: In cases where there is no ambiguity, the LUN of an iSCSI target is automatically determined, for targets with non-zero LUNs
- Transparently resize disk images when merging if possible
- VBoxManage: support for DHCP options
- Fixed VNC/RDP (bug #18153)
- Guest Control: various new interfaces and features (see SDK documentation)
- Linux hosts: support Linux 4.20 (thank you Larry Finger)
- Solaris: installer fixes
- Shared folders: performance improvements
- Guest Additions: improved shared folder auto-mounting
- Windows Guest Additions: fix incorrect tablet co-ordinate handling with recent Windows 10 builds
- Linux Additions: fix for building vboxvideo on EL 7.6 standard kernel, contributed by Robert Conde (bug #18093)
- Linux guests: support Linux 4.20 (thank you Larry Finger)
- Linux guests: support VMSVGA in the Linux and X11 Additions
- MacOS Guest Additions: initial support
- OS/2 Guest Additions: initial shared folder support
- BIOS fixes
- ACPI: Up to four custom ACPI tables can now be configured for a VM

## 15.6 Change Logs for Legacy Versions

To view the change log for a legacy version of VirtualBox see the documentation for the relevant Oracle VM VirtualBox release.

Change logs are also available at:
https://www.virtualbox.org/wiki/Changelog.

# 16 Third-Party Materials and Licenses

Oracle VM VirtualBox incorporates materials from several Open Source software projects. Therefore the use of these materials by Oracle VM VirtualBox is governed by different Open Source licenses. This document reproduces these licenses and provides a list of the materials used and their respective licensing conditions. Section 1 contains a list of the materials used. Section 2 reproduces the applicable Open Source licenses. For each material, a reference to its license is provided.

The source code for the materials listed below as well as the rest of the Oracle VM VirtualBox code which is released as open source are available at `http://www.virtualbox.org`, both as tarballs for particular releases and as a live SVN repository.

## 16.1 Third-Party Materials

- Oracle VM VirtualBox contains portions of QEMU which is governed by the licenses in chapter 16.2.5, *X Consortium License (X11)*, page 328 and chapter 16.2.2, *GNU Lesser General Public License (LGPL)*, page 317 and

  (C) 2003-2005 Fabrice Bellard; Copyright (C) 2004-2005 Vassili Karpov (malc); Copyright (c) 2004 Antony T Curtis; Copyright (C) 2003 Jocelyn Mayer

- Oracle VM VirtualBox contains code which is governed by the license in chapter 16.2.5, *X Consortium License (X11)*, page 328 and

  Copyright 2004 by the Massachusetts Institute of Technology.

- Oracle VM VirtualBox contains code of the BOCHS VGA BIOS which is governed by the license in chapter 16.2.2, *GNU Lesser General Public License (LGPL)*, page 317 and

  Copyright (C) 2001, 2002 the LGPL VGABios developers Team.

- Oracle VM VirtualBox contains code of the BOCHS ROM BIOS which is governed by the license in chapter 16.2.2, *GNU Lesser General Public License (LGPL)*, page 317 and

  Copyright (C) 2002 MandrakeSoft S.A.; Copyright (C) 2004 Fabrice Bellard; Copyright (C) 2005 Struan Bartlett.

- Oracle VM VirtualBox contains the zlib library which is governed by the license in chapter 16.2.6, *zlib License*, page 328 and

  Copyright (C) 1995-2003 Jean-loup Gailly and Mark Adler.

- Oracle VM VirtualBox may contain OpenSSL which is governed by the license in chapter 16.2.7, *OpenSSL License*, page 329 and

  Copyright (C) 1995-1998 Eric Young (eay@cryptsoft.com). This product includes software written by Tim Hudson (tjh@cryptsoft.com).

- Oracle VM VirtualBox may contain NSPR and XPCOM which is governed by the license in chapter 16.2.3, *Mozilla Public License (MPL)*, page 322 and

  Copyright (C) The Authors.

- Oracle VM VirtualBox contains Slirp which is governed by the license in chapter 16.2.8, *Slirp License*, page 330 and was written by Danny Gasparovski.

  Copyright (C) 1995, 1996 All Rights Reserved.

- Oracle VM VirtualBox contains liblzf which is governed by the license in chapter 16.2.9, *liblzf License*, page 330 and

  Copyright (C) 2000-2005 Marc Alexander Lehmann <schmorp@schmorp.de>

- Oracle VM VirtualBox may ship with a modified copy of rdesktop which is governed by the license in chapter 16.2.1, *GNU General Public License (GPL)*, page 313 and

  Copyright (C) Matthew Chapman and others.

- Oracle VM VirtualBox may ship with a copy of kchmviewer which is governed by the license in chapter 16.2.1, *GNU General Public License (GPL)*, page 313 and

  Copyright (C) George Yunaev and others.

- Oracle VM VirtualBox may contain Etherboot which is governed by the license in chapter 16.2.1, *GNU General Public License (GPL)*, page 313 with the exception that aggregating Etherboot with another work does not require the other work to be released under the same license (see `http://etherboot.sourceforge.net/clinks.html`). Etherboot is

  Copyright (C) Etherboot team.

- Oracle VM VirtualBox may contain iPXE which is governed by the license in chapter 16.2.1, *GNU General Public License (GPL)*, page 313 and

  Copyright (C) Michael Brown <mbrown@fensystems.co.uk> and others.

- Oracle VM VirtualBox contains code from Wine which is governed by the license in chapter 16.2.2, *GNU Lesser General Public License (LGPL)*, page 317 and

  Copyright 1993 Bob Amstadt, Copyright 1996 Albrecht Kleine, Copyright 1997 David Faure, Copyright 1998 Morten Welinder, Copyright 1998 Ulrich Weigand, Copyright 1999 Ove Koven

- Oracle VM VirtualBox contains code from lwIP which is governed by the license in chapter 16.2.11, *lwIP License*, page 331 and

  Copyright (C) 2001, 2002 Swedish Institute of Computer Science.

- Oracle VM VirtualBox contains libxml which is governed by the license in chapter 16.2.12, *libxml License*, page 331 and

  Copyright (C) 1998-2003 Daniel Veillard.

- Oracle VM VirtualBox contains libxslt which is governed by the license in chapter 16.2.13, *libxslt Licenses*, page 332 and

  Copyright (C) 2001-2002 Daniel Veillard and Copyright (C) 2001-2002 Thomas Broyer, Charlie Bozeman and Daniel Veillard.

- Oracle VM VirtualBox contains code from the gSOAP XML web services tools, which are licensed under the license in chapter 16.2.14, *gSOAP Public License Version 1.3a*, page 332 and

  Copyright (C) 2000-2007, Robert van Engelen, Genivia Inc., and others.

- Oracle VM VirtualBox ships with the application tunctl (shipped as VBoxTunctl) from the User-mode Linux suite which is governed by the license in chapter 16.2.1, *GNU General Public License (GPL)*, page 313 and

  Copyright (C) 2002 Jeff Dike.

- Oracle VM VirtualBox contains code from Chromium, an OpenGL implementation, which is goverened by the licenses in chapter 16.2.15, *Chromium Licenses*, page 337 and

  Copyright (C) Stanford University, The Regents of the University of California, Red Hat, and others.

- Oracle VM VirtualBox contains libcurl which is governed by the license in chapter 16.2.16, *curl License*, page 339 and

  Copyright (C) 1996-2009, Daniel Stenberg.

- Oracle VM VirtualBox contains dnsproxy which is governed by the license in chapter 16.2.4, *MIT License*, page 328 and

  Copyright (c) 2003, 2004, 2005 Armin Wolfermann.

- Oracle VM VirtualBox may contain iniparser which is governed by the license in chapter 16.2.4, *MIT License*, page 328 and

  Copyright (c) 2000-2008 by Nicolas Devillard.

- Oracle VM VirtualBox contains some code from libgd which is governed by the license in chapter 16.2.17, *libgd License*, page 340 and

  Copyright 2000, 2001, 2002, 2003, 2004, 2005, 2006, 2007 Pierre-Alain Joye (pierre@libgd.org).

- Oracle VM VirtualBox contains code from the EFI Development Kit II which is governed by the license in chapter 16.2.18, *BSD License from Intel*, page 340 and

  Copyright (c) 2004-2008, Intel Corporation.

- Oracle VM VirtualBox contains libjpeg which is governed by the license in chapter 16.2.19, *libjpeg License*, page 341 and

  Copyright (C) 1991-2010, Thomas G. Lane, Guido Vollbeding.

- Oracle VM VirtualBox may contain x86 SIMD extension for IJG JPEG library which is governed by the license in chapter 16.2.20, *x86 SIMD Extension for IJG JPEG Library License*, page 341 and

  Copyright 2009 Pierre Ossman <ossman@cendio.se> for Cendio AB; Copyright 2010 D. R. Commander; Copyright (C) 1999-2006, MIYASAKA Masaru.

- Oracle VM VirtualBox may ship a copy of Qt which is governed by the license in chapter 16.2.2, *GNU Lesser General Public License (LGPL)*, page 317 and

  Copyright (C) 2010, 2011 Nokia Corporation and/or its subsidiary(-ies).

- Oracle VM VirtualBox contains parts of the FreeBSD kernel which is governed by the license in chapter 16.2.21, *FreeBSD License*, page 342.

- Oracle VM VirtualBox contains parts of the NetBSD kernel which is governed by the license in chapter 16.2.22, *NetBSD License*, page 342.

- Oracle VM VirtualBox contains portions of liblightdm-gobject which is governed by the license in chapter 16.2.2, *GNU Lesser General Public License (LGPL)*, page 317 and

  Copyright (C) 2010-2013 Canonical Ltd.; Copyright (C) 2010-2011 Robert Ancell.

- Oracle VM VirtualBox contains portions of glib which is governed by the license in chapter 16.2.2, *GNU Lesser General Public License (LGPL)*, page 317 and

  Copyright (C) 1995-2011 The Glib team

- Oracle VM VirtualBox contains portions of PCRE which is governed by the license in chapter 16.2.23, *PCRE License*, page 343 and

  Copyright (c) 1997-2012 University of Cambridge; Copyright(c) 2009-2012 Zoltan Herczeg; Copyright (c) 2007-2012, Google Inc.

- Oracle VM VirtualBox contains portions of libffi which is governed by the license in chapter 16.2.24, *libffi License*, page 344 and

  Copyright (c) 1996-2012 Anthony Green, Red Hat, Inc and others. See source files for details.

- Oracle VM VirtualBox contains portions of FLTK which is governed by the licenses in chapter 16.2.25, *FLTK License*, page 344 and chapter 16.2.2, *GNU Lesser General Public License (LGPL)*, page 317 and

  Copyright (C) 1991-2012 The FLTK team

- Oracle VM VirtualBox contains portions of Expat which is governed by the license in chapter 16.2.26, *Expat License*, page 344 and

  Copyright (c) 1998, 1999, 2000 Thai Open Source Software Center Ltd and Clark Cooper; Copyright (c) 2001, 2002, 2003, 2004, 2005, 2006 Expat maintainers.

- Oracle VM VirtualBox contains portions of Fontconfig which is governed by the license in chapter 16.2.27, *Fontconfig License*, page 345 and

  Copyright (C) 2001, 2003 Keith Packard

- Oracle VM VirtualBox contains portions of Freetype which is governed by the license in chapter 16.2.28, *Freetype License*, page 345 and

  Copyright 2012 The FreeType Project (www.freetype.org). All rights reserved.

- Oracle VM VirtualBox may contain code from the WebM VP8 Codec SDK which is governed by the license in chapter 16.2.29, *VPX License*, page 347 and

  Copyright (c) 2010, The WebM Project authors. All rights reserved.

- Oracle VM VirtualBox may contain code from libopus (“Opus”) which is governed by the license in chapter 16.2.30, *Opus License*, page 347 and

  Copyright 2001-2011 Xiph.Org, Skype Limited, Octasic, Jean-Marc Valin, Timothy B. Terriberry, CSIRO, Gregory Maxwell, Mark Borgerding, Erik de Castro Lopo

- Oracle VM VirtualBox may contain portions of FUSE for macOS which is governed by the licenses in chapter 16.2.31, *FUSE for macOS License*, page 348 and chapter 16.2.2, *GNU Lesser General Public License (LGPL)*, page 317 and

  Copyright (c) 2011-2017 Benjamin Fleischer; Copyright (c) 2011-2012 Erik Larsson All rights reserved.

## 16.2 Third-Party Licenses

### 16.2.1 GNU General Public License (GPL)

GNU GENERAL PUBLIC LICENSE Version 2, June 1991
Copyright (C) 1989, 1991 Free Software Foundation, Inc.
51 Franklin St, Fifth Floor, Boston, MA 02110-1301 USA
Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.
Preamble
The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change free software–to make sure the software is free for all its users. This General Public License applies to most of the Free Software Foundation’s software and to any other program

whose authors commit to using it. (Some other Free Software Foundation software is covered by the GNU Library General Public License instead.) You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs; and that you know you can do these things.

To protect your rights, we need to make restrictions that forbid anyone to deny you these rights or to ask you to surrender the rights. These restrictions translate to certain responsibilities for you if you distribute copies of the software, or if you modify it.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must give the recipients all the rights that you have. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

We protect your rights with two steps: (1) copyright the software, and (2) offer you this license which gives you legal permission to copy, distribute and/or modify the software.

Also, for each author's protection and ours, we want to make certain that everyone understands that there is no warranty for this free software. If the software is modified by someone else and passed on, we want its recipients to know that what they have is not the original, so that any problems introduced by others will not reflect on the original authors' reputations.

Finally, any free program is threatened constantly by software patents. We wish to avoid the danger that redistributors of a free program will individually obtain patent licenses, in effect making the program proprietary. To prevent this, we have made it clear that any patent must be licensed for everyone's free use or not licensed at all.

The precise terms and conditions for copying, distribution and modification follow.

GNU GENERAL PUBLIC LICENSE TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License applies to any program or other work which contains a notice placed by the copyright holder saying it may be distributed under the terms of this General Public License. The "Program", below, refers to any such program or work, and a "work based on the Program" means either the Program or any derivative work under copyright law: that is to say, a work containing the Program or a portion of it, either verbatim or with modifications and/or translated into another language. (Hereinafter, translation is included without limitation in the term "modification".) Each licensee is addressed as "you".

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running the Program is not restricted, and the output from the Program is covered only if its contents constitute a work based on the Program (independent of having been made by running the Program). Whether that is true depends on what the Program does.

1. You may copy and distribute verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and give any other recipients of the Program a copy of this License along with the Program.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Program or any portion of it, thus forming a work based on the Program, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

a) You must cause the modified files to carry prominent notices stating that you changed the files and the date of any change.

b) You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License.

c) If the modified program normally reads commands interactively when run, you must cause it, when started running for such interactive use in the most ordinary way, to print or display an announcement including an appropriate copyright notice and a notice that there is no warranty (or else, saying that you provide a warranty) and that users may redistribute the program under these conditions, and telling the user how to view a copy of this License. (Exception: if the Program itself is interactive but does not normally print such an announcement, your work based on the Program is not required to print an announcement.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Program, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Program, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Program.

In addition, mere aggregation of another work not based on the Program with the Program (or with a work based on the Program) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may copy and distribute the Program (or a work based on it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you also do one of the following:

a) Accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,

b) Accompany it with a written offer, valid for at least three years, to give any third party, for a charge no more than your cost of physically performing source distribution, a complete machine-readable copy of the corresponding source code, to be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,

c) Accompany it with the information you received as to the offer to distribute corresponding source code. (This alternative is allowed only for noncommercial distribution and only if you received the program in object code or executable form with such an offer, in accord with Subsection b above.)

The source code for a work means the preferred form of the work for making modifications to it. For an executable work, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the executable. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

If distribution of executable or object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place counts as distribution of the source code, even though third parties are not compelled to copy the source along with the object code.

4. You may not copy, modify, sublicense, or distribute the Program except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense or distribute the Program is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

5. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Program or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or

distributing the Program (or any work based on the Program), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Program or works based on it.

6. Each time you redistribute the Program (or any work based on the Program), the recipient automatically receives a license from the original licensor to copy, distribute or modify the Program subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.

7. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Program at all. For example, if a patent license would not permit royalty-free redistribution of the Program by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Program.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system, which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

8. If the distribution and/or use of the Program is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Program under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

9. The Free Software Foundation may publish revised and/or new versions of the General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of this License, you may choose any version ever published by the Free Software Foundation.

10. If you wish to incorporate parts of the Program into other free programs whose distribution conditions are different, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

11. BECAUSE THE PROGRAM IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IM-

PLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

12. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

### 16.2.2 GNU Lesser General Public License (LGPL)

GNU LESSER GENERAL PUBLIC LICENSE Version 2.1, February 1999

Copyright (C) 1991, 1999 Free Software Foundation, Inc. 59 Temple Place, Suite 330, Boston, MA 02111-1307 USA Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.

[This is the first released version of the Lesser GPL. It also counts as the successor of the GNU Library Public License, version 2, hence the version number 2.1.]

Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public Licenses are intended to guarantee your freedom to share and change free software–to make sure the software is free for all its users.

This license, the Lesser General Public License, applies to some specially designated software packages–typically libraries–of the Free Software Foundation and other authors who decide to use it. You can use it too, but we suggest you first think carefully about whether this license or the ordinary General Public License is the better strategy to use in any particular case, based on the explanations below.

When we speak of free software, we are referring to freedom of use, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish); that you receive source code or can get it if you want it; that you can change the software and use pieces of it in new free programs; and that you are informed that you can do these things.

To protect your rights, we need to make restrictions that forbid distributors to deny you these rights or to ask you to surrender these rights. These restrictions translate to certain responsibilities for you if you distribute copies of the library or if you modify it.

For example, if you distribute copies of the library, whether gratis or for a fee, you must give the recipients all the rights that we gave you. You must make sure that they, too, receive or can get the source code. If you link other code with the library, you must provide complete object files to the recipients, so that they can relink them with the library after making changes to the library and recompiling it. And you must show them these terms so they know their rights.

We protect your rights with a two-step method: (1) we copyright the library, and (2) we offer you this license, which gives you legal permission to copy, distribute and/or modify the library.

To protect each distributor, we want to make it very clear that there is no warranty for the free library. Also, if the library is modified by someone else and passed on, the recipients should know that what they have is not the original version, so that the original author's reputation will not be affected by problems that might be introduced by others.

Finally, software patents pose a constant threat to the existence of any free program. We wish to make sure that a company cannot effectively restrict the users of a free program by obtaining a

restrictive license from a patent holder. Therefore, we insist that any patent license obtained for a version of the library must be consistent with the full freedom of use specified in this license.

Most GNU software, including some libraries, is covered by the ordinary GNU General Public License. This license, the GNU Lesser General Public License, applies to certain designated libraries, and is quite different from the ordinary General Public License. We use this license for certain libraries in order to permit linking those libraries into non-free programs.

When a program is linked with a library, whether statically or using a shared library, the combination of the two is legally speaking a combined work, a derivative of the original library. The ordinary General Public License therefore permits such linking only if the entire combination fits its criteria of freedom. The Lesser General Public License permits more lax criteria for linking other code with the library.

We call this license the "Lesser" General Public License because it does Less to protect the user's freedom than the ordinary General Public License. It also provides other free software developers Less of an advantage over competing non-free programs. These disadvantages are the reason we use the ordinary General Public License for many libraries. However, the Lesser license provides advantages in certain special circumstances.

For example, on rare occasions, there may be a special need to encourage the widest possible use of a certain library, so that it becomes a de-facto standard. To achieve this, non-free programs must be allowed to use the library. A more frequent case is that a free library does the same job as widely used non-free libraries. In this case, there is little to gain by limiting the free library to free software only, so we use the Lesser General Public License.

In other cases, permission to use a particular library in non-free programs enables a greater number of people to use a large body of free software. For example, permission to use the GNU C Library in non-free programs enables many more people to use the whole GNU operating system, as well as its variant, the GNU/Linux operating system.

Although the Lesser General Public License is Less protective of the users' freedom, it does ensure that the user of a program that is linked with the Library has the freedom and the wherewithal to run that program using a modified version of the Library.

The precise terms and conditions for copying, distribution and modification follow. Pay close attention to the difference between a "work based on the library" and a "work that uses the library". The former contains code derived from the library, whereas the latter must be combined with the library in order to run.

GNU LESSER GENERAL PUBLIC LICENSE TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License Agreement applies to any software library or other program which contains a notice placed by the copyright holder or other authorized party saying it may be distributed under the terms of this Lesser General Public License (also called "this License"). Each licensee is addressed as "you".

A "library" means a collection of software functions and/or data prepared so as to be conveniently linked with application programs (which use some of those functions and data) to form executables.

The "Library", below, refers to any such software library or work which has been distributed under these terms. A "work based on the Library" means either the Library or any derivative work under copyright law: that is to say, a work containing the Library or a portion of it, either verbatim or with modifications and/or translated straightforwardly into another language. (Hereinafter, translation is included without limitation in the term "modification".)

"Source code" for a work means the preferred form of the work for making modifications to it. For a library, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the library.

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running a program using the Library is not restricted, and output from such a program is covered only if its contents constitute a work based on the Library

(independent of the use of the Library in a tool for writing it). Whether that is true depends on what the Library does and what the program that uses the Library does.

1. You may copy and distribute verbatim copies of the Library's complete source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and distribute a copy of this License along with the Library.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Library or any portion of it, thus forming a work based on the Library, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

a) The modified work must itself be a software library.

b) You must cause the files modified to carry prominent notices stating that you changed the files and the date of any change.

c) You must cause the whole of the work to be licensed at no charge to all third parties under the terms of this License.

d) If a facility in the modified Library refers to a function or a table of data to be supplied by an application program that uses the facility, other than as an argument passed when the facility is invoked, then you must make a good faith effort to ensure that, in the event an application does not supply such function or table, the facility still operates, and performs whatever part of its purpose remains meaningful.

(For example, a function in a library to compute square roots has a purpose that is entirely well-defined independent of the application. Therefore, Subsection 2d requires that any application-supplied function or table used by this function must be optional: if the application does not supply it, the square root function must still compute square roots.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Library, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Library, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Library.

In addition, mere aggregation of another work not based on the Library with the Library (or with a work based on the Library) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may opt to apply the terms of the ordinary GNU General Public License instead of this License to a given copy of the Library. To do this, you must alter all the notices that refer to this License, so that they refer to the ordinary GNU General Public License, version 2, instead of to this License. (If a newer version than version 2 of the ordinary GNU General Public License has appeared, then you can specify that version instead if you wish.) Do not make any other change in these notices.

Once this change is made in a given copy, it is irreversible for that copy, so the ordinary GNU General Public License applies to all subsequent copies and derivative works made from that copy.

This option is useful when you wish to copy part of the code of the Library into a program that is not a library.

4. You may copy and distribute the Library (or a portion or derivative of it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you accompany it with the complete corresponding machine-readable source code, which must

be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange.

If distribution of object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place satisfies the requirement to distribute the source code, even though third parties are not compelled to copy the source along with the object code.

5. A program that contains no derivative of any portion of the Library, but is designed to work with the Library by being compiled or linked with it, is called a "work that uses the Library". Such a work, in isolation, is not a derivative work of the Library, and therefore falls outside the scope of this License.

However, linking a "work that uses the Library" with the Library creates an executable that is a derivative of the Library (because it contains portions of the Library), rather than a "work that uses the library". The executable is therefore covered by this License. Section 6 states terms for distribution of such executables.

When a "work that uses the Library" uses material from a header file that is part of the Library, the object code for the work may be a derivative work of the Library even though the source code is not. Whether this is true is especially significant if the work can be linked without the Library, or if the work is itself a library. The threshold for this to be true is not precisely defined by law.

If such an object file uses only numerical parameters, data structure layouts and accessors, and small macros and small inline functions (ten lines or less in length), then the use of the object file is unrestricted, regardless of whether it is legally a derivative work. (Executables containing this object code plus portions of the Library will still fall under Section 6.) Otherwise, if the work is a derivative of the Library, you may distribute the object code for the work under the terms of Section 6. Any executables containing that work also fall under Section 6, whether or not they are linked directly with the Library itself.

6. As an exception to the Sections above, you may also combine or link a "work that uses the Library" with the Library to produce a work containing portions of the Library, and distribute that work under terms of your choice, provided that the terms permit modification of the work for the customer's own use and reverse engineering for debugging such modifications.

You must give prominent notice with each copy of the work that the Library is used in it and that the Library and its use are covered by this License. You must supply a copy of this License. If the work during execution displays copyright notices, you must include the copyright notice for the Library among them, as well as a reference directing the user to the copy of this License. Also, you must do one of these things:

a) Accompany the work with the complete corresponding machine-readable source code for the Library including whatever changes were used in the work (which must be distributed under Sections 1 and 2 above); and, if the work is an executable linked with the Library, with the complete machine-readable "work that uses the Library", as object code and/or source code, so that the user can modify the Library and then relink to produce a modified executable containing the modified Library. (It is understood that the user who changes the contents of definitions files in the Library will not necessarily be able to recompile the application to use the modified definitions.)

b) Use a suitable shared library mechanism for linking with the Library. A suitable mechanism is one that (1) uses at run time a copy of the library already present on the user's computer system, rather than copying library functions into the executable, and (2) will operate properly with a modified version of the library, if the user installs one, as long as the modified version is interface-compatible with the version that the work was made with.

c) Accompany the work with a written offer, valid for at least three years, to give the same user the materials specified in Subsection 6a, above, for a charge no more than the cost of performing this distribution.

d) If distribution of the work is made by offering access to copy from a designated place, offer equivalent access to copy the above specified materials from the same place.

e) Verify that the user has already received a copy of these materials or that you have already sent this user a copy.

For an executable, the required form of the "work that uses the Library" must include any data and utility programs needed for reproducing the executable from it. However, as a special exception, the materials to be distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

It may happen that this requirement contradicts the license restrictions of other proprietary libraries that do not normally accompany the operating system. Such a contradiction means you cannot use both them and the Library together in an executable that you distribute.

7. You may place library facilities that are a work based on the Library side-by-side in a single library together with other library facilities not covered by this License, and distribute such a combined library, provided that the separate distribution of the work based on the Library and of the other library facilities is otherwise permitted, and provided that you do these two things:

a) Accompany the combined library with a copy of the same work based on the Library, uncombined with any other library facilities. This must be distributed under the terms of the Sections above.

b) Give prominent notice with the combined library of the fact that part of it is a work based on the Library, and explaining where to find the accompanying uncombined form of the same work.

8. You may not copy, modify, sublicense, link with, or distribute the Library except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense, link with, or distribute the Library is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

9. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Library or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Library (or any work based on the Library), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Library or works based on it.

10. Each time you redistribute the Library (or any work based on the Library), the recipient automatically receives a license from the original licensor to copy, distribute, link with or modify the Library subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties with this License.

11. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Library at all. For example, if a patent license would not permit royalty-free redistribution of the Library by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Library.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply, and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system which is implemented by public license practices. Many people have made generous contributions to the wide range of software dis-

tributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

12. If the distribution and/or use of the Library is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Library under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

13. The Free Software Foundation may publish revised and/or new versions of the Lesser General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Library specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Library does not specify a license version number, you may choose any version ever published by the Free Software Foundation.

14. If you wish to incorporate parts of the Library into other free programs whose distribution conditions are incompatible with these, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

15. BECAUSE THE LIBRARY IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE LIBRARY, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE LIBRARY "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE LIBRARY IS WITH YOU. SHOULD THE LIBRARY PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

16. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE LIBRARY AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE LIBRARY (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE LIBRARY TO OPERATE WITH ANY OTHER SOFTWARE), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

### 16.2.3 Mozilla Public License (MPL)

MOZILLA PUBLIC LICENSE Version 1.1

1. Definitions.

1.0.1. "Commercial Use" means distribution or otherwise making the Covered Code available to a third party.

1.1. "Contributor" means each entity that creates or contributes to the creation of Modifications.

1.2. “Contributor Version” means the combination of the Original Code, prior Modifications used by a Contributor, and the Modifications made by that particular Contributor.

1.3. “Covered Code” means the Original Code or Modifications or the combination of the Original Code and Modifications, in each case including portions thereof.

1.4. “Electronic Distribution Mechanism” means a mechanism generally accepted in the software development community for the electronic transfer of data.

1.5. “Executable” means Covered Code in any form other than Source Code.

1.6. “Initial Developer” means the individual or entity identified as the Initial Developer in the Source Code notice required by Exhibit A.

1.7. “Larger Work” means a work which combines Covered Code or portions thereof with code not governed by the terms of this License.

1.8. “License” means this document.

1.8.1. “Licensable” means having the right to grant, to the maximum extent possible, whether at the time of the initial grant or subsequently acquired, any and all of the rights conveyed herein.

1.9. “Modifications” means any addition to or deletion from the substance or structure of either the Original Code or any previous Modifications. When Covered Code is released as a series of files, a Modification is:

A. Any addition to or deletion from the contents of a file containing Original Code or previous Modifications.

B. Any new file that contains any part of the Original Code or previous Modifications.

1.10. “Original Code” means Source Code of computer software code which is described in the Source Code notice required by Exhibit A as Original Code, and which, at the time of its release under this License is not already Covered Code governed by this License.

1.10.1. “Patent Claims” means any patent claim(s), now owned or hereafter acquired, including without limitation, method, process, and apparatus claims, in any patent Licensable by grantor.

1.11. “Source Code” means the preferred form of the Covered Code for making modifications to it, including all modules it contains, plus any associated interface definition files, scripts used to control compilation and installation of an Executable, or source code differential comparisons against either the Original Code or another well known, available Covered Code of the Contributor’s choice. The Source Code can be in a compressed or archival form, provided the appropriate decompression or de-archiving software is widely available for no charge.

1.12. “You” (or “Your”) means an individual or a legal entity exercising rights under, and complying with all of the terms of, this License or a future version of this License issued under Section 6.1. For legal entities, “You” includes any entity which controls, is controlled by, or is under common control with You. For purposes of this definition, “control” means (a) the power, direct or indirect, to cause the direction or management of such entity, whether by contract or otherwise, or (b) ownership of more than fifty percent (50%) of the outstanding shares or beneficial ownership of such entity.

2. Source Code License.

2.1. The Initial Developer Grant. The Initial Developer hereby grants You a world-wide, royalty-free, non-exclusive license, subject to third party intellectual property claims:

(a) under intellectual property rights (other than patent or trademark) Licensable by Initial Developer to use, reproduce, modify, display, perform, sublicense and distribute the Original Code (or portions thereof) with or without Modifications, and/or as part of a Larger Work; and

(b) under Patents Claims infringed by the making, using or selling of Original Code, to make, have made, use, practice, sell, and offer for sale, and/or otherwise dispose of the Original Code (or portions thereof).

(c) the licenses granted in this Section 2.1(a) and (b) are effective on the date Initial Developer first distributes Original Code under the terms of this License.

(d) Notwithstanding Section 2.1(b) above, no patent license is granted: 1) for code that You delete from the Original Code; 2) separate from the Original Code; or 3) for infringements

caused by: i) the modification of the Original Code or ii) the combination of the Original Code with other software or devices.

2.2. Contributor Grant. Subject to third party intellectual property claims, each Contributor hereby grants You a world-wide, royalty-free, non-exclusive license

(a) under intellectual property rights (other than patent or trademark) Licensable by Contributor, to use, reproduce, modify, display, perform, sublicense and distribute the Modifications created by such Contributor (or portions thereof) either on an unmodified basis, with other Modifications, as Covered Code and/or as part of a Larger Work; and

(b) under Patent Claims infringed by the making, using, or selling of Modifications made by that Contributor either alone and/or in combination with its Contributor Version (or portions of such combination), to make, use, sell, offer for sale, have made, and/or otherwise dispose of: 1) Modifications made by that Contributor (or portions thereof); and 2) the combination of Modifications made by that Contributor with its Contributor Version (or portions of such combination).

(c) the licenses granted in Sections 2.2(a) and 2.2(b) are effective on the date Contributor first makes Commercial Use of the Covered Code.

(d) Notwithstanding Section 2.2(b) above, no patent license is granted: 1) for any code that Contributor has deleted from the Contributor Version; 2) separate from the Contributor Version; 3) for infringements caused by: i) third party modifications of Contributor Version or ii) the combination of Modifications made by that Contributor with other software (except as part of the Contributor Version) or other devices; or 4) under Patent Claims infringed by Covered Code in the absence of Modifications made by that Contributor.

3. Distribution Obligations.

3.1. Application of License. The Modifications which You create or to which You contribute are governed by the terms of this License, including without limitation Section 2.2. The Source Code version of Covered Code may be distributed only under the terms of this License or a future version of this License released under Section 6.1, and You must include a copy of this License with every copy of the Source Code You distribute. You may not offer or impose any terms on any Source Code version that alters or restricts the applicable version of this License or the recipients' rights hereunder. However, You may include an additional document offering the additional rights described in Section 3.5.

3.2. Availability of Source Code. Any Modification which You create or to which You contribute must be made available in Source Code form under the terms of this License either on the same media as an Executable version or via an accepted Electronic Distribution Mechanism to anyone to whom you made an Executable version available; and if made available via Electronic Distribution Mechanism, must remain available for at least twelve (12) months after the date it initially became available, or at least six (6) months after a subsequent version of that particular Modification has been made available to such recipients. You are responsible for ensuring that the Source Code version remains available even if the Electronic Distribution Mechanism is maintained by a third party.

3.3. Description of Modifications. You must cause all Covered Code to which You contribute to contain a file documenting the changes You made to create that Covered Code and the date of any change. You must include a prominent statement that the Modification is derived, directly or indirectly, from Original Code provided by the Initial Developer and including the name of the Initial Developer in (a) the Source Code, and (b) in any notice in an Executable version or related documentation in which You describe the origin or ownership of the Covered Code.

3.4. Intellectual Property Matters

(a) Third Party Claims. If Contributor has knowledge that a license under a third party's intellectual property rights is required to exercise the rights granted by such Contributor under Sections 2.1 or 2.2, Contributor must include a text file with the Source Code distribution titled "LEGAL" which describes the claim and the party making the claim in sufficient detail that a recipient will know whom to contact. If Contributor obtains such knowledge after the Modification is made available as described in Section 3.2, Contributor shall promptly modify the LEGAL file

in all copies Contributor makes available thereafter and shall take other steps (such as notifying appropriate mailing lists or newsgroups) reasonably calculated to inform those who received the Covered Code that new knowledge has been obtained.

(b) Contributor APIs. If Contributor's Modifications include an application programming interface and Contributor has knowledge of patent licenses which are reasonably necessary to implement that API, Contributor must also include this information in the LEGAL file.

3.5. Required Notices. You must duplicate the notice in Exhibit A in each file of the Source Code. If it is not possible to put such notice in a particular Source Code file due to its structure, then You must include such notice in a location (such as a relevant directory) where a user would be likely to look for such a notice. If You created one or more Modification(s) You may add your name as a Contributor to the notice described in Exhibit A. You must also duplicate this License in any documentation for the Source Code where You describe recipients' rights or ownership rights relating to Covered Code. You may choose to offer, and to charge a fee for, warranty, support, indemnity or liability obligations to one or more recipients of Covered Code. However, You may do so only on Your own behalf, and not on behalf of the Initial Developer or any Contributor. You must make it absolutely clear than any such warranty, support, indemnity or liability obligation is offered by You alone, and You hereby agree to indemnify the Initial Developer and every Contributor for any liability incurred by the Initial Developer or such Contributor as a result of warranty, support, indemnity or liability terms You offer.

3.6. Distribution of Executable Versions. You may distribute Covered Code in Executable form only if the requirements of Section 3.1-3.5 have been met for that Covered Code, and if You include a notice stating that the Source Code version of the Covered Code is available under the terms of this License, including a description of how and where You have fulfilled the obligations of Section 3.2. The notice must be conspicuously included in any notice in an Executable version, related documentation or collateral in which You describe recipients' rights relating to the Covered Code. You may distribute the Executable version of Covered Code or ownership rights under a license of Your choice, which may contain terms different from this License, provided that You are in compliance with the terms of this License and that the license for the Executable version does not attempt to limit or alter the recipient's rights in the Source Code version from the rights set forth in this License. If You distribute the Executable version under a different license You must make it absolutely clear that any terms which differ from this License are offered by You alone, not by the Initial Developer or any Contributor. You hereby agree to indemnify the Initial Developer and every Contributor for any liability incurred by the Initial Developer or such Contributor as a result of any such terms You offer.

3.7. Larger Works. You may create a Larger Work by combining Covered Code with other code not governed by the terms of this License and distribute the Larger Work as a single product. In such a case, You must make sure the requirements of this License are fulfilled for the Covered Code.

4. Inability to Comply Due to Statute or Regulation.If it is impossible for You to comply with any of the terms of this License with respect to some or all of the Covered Code due to statute, judicial order, or regulation then You must: (a) comply with the terms of this License to the maximum extent possible; and (b) describe the limitations and the code they affect. Such description must be included in the LEGAL file described in Section 3.4 and must be included with all distributions of the Source Code. Except to the extent prohibited by statute or regulation, such description must be sufficiently detailed for a recipient of ordinary skill to be able to understand it.

5. Application of this License. This License applies to code to which the Initial Developer has attached the notice in Exhibit A and to related Covered Code.

6. Versions of the License.

6.1. New Versions. Netscape Communications Corporation ("Netscape") may publish revised and/or new versions of the License from time to time. Each version will be given a distinguishing version number.

6.2. Effect of New Versions. Once Covered Code has been published under a particular version of the License, You may always continue to use it under the terms of that version. You may also choose to use such Covered Code under the terms of any subsequent version of the License published by Netscape. No one other than Netscape has the right to modify the terms applicable to Covered Code created under this License.

6.3. Derivative Works. If You create or use a modified version of this License (which you may only do in order to apply it to code which is not already Covered Code governed by this License), You must (a) rename Your license so that the phrases "Mozilla", "MOZILLAPL", "MOZPL", "Netscape", "MPL", "NPL" or any confusingly similar phrase do not appear in your license (except to note that your license differs from this License) and (b) otherwise make it clear that Your version of the license contains terms which differ from the Mozilla Public License and Netscape Public License. (Filling in the name of the Initial Developer, Original Code or Contributor in the notice described in Exhibit A shall not of themselves be deemed to be modifications of this License.)

7. DISCLAIMER OF WARRANTY.

COVERED CODE IS PROVIDED UNDER THIS LICENSE ON AN "AS IS" BASIS, WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, WITHOUT LIMITATION, WARRANTIES THAT THE COVERED CODE IS FREE OF DEFECTS, MERCHANTABLE, FIT FOR A PARTICULAR PURPOSE OR NON-INFRINGING. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE COVERED CODE IS WITH YOU. SHOULD ANY COVERED CODE PROVE DEFECTIVE IN ANY RESPECT, YOU (NOT THE INITIAL DEVELOPER OR ANY OTHER CONTRIBUTOR) ASSUME THE COST OF ANY NECESSARY SERVICING, REPAIR OR CORRECTION. THIS DISCLAIMER OF WARRANTY CONSTITUTES AN ESSENTIAL PART OF THIS LICENSE. NO USE OF ANY COVERED CODE IS AUTHORIZED HEREUNDER EXCEPT UNDER THIS DISCLAIMER.

8. TERMINATION.

8.1. This License and the rights granted hereunder will terminate automatically if You fail to comply with terms herein and fail to cure such breach within 30 days of becoming aware of the breach. All sublicenses to the Covered Code which are properly granted shall survive any termination of this License. Provisions which, by their nature, must remain in effect beyond the termination of this License shall survive.

8.2. If You initiate litigation by asserting a patent infringement claim (excluding declaratory judgment actions) against Initial Developer or a Contributor (the Initial Developer or Contributor against whom You file such action is referred to as "Participant") alleging that:

(a) such Participant's Contributor Version directly or indirectly infringes any patent, then any and all rights granted by such Participant to You under Sections 2.1 and/or 2.2 of this License shall, upon 60 days notice from Participant terminate prospectively, unless if within 60 days after receipt of notice You either: (i) agree in writing to pay Participant a mutually agreeable reasonable royalty for Your past and future use of Modifications made by such Participant, or (ii) withdraw Your litigation claim with respect to the Contributor Version against such Participant. If within 60 days of notice, a reasonable royalty and payment arrangement are not mutually agreed upon in writing by the parties or the litigation claim is not withdrawn, the rights granted by Participant to You under Sections 2.1 and/or 2.2 automatically terminate at the expiration of the 60 day notice period specified above.

(b) any software, hardware, or device, other than such Participant's Contributor Version, directly or indirectly infringes any patent, then any rights granted to You by such Participant under Sections 2.1(b) and 2.2(b) are revoked effective as of the date You first made, used, sold, distributed, or had made, Modifications made by that Participant.

8.3. If You assert a patent infringement claim against Participant alleging that such Participant's Contributor Version directly or indirectly infringes any patent where such claim is resolved (such as by license or settlement) prior to the initiation of patent infringement litigation, then the reasonable value of the licenses granted by such Participant under Sections 2.1 or 2.2 shall be taken into account in determining the amount or value of any payment or license.

8.4. In the event of termination under Sections 8.1 or 8.2 above, all end user license agreements (excluding distributors and resellers) which have been validly granted by You or any distributor hereunder prior to termination shall survive termination.

9. LIMITATION OF LIABILITY. UNDER NO CIRCUMSTANCES AND UNDER NO LEGAL THEORY, WHETHER TORT (INCLUDING NEGLIGENCE), CONTRACT, OR OTHERWISE, SHALL YOU, THE INITIAL DEVELOPER, ANY OTHER CONTRIBUTOR, OR ANY DISTRIBUTOR OF COVERED CODE, OR ANY SUPPLIER OF ANY OF SUCH PARTIES, BE LIABLE TO ANY PERSON FOR ANY INDIRECT, SPECIAL, INCIDENTAL, OR CONSEQUENTIAL DAMAGES OF ANY CHARACTER INCLUDING, WITHOUT LIMITATION, DAMAGES FOR LOSS OF GOODWILL, WORK STOPPAGE, COMPUTER FAILURE OR MALFUNCTION, OR ANY AND ALL OTHER COMMERCIAL DAMAGES OR LOSSES, EVEN IF SUCH PARTY SHALL HAVE BEEN INFORMED OF THE POSSIBILITY OF SUCH DAMAGES. THIS LIMITATION OF LIABILITY SHALL NOT APPLY TO LIABILITY FOR DEATH OR PERSONAL INJURY RESULTING FROM SUCH PARTY'S NEGLIGENCE TO THE EXTENT APPLICABLE LAW PROHIBITS SUCH LIMITATION. SOME JURISDICTIONS DO NOT ALLOW THE EXCLUSION OR LIMITATION OF INCIDENTAL OR CONSEQUENTIAL DAMAGES, SO THIS EXCLUSION AND LIMITATION MAY NOT APPLY TO YOU.

10. U.S. GOVERNMENT END USERS. The Covered Code is a "commercial item," as that term is defined in 48 C.F.R. 2.101 (Oct. 1995), consisting of "commercial computer software" and "commercial computer software documentation," as such terms are used in 48 C.F.R. 12.212 (Sept. 1995). Consistent with 48 C.F.R. 12.212 and 48 C.F.R. 227.7202-1 through 227.7202-4 (June 1995), all U.S. Government End Users acquire Covered Code with only those rights set forth herein.

11. MISCELLANEOUS. This License represents the complete agreement concerning subject matter hereof. If any provision of this License is held to be unenforceable, such provision shall be reformed only to the extent necessary to make it enforceable. This License shall be governed by California law provisions (except to the extent applicable law, if any, provides otherwise), excluding its conflict-of-law provisions. With respect to disputes in which at least one party is a citizen of, or an entity chartered or registered to do business in the United States of America, any litigation relating to this License shall be subject to the jurisdiction of the Federal Courts of the Northern District of California, with venue lying in Santa Clara County, California, with the losing party responsible for costs, including without limitation, court costs and reasonable attorneys' fees and expenses. The application of the United Nations Convention on Contracts for the International Sale of Goods is expressly excluded. Any law or regulation which provides that the language of a contract shall be construed against the drafter shall not apply to this License.

12. RESPONSIBILITY FOR CLAIMS. As between Initial Developer and the Contributors, each party is responsible for claims and damages arising, directly or indirectly, out of its utilization of rights under this License and You agree to work with Initial Developer and Contributors to distribute such responsibility on an equitable basis. Nothing herein is intended or shall be deemed to constitute any admission of liability.

13. MULTIPLE-LICENSED CODE. Initial Developer may designate portions of the Covered Code as "Multiple-Licensed". "Multiple-Licensed" means that the Initial Developer permits you to utilize portions of the Covered Code under Your choice of the NPL or the alternative licenses, if any, specified by the Initial Developer in the file described in Exhibit A.

EXHIBIT A -Mozilla Public License.

"The contents of this file are subject to the Mozilla Public License Version 1.1 (the "License"); you may not use this file except in compliance with the License. You may obtain a copy of the License at http://www.mozilla.org/MPL/

Software distributed under the License is distributed on an "AS IS" basis, WITHOUT WARRANTY OF ANY KIND, either express or implied. See the License for the specific language governing rights and limitations under the License.

The Original Code is _____.

The Initial Developer of the Original Code is _____. Portions created by _____ are Copyright (C) _____ _____. All Rights Reserved.

Contributor(s): _____.

Alternatively, the contents of this file may be used under the terms of the _____ license (the "[___] License"), in which case the provisions of [_____] License are applicable instead of those above. If you wish to allow use of your version of this file only under the terms of the [____] License and not to allow others to use your version of this file under the MPL, indicate your decision by deleting the provisions above and replace them with the notice and other provisions required by the [___] License. If you do not delete the provisions above, a recipient may use your version of this file under either the MPL or the [___] License."

[NOTE: The text of this Exhibit A may differ slightly from the text of the notices in the Source Code files of the Original Code. You should use the text of this Exhibit A rather than the text found in the Original Code Source Code for Your Modifications.]

### 16.2.4 MIT License

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

### 16.2.5 X Consortium License (X11)

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

### 16.2.6 zlib License

This software is provided 'as-is', without any express or implied warranty. In no event will the authors be held liable for any damages arising from the use of this software.

Permission is granted to anyone to use this software for any purpose, including commercial applications, and to alter it and redistribute it freely, subject to the following restrictions:

1. The origin of this software must not be misrepresented; you must not claim that you wrote the original software. If you use this software in a product, an acknowledgment in the product documentation would be appreciated but is not required.

2. Altered source versions must be plainly marked as such, and must not be misrepresented as being the original software.

3. This notice may not be removed or altered from any source distribution.

```
Jean-loup Gailly        Mark Adler
jloup@gzip.org          madler@alumni.caltech.edu
```

### 16.2.7 OpenSSL License

This package is an SSL implementation written by Eric Young (eay@cryptsoft.com). The implementation was written so as to conform with Netscape's SSL.

This library is free for commercial and non-commercial use as long as the following conditions are adhered to. The following conditions apply to all code found in this distribution, be it the RC4, RSA, lhash, DES, etc., code; not just the SSL code. The SSL documentation included with this distribution is covered by the same copyright terms except that the holder is Tim Hudson (tjh@cryptsoft.com).

Copyright remains Eric Young's, and as such any Copyright notices in the code are not to be removed. If this package is used in a product, Eric Young should be given attribution as the author of the parts of the library used. This can be in the form of a textual message at program startup or in documentation (online or textual) provided with the package.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the copyright notice, this list of conditions and the following disclaimer.

2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

3. All advertising materials mentioning features or use of this software must display the following acknowledgement: "This product includes cryptographic software written by Eric Young (eay@cryptsoft.com)" The word 'cryptographic' can be left out if the routines from the library being used are not cryptographic related :-).

4. If you include any Windows specific code (or a derivative thereof) from the apps directory (application code) you must include an acknowledgement: "This product includes software written by Tim Hudson (tjh@cryptsoft.com)"

THIS SOFTWARE IS PROVIDED BY ERIC YOUNG "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE AUTHOR OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

The licence and distribution terms for any publicly available version or derivative of this code cannot be changed. i.e. this code cannot simply be copied and put under another distribution licence [including the GNU Public Licence.]

### 16.2.8 Slirp License

Copyright (c) 1995,1996 Danny Gasparovski. All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

3. All advertising materials mentioning features or use of this software must display the following acknowledgment: This product includes software developed by Danny Gasparovski.

THIS SOFTWARE IS PROVIDED "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL DANNY GASPAROVSKI OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

### 16.2.9 liblzf License

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

3. The name of the author may not be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE AUTHOR "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE AUTHOR BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

### 16.2.10 libpng License

The PNG Reference Library is supplied "AS IS". The Contributing Authors and Group 42, Inc. disclaim all warranties, expressed or implied, including, without limitation, the warranties of merchantability and of fitness for any purpose. The Contributing Authors and Group 42, Inc. assume no liability for direct, indirect, incidental, special, exemplary, or consequential damages, which may result from the use of the PNG Reference Library, even if advised of the possibility of such damage.

Permission is hereby granted to use, copy, modify, and distribute this source code, or portions hereof, for any purpose, without fee, subject to the following restrictions:

1. The origin of this source code must not be misrepresented.

2. Altered versions must be plainly marked as such and must not be misrepresented as being the original source.

3. This Copyright notice may not be removed or altered from any source or altered source distribution.

The Contributing Authors and Group 42, Inc. specifically permit, without fee, and encourage the use of this source code as a component to supporting the PNG file format in commercial products. If you use this source code in a product, acknowledgment is not required but would be appreciated.

### 16.2.11 lwIP License

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

3. The name of the author may not be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE AUTHOR “AS IS” AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE AUTHOR BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

### 16.2.12 libxml License

Except where otherwise noted in the source code (e.g. the files hash.c, list.c and the trio files, which are covered by a similar licence but with different Copyright notices) all the files are:

Copyright (C) 1998-2003 Daniel Veillard. All Rights Reserved.

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the “Software”), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED “AS IS”, WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE DANIEL VEILLARD BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

Except as contained in this notice, the name of Daniel Veillard shall not be used in advertising or otherwise to promote the sale, use or other dealings in this Software without prior written authorization from him.

### 16.2.13 libxslt Licenses

Licence for libxslt except libexslt:

Copyright (C) 2001-2002 Daniel Veillard. All Rights Reserved.

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the “Software”), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED “AS IS”, WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE DANIEL VEILLARD BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

Except as contained in this notice, the name of Daniel Veillard shall not be used in advertising or otherwise to promote the sale, use or other dealings in this Software without prior written authorization from him.

Licence for libexslt:

Copyright (C) 2001-2002 Thomas Broyer, Charlie Bozeman and Daniel Veillard. All Rights Reserved.

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the “Software”), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED “AS IS”, WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

Except as contained in this notice, the name of the authors shall not be used in advertising or otherwise to promote the sale, use or other dealings in this Software without prior written authorization from him.

### 16.2.14 gSOAP Public License Version 1.3a

The gSOAP public license is derived from the Mozilla Public License (MPL1.1). The sections that were deleted from the original MPL1.1 text are 1.0.1, 2.1.(c),(d), 2.2.(c),(d), 8.2.(b), 10, and 11. Section 3.8 was added. The modified sections are 2.1.(b), 2.2.(b), 3.2 (simplified), 3.5 (deleted the last sentence), and 3.6 (simplified).

1 DEFINITIONS

1.1. “Contributor” means each entity that creates or contributes to the creation of Modifications.

1.2. “Contributor Version” means the combination of the Original Code, prior Modifications used by a Contributor, and the Modifications made by that particular Contributor.

1.3. “Covered Code” means the Original Code, or Modifications or the combination of the Original Code, and Modifications, in each case including portions thereof.

1.4. “Electronic Distribution Mechanism” means a mechanism generally accepted in the software development community for the electronic transfer of data.

1.5. “Executable” means Covered Code in any form other than Source Code.

1.6. “Initial Developer” means the individual or entity identified as the Initial Developer in the Source Code notice required by Exhibit A.

1.7. “Larger Work” means a work which combines Covered Code or portions thereof with code not governed by the terms of this License.

1.8. “License” means this document.

1.8.1. “Licensable” means having the right to grant, to the maximum extent possible, whether at the time of the initial grant or subsequently acquired, any and all of the rights conveyed herein.

1.9. “Modifications” means any addition to or deletion from the substance or structure of either the Original Code or any previous Modifications. When Covered Code is released as a series of files, a Modification is:

A. Any addition to or deletion from the contents of a file containing Original Code or previous Modifications.

B. Any new file that contains any part of the Original Code, or previous Modifications.

1.10. “Original Code” means Source Code of computer software code which is described in the Source Code notice required by Exhibit A as Original Code, and which, at the time of its release under this License is not already Covered Code governed by this License.

1.10.1. “Patent Claims” means any patent claim(s), now owned or hereafter acquired, including without limitation, method, process, and apparatus claims, in any patent Licensable by grantor.

1.11. “Source Code” means the preferred form of the Covered Code for making modifications to it, including all modules it contains, plus any associated interface definition files, scripts used to control compilation and installation of an Executable, or source code differential comparisons against either the Original Code or another well known, available Covered Code of the Contributor’s choice. The Source Code can be in a compressed or archival form, provided the appropriate decompression or de-archiving software is widely available for no charge.

1.12. “You” (or “Your”) means an individual or a legal entity exercising rights under, and complying with all of the terms of, this License or a future version of this License issued under Section 6.1. For legal entities, “You” includes any entity which controls, is controlled by, or is under common control with You. For purposes of this definition, “control” means (a) the power, direct or indirect, to cause the direction or management of such entity, whether by contract or otherwise, or (b) ownership of more than fifty percent (50%) of the outstanding shares or beneficial ownership of such entity.

2 SOURCE CODE LICENSE.

2.1. The Initial Developer Grant.

The Initial Developer hereby grants You a world-wide, royalty-free, non-exclusive license, subject to third party intellectual property claims:

(a) under intellectual property rights (other than patent or trademark) Licensable by Initial Developer to use, reproduce, modify, display, perform, sublicense and distribute the Original Code (or portions thereof) with or without Modifications, and/or as part of a Larger Work; and

(b) under patents now or hereafter owned or controlled by Initial Developer, to make, have made, use and sell (“offer to sell and import”) the Original Code, Modifications, or portions thereof, but solely to the extent that any such patent is reasonably necessary to enable You to utilize, alone or in combination with other software, the Original Code, Modifications, or any combination or portions thereof.

(c)

(d)

2.2. Contributor Grant.

Subject to third party intellectual property claims, each Contributor hereby grants You a worldwide, royalty-free, non-exclusive license

(a) under intellectual property rights (other than patent or trademark) Licensable by Contributor, to use, reproduce, modify, display, perform, sublicense and distribute the Modifications created by such Contributor (or portions thereof) either on an unmodified basis, with other Modifications, as Covered Code and/or as part of a Larger Work; and

(b) under patents now or hereafter owned or controlled by Contributor, to make, have made, use and sell ("offer to sell and import") the Contributor Version (or portions thereof), but solely to the extent that any such patent is reasonably necessary to enable You to utilize, alone or in combination with other software, the Contributor Version (or portions thereof).

(c)

(d)

3 DISTRIBUTION OBLIGATIONS.

3.1. Application of License.

The Modifications which You create or to which You contribute are governed by the terms of this License, including without limitation Section 2.2. The Source Code version of Covered Code may be distributed only under the terms of this License or a future version of this License released under Section 6.1, and You must include a copy of this License with every copy of the Source Code You distribute. You may not offer or impose any terms on any Source Code version that alters or restricts the applicable version of this License or the recipients' rights hereunder. However, You may include an additional document offering the additional rights described in Section 3.5.

3.2. Availability of Source Code.

Any Modification created by You will be provided to the Initial Developer in Source Code form and are subject to the terms of the License. 3.3. Description of Modifications.

You must cause all Covered Code to which You contribute to contain a file documenting the changes You made to create that Covered Code and the date of any change. You must include a prominent statement that the Modification is derived, directly or indirectly, from Original Code provided by the Initial Developer and including the name of the Initial Developer in (a) the Source Code, and (b) in any notice in an Executable version or related documentation in which You describe the origin or ownership of the Covered Code.

3.4. Intellectual Property Matters.

(a) Third Party Claims. If Contributor has knowledge that a license under a third party's intellectual property rights is required to exercise the rights granted by such Contributor under Sections 2.1 or 2.2, Contributor must include a text file with the Source Code distribution titled "LEGAL" which describes the claim and the party making the claim in sufficient detail that a recipient will know whom to contact. If Contributor obtains such knowledge after the Modification is made available as described in Section 3.2, Contributor shall promptly modify the LEGAL file in all copies Contributor makes available thereafter and shall take other steps (such as notifying appropriate mailing lists or newsgroups) reasonably calculated to inform those who received the Covered Code that new knowledge has been obtained.

(b) Contributor APIs. If Contributor's Modifications include an application programming interface and Contributor has knowledge of patent licenses which are reasonably necessary to implement that API, Contributor must also include this information in the LEGAL file.

(c) Representations. Contributor represents that, except as disclosed pursuant to Section 3.4(a) above, Contributor believes that Contributor's Modifications are Contributor's original creation(s) and/or Contributor has sufficient rights to grant the rights conveyed by this License.

3.5. Required Notices. You must duplicate the notice in Exhibit A in each file of the Source Code. If it is not possible to put such notice in a particular Source Code file due to its structure, then You must include such notice in a location (such as a relevant directory) where a user would be likely to look for such a notice. If You created one or more Modification(s) You may add your name as a Contributor to the notice described in Exhibit A. You must also duplicate this License in

any documentation for the Source Code where You describe recipients' rights or ownership rights relating to Covered Code. You may choose to offer, and to charge a fee for, warranty, support, indemnity or liability obligations to one or more recipients of Covered Code. However, You may do so only on Your own behalf, and not on behalf of the Initial Developer or any Contributor.

3.6. Distribution of Executable Versions. You may distribute Covered Code in Executable form only if the requirements of Section 3.1-3.5 have been met for that Covered Code. You may distribute the Executable version of Covered Code or ownership rights under a license of Your choice, which may contain terms different from this License, provided that You are in compliance with the terms of this License and that the license for the Executable version does not attempt to limit or alter the recipient's rights in the Source Code version from the rights set forth in this License. If You distribute the Executable version under a different license You must make it absolutely clear that any terms which differ from this License are offered by You alone, not by the Initial Developer or any Contributor. If you distribute executable versions containing Covered Code as part of a product, you must reproduce the notice in Exhibit B in the documentation and/or other materials provided with the product.

3.7. Larger Works. You may create a Larger Work by combining Covered Code with other code not governed by the terms of this License and distribute the Larger Work as a single product. In such a case, You must make sure the requirements of this License are fulfilled for the Covered Code.

3.8. Restrictions. You may not remove any product identification, copyright, proprietary notices or labels from gSOAP.

4 INABILITY TO COMPLY DUE TO STATUTE OR REGULATION.

If it is impossible for You to comply with any of the terms of this License with respect to some or all of the Covered Code due to statute, judicial order, or regulation then You must: (a) comply with the terms of this License to the maximum extent possible; and (b) describe the limitations and the code they affect. Such description must be included in the LEGAL file described in Section 3.4 and must be included with all distributions of the Source Code. Except to the extent prohibited by statute or regulation, such description must be sufficiently detailed for a recipient of ordinary skill to be able to understand it.

5 APPLICATION OF THIS LICENSE.

This License applies to code to which the Initial Developer has attached the notice in Exhibit A and to related Covered Code.

6 VERSIONS OF THE LICENSE.

6.1. New Versions.

Grantor may publish revised and/or new versions of the License from time to time. Each version will be given a distinguishing version number.

6.2. Effect of New Versions.

Once Covered Code has been published under a particular version of the License, You may always continue to use it under the terms of that version. You may also choose to use such Covered Code under the terms of any subsequent version of the License.

6.3. Derivative Works.

If You create or use a modified version of this License (which you may only do in order to apply it to code which is not already Covered Code governed by this License), You must (a) rename Your license so that the phrase "gSOAP" or any confusingly similar phrase do not appear in your license (except to note that your license differs from this License) and (b) otherwise make it clear that Your version of the license contains terms which differ from the gSOAP Public License. (Filling in the name of the Initial Developer, Original Code or Contributor in the notice described in Exhibit A shall not of themselves be deemed to be modifications of this License.)

7 DISCLAIMER OF WARRANTY.

COVERED CODE IS PROVIDED UNDER THIS LICENSE ON AN "AS IS" BASIS, WITHOUT WARRANTY OF ANY KIND, WHETHER EXPRESS, IMPLIED OR STATUTORY, INCLUDING, WITHOUT LIMITATION, THE IMPLIED WARRANTIES OF MERCHANTABILITY, OF FITNESS FOR A PARTICULAR PURPOSE, NONINFRINGEMENT OF THIRD PARTY INTELLECTUAL PROPERTY RIGHTS,

AND ANY WARRANTY THAT MAY ARISE BY REASON OF TRADE USAGE, CUSTOM, OR COURSE OF DEALING. WITHOUT LIMITING THE FOREGOING, YOU ACKNOWLEDGE THAT THE SOFTWARE IS PROVIDED "AS IS" AND THAT THE AUTHORS DO NOT WARRANT THE SOFTWARE WILL RUN UNINTERRUPTED OR ERROR FREE. LIMITED LIABILITY THE ENTIRE RISK AS TO RESULTS AND PERFORMANCE OF THE SOFTWARE IS ASSUMED BY YOU. UNDER NO CIRCUMSTANCES WILL THE AUTHORS BE LIABLE FOR ANY SPECIAL, INDIRECT, INCIDENTAL, EXEMPLARY OR CONSEQUENTIAL DAMAGES OF ANY KIND OR NATURE WHATSOEVER, WHETHER BASED ON CONTRACT, WARRANTY, TORT (INCLUDING NEGLIGENCE), STRICT LIABILITY OR OTHERWISE, ARISING OUT OF OR IN ANY WAY RELATED TO THE SOFTWARE, EVEN IF THE AUTHORS HAVE BEEN ADVISED ON THE POSSIBILITY OF SUCH DAMAGE OR IF SUCH DAMAGE COULD HAVE BEEN REASONABLY FORESEEN, AND NOTWITHSTANDING ANY FAILURE OF ESSENTIAL PURPOSE OF ANY EXCLUSIVE REMEDY PROVIDED. SUCH LIMITATION ON DAMAGES INCLUDES, BUT IS NOT LIMITED TO, DAMAGES FOR LOSS OF GOODWILL, LOST PROFITS, LOSS OF DATA OR SOFTWARE, WORK STOPPAGE, COMPUTER FAILURE OR MALFUNCTION OR IMPAIRMENT OF OTHER GOODS. IN NO EVENT WILL THE AUTHORS BE LIABLE FOR THE COSTS OF PROCUREMENT OF SUBSTITUTE SOFTWARE OR SERVICES. YOU ACKNOWLEDGE THAT THIS SOFTWARE IS NOT DESIGNED FOR USE IN ON-LINE EQUIPMENT IN HAZARDOUS ENVIRONMENTS SUCH AS OPERATION OF NUCLEAR FACILITIES, AIRCRAFT NAVIGATION OR CONTROL, OR LIFE-CRITICAL APPLICATIONS. THE AUTHORS EXPRESSLY DISCLAIM ANY LIABILITY RESULTING FROM USE OF THE SOFTWARE IN ANY SUCH ON-LINE EQUIPMENT IN HAZARDOUS ENVIRONMENTS AND ACCEPTS NO LIABILITY IN RESPECT OF ANY ACTIONS OR CLAIMS BASED ON THE USE OF THE SOFTWARE IN ANY SUCH ON-LINE EQUIPMENT IN HAZARDOUS ENVIRONMENTS BY YOU. FOR PURPOSES OF THIS PARAGRAPH, THE TERM "LIFE-CRITICAL APPLICATION" MEANS AN APPLICATION IN WHICH THE FUNCTIONING OR MALFUNCTIONING OF THE SOFTWARE MAY RESULT DIRECTLY OR INDIRECTLY IN PHYSICAL INJURY OR LOSS OF HUMAN LIFE. THIS DISCLAIMER OF WARRANTY CONSTITUTES AN ESSENTIAL PART OF THIS LICENSE. NO USE OF ANY COVERED CODE IS AUTHORIZED HEREUNDER EXCEPT UNDER THIS DISCLAIMER.

8 TERMINATION.

8.1.

This License and the rights granted hereunder will terminate automatically if You fail to comply with terms herein and fail to cure such breach within 30 days of becoming aware of the breach. All sublicenses to the Covered Code which are properly granted shall survive any termination of this License. Provisions which, by their nature, must remain in effect beyond the termination of this License shall survive.

8.2.

8.3.

If You assert a patent infringement claim against Participant alleging that such Participant's Contributor Version directly or indirectly infringes any patent where such claim is resolved (such as by license or settlement) prior to the initiation of patent infringement litigation, then the reasonable value of the licenses granted by such Participant under Sections 2.1 or 2.2 shall be taken into account in determining the amount or value of any payment or license.

8.4. In the event of termination under Sections 8.1 or 8.2 above, all end user license agreements (excluding distributors and resellers) which have been validly granted by You or any distributor hereunder prior to termination shall survive termination.

9 LIMITATION OF LIABILITY.

UNDER NO CIRCUMSTANCES AND UNDER NO LEGAL THEORY, WHETHER TORT (INCLUDING NEGLIGENCE), CONTRACT, OR OTHERWISE, SHALL YOU, THE INITIAL DEVELOPER, ANY OTHER CONTRIBUTOR, OR ANY DISTRIBUTOR OF COVERED CODE, OR ANY SUPPLIER OF ANY OF SUCH PARTIES, BE LIABLE TO ANY PERSON FOR ANY INDIRECT, SPECIAL, INCIDENTAL, OR CONSEQUENTIAL DAMAGES OF ANY CHARACTER INCLUDING, WITHOUT LIMITATION, DAMAGES FOR LOSS OF GOODWILL, WORK STOPPAGE, COMPUTER FAILURE OR MALFUNCTION, OR ANY AND ALL OTHER COMMERCIAL DAMAGES OR LOSSES, EVEN IF SUCH

PARTY SHALL HAVE BEEN INFORMED OF THE POSSIBILITY OF SUCH DAMAGES. THIS LIMITATION OF LIABILITY SHALL NOT APPLY TO LIABILITY FOR DEATH OR PERSONAL INJURY RESULTING FROM SUCH PARTY'S NEGLIGENCE TO THE EXTENT APPLICABLE LAW PROHIBITS SUCH LIMITATION. SOME JURISDICTIONS DO NOT ALLOW THE EXCLUSION OR LIMITATION OF INCIDENTAL OR CONSEQUENTIAL DAMAGES, SO THIS EXCLUSION AND LIMITATION MAY NOT APPLY TO YOU.

10 U.S. GOVERNMENT END USERS.

11 MISCELLANEOUS.

12 RESPONSIBILITY FOR CLAIMS.

As between Initial Developer and the Contributors, each party is responsible for claims and damages arising, directly or indirectly, out of its utilization of rights under this License and You agree to work with Initial Developer and Contributors to distribute such responsibility on an equitable basis. Nothing herein is intended or shall be deemed to constitute any admission of liability.

EXHIBIT A.

"The contents of this file are subject to the gSOAP Public License Version 1.3 (the "License"); you may not use this file except in compliance with the License. You may obtain a copy of the License at http://www.cs.fsu.edu/~{}engelen/soaplicense.html. Software distributed under the License is distributed on an "AS IS" basis, WITHOUT WARRANTY OF ANY KIND, either express or implied. See the License for the specific language governing rights and limitations under the License.

The Original Code of the gSOAP Software is: stdsoap.h, stdsoap2.h, stdsoap.c, stdsoap2.c, stdsoap.cpp, stdsoap2.cpp, soapcpp2.h, soapcpp2.c, soapcpp2_lex.l, soapcpp2_yacc.y, error2.h, error2.c, symbol2.c, init2.c, soapdoc2.html, and soapdoc2.pdf, httpget.h, httpget.c, stl.h, stldeque.h, stllist.h, stlvector.h, stlset.h.

The Initial Developer of the Original Code is Robert A. van Engelen. Portions created by Robert A. van Engelen are Copyright (C) 2001-2004 Robert A. van Engelen, Genivia inc. All Rights Reserved.

Contributor(s): "_____." [Note: The text of this Exhibit A may differ slightly form the text of the notices in the Source Code files of the Original code. You should use the text of this Exhibit A rather than the text found in the Original Code Source Code for Your Modifications.]

EXHIBIT B.

"Part of the software embedded in this product is gSOAP software. Portions created by gSOAP are Copyright (C) 2001-2004 Robert A. van Engelen, Genivia inc. All Rights Reserved. THE SOFTWARE IN THIS PRODUCT WAS IN PART PROVIDED BY GENIVIA INC AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE AUTHOR BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE."

## 16.2.15 Chromium Licenses

### 16.2.15.1 Main License

Copyright (c) 2002, Stanford University All rights reserved.

Some portions of Chromium are copyrighted by individiual organizations. Please see the files COPYRIGHT.LLNL and COPYRIGHT.REDHAT for more information.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

- Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
- Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
- Neither the name of Stanford University nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

#### 16.2.15.2 COPYRIGHT.LLNL File

This Chromium distribution contains information and code which is covered under the following notice:

Copyright (c) 2002, The Regents of the University of California. Produced at the Lawrence Livermore National Laboratory For details, contact: Randall Frank (rjfrank@llnl.gov). UCRL-CODE-2002-058 All rights reserved.

This file is part of Chromium. For details, see accompanying documentation.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

Redistributions of source code must retain the above copyright notice, this list of conditions and the disclaimer below.

Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the disclaimer (as noted below) in the documentation and/or other materials provided with the distribution.

Neither the name of the UC/LLNL nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE REGENTS OF THE UNIVERSITY OF CALIFORNIA, THE U.S. DEPARTMENT OF ENERGY OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Additional BSD Notice

1. This notice is required to be provided under our contract with the U.S. Department of Energy (DOE). This work was produced at the University of California, Lawrence Livermore National Laboratory under Contract No. W-7405-ENG-48 with the DOE.

2. Neither the United States Government nor the University of California nor any of their employees, makes any warranty, express or implied, or assumes any liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately-owned rights.

3. Also, reference herein to any specific commercial products, process, or services by trade name, trademark, manufacturer or otherwise does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or the University of California. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or the University of California, and shall not be used for advertising or product endorsement purposes.

#### 16.2.15.3 COPYRIGHT.REDHAT File

This Chromium distribution contains information and code which is covered under the following notice:

Copyright 2001,2002 Red Hat Inc., Durham, North Carolina.

All Rights Reserved.

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the “Software”), to deal in the Software without restriction, including without limitation on the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice (including the next paragraph) shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED “AS IS”, WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NON-INFRINGEMENT. IN NO EVENT SHALL RED HAT AND/OR THEIR SUPPLIERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

### 16.2.16 curl License

COPYRIGHT AND PERMISSION NOTICE

Copyright (c) 1996 - 2009, Daniel Stenberg, daniel@haxx.se.

All rights reserved.

Permission to use, copy, modify, and distribute this software for any purpose with or without fee is hereby granted, provided that the above copyright notice and this permission notice appear in all copies.

THE SOFTWARE IS PROVIDED “AS IS”, WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT OF THIRD PARTY RIGHTS. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

Except as contained in this notice, the name of a copyright holder shall not be used in advertising or otherwise to promote the sale, use or other dealings in this Software without prior written authorization of the copyright holder.

### 16.2.17 libgd License

Portions copyright 1994, 1995, 1996, 1997, 1998, 1999, 2000, 2001, 2002 by Cold Spring Harbor Laboratory. Funded under Grant P41-RR02188 by the National Institutes of Health.

Portions copyright 1996, 1997, 1998, 1999, 2000, 2001, 2002 by Boutell.Com, Inc.

Portions relating to GD2 format copyright 1999, 2000, 2001, 2002 Philip Warner.

Portions relating to PNG copyright 1999, 2000, 2001, 2002 Greg Roelofs.

Portions relating to gdttf.c copyright 1999, 2000, 2001, 2002 John Ellson (ellson@lucent.com).

Portions relating to gdft.c copyright 2001, 2002 John Ellson (ellson@lucent.com).

Portions copyright 2000, 2001, 2002, 2003, 2004, 2005, 2006, 2007 Pierre-Alain Joye (pierre@libgd.org).

Portions relating to JPEG and to color quantization copyright 2000, 2001, 2002, Doug Becker and copyright (C) 1994, 1995, 1996, 1997, 1998, 1999, 2000, 2001, 2002, Thomas G. Lane. This software is based in part on the work of the Independent JPEG Group. See the file README-JPEG.TXT for more information.

Portions relating to WBMP copyright 2000, 2001, 2002 Maurice Szmurlo and Johan Van den Brande.

Permission has been granted to copy, distribute and modify gd in any context without fee, including a commercial application, provided that this notice is present in user-accessible supporting documentation.

This does not affect your ownership of the derived work itself, and the intent is to assure proper credit for the authors of gd, not to interfere with your productive use of gd. If you have questions, ask. "Derived works" includes all programs that utilize the library. Credit must be given in user-accessible documentation.

This software is provided "AS IS." The copyright holders disclaim all warranties, either express or implied, including but not limited to implied warranties of merchantability and fitness for a particular purpose, with respect to this code and accompanying documentation.

Although their code does not appear in gd, the authors wish to thank David Koblas, David Rowley, and Hutchison Avenue Software Corporation for their prior contributions.

### 16.2.18 BSD License from Intel

All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

- Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
- Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
- Neither the name of the Intel Corporation nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY,

OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

### 16.2.19 libjpeg License

The authors make NO WARRANTY or representation, either express or implied, with respect to this software, its quality, accuracy, merchantability, or fitness for a particular purpose. This software is provided "AS IS", and you, its user, assume the entire risk as to its quality and accuracy.

This software is copyright (C) 1991-2010, Thomas G. Lane, Guido Vollbeding. All Rights Reserved except as specified below.

Permission is hereby granted to use, copy, modify, and distribute this software (or portions thereof) for any purpose, without fee, subject to these conditions:

(1) If any part of the source code for this software is distributed, then this README file must be included, with this copyright and no-warranty notice unaltered; and any additions, deletions, or changes to the original files must be clearly indicated in accompanying documentation.

(2) If only executable code is distributed, then the accompanying documentation must state that "this software is based in part on the work of the Independent JPEG Group".

(3) Permission for use of this software is granted only if the user accepts full responsibility for any undesirable consequences; the authors accept NO LIABILITY for damages of any kind.

These conditions apply to any software derived from or based on the IJG code, not just to the unmodified library. If you use our work, you ought to acknowledge us.

Permission is NOT granted for the use of any IJG author's name or company name in advertising or publicity relating to this software or products derived from it. This software may be referred to only as "the Independent JPEG Group's software".

We specifically permit and encourage the use of this software as the basis of commercial products, provided that all warranty or liability claims are assumed by the product vendor.

ansi2knr.c is included in this distribution by permission of L. Peter Deutsch, sole proprietor of its copyright holder, Aladdin Enterprises of Menlo Park, CA. ansi2knr.c is NOT covered by the above copyright and conditions, but instead by the usual distribution terms of the Free Software Foundation; principally, that you must include source code if you redistribute it. (See the file ansi2knr.c for full details.) However, since ansi2knr.c is not needed as part of any program generated from the IJG code, this does not limit you more than the foregoing paragraphs do.

The Unix configuration script "configure" was produced with GNU Autoconf. It is copyright by the Free Software Foundation but is freely distributable. The same holds for its supporting scripts (config.guess, config.sub, ltmain.sh). Another support script, install-sh, is copyright by X Consortium but is also freely distributable.

The IJG distribution formerly included code to read and write GIF files. To avoid entanglement with the Unisys LZW patent, GIF reading support has been removed altogether, and the GIF writer has been simplified to produce "uncompressed GIFs". This technique does not use the LZW algorithm; the resulting GIF files are larger than usual, but are readable by all standard GIF decoders.

We are required to state that

"The Graphics Interchange Format(c) is the Copyright property of CompuServe Incorporated. GIF(sm) is a Service Mark property of CompuServe Incorporated."

### 16.2.20 x86 SIMD Extension for IJG JPEG Library License

Copyright 2009 Pierre Ossman <ossman@cendio.se> for Cendio AB

Copyright 2010 D. R. Commander

Based on

x86 SIMD extension for IJG JPEG library - version 1.02

Copyright (C) 1999-2006, MIYASAKA Masaru.

This software is provided 'as-is', without any express or implied warranty. In no event will the authors be held liable for any damages arising from the use of this software.

Permission is granted to anyone to use this software for any purpose, including commercial applications, and to alter it and redistribute it freely, subject to the following restrictions:

1. The origin of this software must not be misrepresented; you must not claim that you wrote the original software. If you use this software in a product, an acknowledgment in the product documentation would be appreciated but is not required.

2. Altered source versions must be plainly marked as such, and must not be misrepresented as being the original software.

3. This notice may not be removed or altered from any source distribution.

### 16.2.21 FreeBSD License

The compilation of software known as FreeBSD is distributed under the following terms:

Copyright (c) 1992-2009 The FreeBSD Project. All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

THIS SOFTWARE IS PROVIDED BY THE AUTHOR AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE AUTHOR OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

### 16.2.22 NetBSD License

Copyright (c) 1992, 1993 The Regents of the University of California. All rights reserved.

This software was developed by the Computer Systems Engineering group at Lawrence Berkeley Laboratory under DARPA contract BG 91-66 and contributed to Berkeley.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

3. Neither the name of the University nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE REGENTS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE REGENTS OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

### 16.2.23 PCRE License

PCRE is a library of functions to support regular expressions whose syntax and semantics are as close as possible to those of the Perl 5 language.

Release 8 of PCRE is distributed under the terms of the "BSD" licence, as specified below. The documentation for PCRE, supplied in the "doc" directory, is distributed under the same terms as the software itself.

The basic library functions are written in C and are freestanding. Also included in the distribution is a set of C++ wrapper functions, and a just-in-time compiler that can be used to optimize pattern matching. These are both optional features that can be omitted when the library is built.

THE BASIC LIBRARY FUNCTIONS. Written by: Philip Hazel; Email local part: ph10; Email domain: cam.ac.uk University of Cambridge Computing Service, Cambridge, England. Copyright (c) 1997-2012 University of Cambridge All rights reserved.

PCRE JUST-IN-TIME COMPILATION SUPPORT. Written by: Zoltan Herczeg; Email local part: hzmester; Emain domain: freemail.hu Copyright(c) 2010-2012 Zoltan Herczeg All rights reserved.

STACK-LESS JUST-IN-TIME COMPILER. Written by: Zoltan Herczeg; Email local part: hzmester; Emain domain: freemail.hu Copyright(c) 2009-2012 Zoltan Herczeg All rights reserved.

THE C++ WRAPPER FUNCTIONS. Contributed by: Google Inc. Copyright (c) 2007-2012, Google Inc. All rights reserved.

THE "BSD" LICENCE. Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

- Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
- Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
- Neither the name of the University of Cambridge nor the name of Google Inc. nor the names of their contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

### 16.2.24 libffi License

Copyright (c) 1996-2012 Anthony Green, Red Hat, Inc and others. See source files for details.

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

### 16.2.25 FLTK License

December 11, 2001

The FLTK library and included programs are provided under the terms of the GNU Library General Public License (LGPL) with the following exceptions:

1. Modifications to the FLTK configure script, config header file, and makefiles by themselves to support a specific platform do not constitute a modified or derivative work.

   The authors do request that such modifications be contributed to the FLTK project - send all contributions through the "Software Trouble Report" on the following page:

   http://www.fltk.org/str.php

2. Widgets that are subclassed from FLTK widgets do not constitute a derivative work.

3. Static linking of applications and widgets to the FLTK library does not constitute a derivative work and does not require the author to provide source code for the application or widget, use the shared FLTK libraries, or link their applications or widgets against a user-supplied version of FLTK.

   If you link the application or widget to a modified version of FLTK, then the changes to FLTK must be provided under the terms of the LGPL in sections 1, 2, and 4.

4. You do not have to provide a copy of the FLTK license with programs that are linked to the FLTK library, nor do you have to identify the FLTK license in your program or documentation as required by section 6 of the LGPL.

   However, programs must still identify their use of FLTK. The following example statement can be included in user documentation to satisfy this requirement:

   [program/widget] is based in part on the work of the FLTK project (http://www.fltk.org).

### 16.2.26 Expat License

Copyright (c) 1998, 1999, 2000 Thai Open Source Software Center Ltd and Clark Cooper

Copyright (c) 2001, 2002, 2003, 2004, 2005, 2006 Expat maintainers.

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute,

sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

## 16.2.27 Fontconfig License

Copyright (C) 2001, 2003 Keith Packard

Permission to use, copy, modify, distribute, and sell this software and its documentation for any purpose is hereby granted without fee, provided that the above copyright notice appear in all copies and that both that copyright notice and this permission notice appear in supporting documentation, and that the name of the author(s) not be used in advertising or publicity pertaining to distribution of the software without specific, written prior permission. The authors make no representations about the suitability of this software for any purpose. It is provided "as is" without express or implied warranty.

THE AUTHOR(S) DISCLAIMS ALL WARRANTIES WITH REGARD TO THIS SOFTWARE, INCLUDING ALL IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS, IN NO EVENT SHALL THE AUTHOR(S) BE LIABLE FOR ANY SPECIAL, INDIRECT OR CONSEQUENTIAL DAMAGES OR ANY DAMAGES WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR PROFITS, WHETHER IN AN ACTION OF CONTRACT, NEGLIGENCE OR OTHER TORTIOUS ACTION, ARISING OUT OF OR IN CONNECTION WITH THE USE OR PERFORMANCE OF THIS SOFTWARE.

## 16.2.28 Freetype License

2006-Jan-27

Copyright 1996-2002, 2006 by David Turner, Robert Wilhelm, and Werner Lemberg

### 16.2.28.1 Introduction

The FreeType Project is distributed in several archive packages; some of them may contain, in addition to the FreeType font engine, various tools and contributions which rely on, or relate to, the FreeType Project.

This license applies to all files found in such packages, and which do not fall under their own explicit license. The license affects thus the FreeType font engine, the test programs, documentation and makefiles, at the very least.

This license was inspired by the BSD, Artistic, and IJG (Independent JPEG Group) licenses, which all encourage inclusion and use of free software in commercial and freeware products alike. As a consequence, its main points are that:

- We don't promise that this software works. However, we will be interested in any kind of bug reports. ('as is' distribution)
- You can use this software for whatever you want, in parts or full form, without having to pay us. ('royalty-free' usage)
- You may not pretend that you wrote this software. If you use it, or only parts of it, in a program, you must acknowledge somewhere in your documentation that you have used the FreeType code. ('credits')

We specifically permit and encourage the inclusion of this software, with or without modifications, in commercial products. We disclaim all warranties covering The FreeType Project and assume no liability related to The FreeType Project.

Finally, many people asked us for a preferred form for a credit/disclaimer to use in compliance with this license. We thus encourage you to use the following text:

Portions of this software are copyright (C) <year> The FreeType Project (www.freetype.org). All rights reserved.

Please replace <year> with the value from the FreeType version you actually use.

#### 16.2.28.2 Legal Terms

0. Definitions

Throughout this license, the terms 'package', 'FreeType Project', and 'FreeType archive' refer to the set of files originally distributed by the authors (David Turner, Robert Wilhelm, and Werner Lemberg) as the 'FreeType Project', be they named as alpha, beta or final release.

'You' refers to the licensee, or person using the project, where 'using' is a generic term including compiling the project's source code as well as linking it to form a 'program' or 'executable'. This program is referred to as 'a program using the FreeType engine'.

This license applies to all files distributed in the original FreeType Project, including all source code, binaries and documentation, unless otherwise stated in the file in its original, unmodified form as distributed in the original archive. If you are unsure whether or not a particular file is covered by this license, you must contact us to verify this.

The FreeType Project is copyright (C) 1996-2000 by David Turner, Robert Wilhelm, and Werner Lemberg. All rights reserved except as specified below.

1. No Warranty

THE FREETYPE PROJECT IS PROVIDED 'AS IS' WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. IN NO EVENT WILL ANY OF THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY DAMAGES CAUSED BY THE USE OR THE INABILITY TO USE, OF THE FREETYPE PROJECT.

2. Redistribution

This license grants a worldwide, royalty-free, perpetual and irrevocable right and license to use, execute, perform, compile, display, copy, create derivative works of, distribute and sublicense the FreeType Project (in both source and object code forms) and derivative works thereof for any purpose; and to authorize others to exercise some or all of the rights granted herein, subject to the following conditions:

- Redistribution of source code must retain this license file ('FTL.TXT') unaltered; any additions, deletions or changes to the original files must be clearly indicated in accompanying documentation. The copyright notices of the unaltered, original files must be preserved in all copies of source files.
- Redistribution in binary form must provide a disclaimer that states that the software is based in part of the work of the FreeType Team, in the distribution documentation. We also encourage you to put an URL to the FreeType web page in your documentation, though this isn't mandatory.

These conditions apply to any software derived from or based on the FreeType Project, not just the unmodified files. If you use our work, you must acknowledge us. However, no fee need be paid to us.

3. Advertising

Neither the FreeType authors and contributors nor you shall use the name of the other for commercial, advertising, or promotional purposes without specific prior written permission.

We suggest, but do not require, that you use one or more of the following phrases to refer to this software in your documentation or advertising materials: 'FreeType Project', 'FreeType Engine', 'FreeType library', or 'FreeType Distribution'.

As you have not signed this license, you are not required to accept it. However, as the FreeType Project is copyrighted material, only this license, or another one contracted with the authors, grants you the right to use, distribute, and modify it. Therefore, by using, distributing, or modifying the FreeType Project, you indicate that you understand and accept all the terms of this license.

4. Contacts

There are two mailing lists related to FreeType:

- freetype@nongnu.org

  Discusses general use and applications of FreeType, as well as future and wanted additions to the library and distribution. If you are looking for support, start in this list if you haven't found anything to help you in the documentation.

- freetype-devel@nongnu.org

  Discusses bugs, as well as engine internals, design issues, specific licenses, porting, etc.

Our home page can be found at
http://www.freetype.org

### 16.2.29 VPX License

Copyright (c) 2010, The WebM Project authors. All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

- Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
- Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
- Neither the name of Google, nor the WebM Project, nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

### 16.2.30 Opus License

Copyright 2001-2011 Xiph.Org, Skype Limited, Octasic, Jean-Marc Valin, Timothy B. Terriberry, CSIRO, Gregory Maxwell, Mark Borgerding, Erik de Castro Lopo

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

- Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
- Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
- Neither the name of Internet Society, IETF or IETF Trust, nor the names of specific contributors, may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS ”AS IS” AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE. Opus is subject to the royalty-free patent licenses which are specified at: Xiph.Org Foundation: https://datatracker.ietf.org/ipr/1524/ Microsoft Corporation: https://datatracker.ietf.org/ipr/1914/ Broadcom Corporation: https://datatracker.ietf.org/ipr/1526/

### 16.2.31 FUSE for macOS License

Copyright (c) 2011-2017 Benjamin Fleischer; Copyright (c) 2011-2012 Erik Larsson; All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

- Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
- Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
- Neither the name of the copyright holder nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS ”AS IS” AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

# 17 Oracle VM VirtualBox Privacy Information

Version 5, Dec 13, 2012

The Oracle Privacy Policies posted on http://www.oracle.com/html/privacy.html apply to your personal data collected and used by Oracle. The following privacy information describes in more detail which information is exchanged between the Oracle VM VirtualBox application and Oracle, and which information is collected by the virtualbox.org website.

**§ 1 virtualbox.org.** The "virtualbox.org" website logs anonymous usage information such as your IP address, geographical location, browser type, referral source, length of visit and number of page views while you visit (collectively, "anonymous data"). In addition, but only if you choose to register, the website's bug tracking and forum services store the data you choose to reveal upon registration, such as your user name and contact information.

**§ 2 Cookies.** The virtualbox.org website, the bug tracker and the forum services use cookies to identify and track the visiting web browser and, if you have registered, to facilitate login. Most browsers allow you to refuse to accept cookies. While you can still visit the website with cookies disabled, logging into the bug tracker and forum services will most likely not work without them.

**§ 3 Oracle VM VirtualBox registration process.** The Oracle VM VirtualBox application may ask that the user optionally register with Oracle. If you choose to register, your name, e-mail address, country and company will be submitted to Oracle and stored together with the IP address of the submitter as well as product version and platform being used.

**§ 4 Update notifications.** The Oracle VM VirtualBox application may contact Oracle to find out whether a new version of Oracle VM VirtualBox has been released and notify the user if that is the case. In the process, anonymous data such as your IP address and a non-identifying counter, together with the product version and the platform being used, is sent so that the server can find out whether an update is available. By default, this check is performed once a day. You change this interval or disable these checks altogether in the Oracle VM VirtualBox preferences.

**§ 5 Usage of personal information.** Oracle may use anonymous and personal data collected by the means above for statistical purposes as well as to automatically inform you about new notices related to your posts on the bug tracker and forum services, to administer the website and to contact you due to technical issues. Oracle may also inform you about new product releases related to Oracle VM VirtualBox.

In no event will personal data without your express consent be provided to any third parties, unless Oracle may be required to do so by law or in connection with legal proceedings.

**§ 6 Updates.** Oracle may update the privacy policy at any time by posting a new version at http://www.oracle.com/html/privacy.html and the privacy information will be kept up to date in the documentation which comes with the Oracle VM VirtualBox application. You should check these places occasionally to ensure you are happy with any changes.

# Glossary

## A

**ACPI** Advanced Configuration and Power Interface, an industry specification for BIOS and hardware extensions to configure PC hardware and perform power management. Windows 2000 and later, as well as Linux 2.4 and later support ACPI. Windows can only enable or disable ACPI support at installation time.

**AHCI** Advanced Host Controller Interface, the interface that supports SATA devices such as hard disks. See chapter 5.1, *Hard Disk Controllers: IDE, SATA (AHCI), SCSI, SAS, USB MSD, NVMe*, page 83.

**AMD-V** The hardware virtualization features built into modern AMD processors. See chapter 10.3, *Hardware vs. Software Virtualization*, page 270.

**API** Application Programming Interface.

**APIC** Advanced Programmable Interrupt Controller, a newer version of the original PC PIC (programmable interrupt controller). Most modern CPUs contain an on-chip APIC, called a local APIC. Many systems also contain an I/O APIC (input output APIC) as a separate chip which provides more than 16 IRQs. Windows 2000 and later use a different kernel if they detect an I/O APIC during installation. Therefore, an I/O APIC must not be removed after installation.

**ATA** Advanced Technology Attachment, an industry standard for hard disk interfaces which is synonymous with IDE. See chapter 5.1, *Hard Disk Controllers: IDE, SATA (AHCI), SCSI, SAS, USB MSD, NVMe*, page 83.

## B

**BIOS** Basic Input/Output System, the firmware built into most personal computers which is responsible of initializing the hardware after the computer has been turned on and then booting an operating system. Oracle VM VirtualBox ships with its own virtual BIOS that runs when a virtual machine is started.

## C

**COM** Microsoft Component Object Model, a programming infrastructure for modular software. COM enables applications to provide application programming interfaces which can be accessed from various other programming languages and applications. Oracle VM VirtualBox makes use of COM both internally and externally to provide a comprehensive API to 3rd party developers.

## D

**DHCP** Dynamic Host Configuration Protocol. This enables a networking device in a network to acquire its IP address and other networking details automatically, in order to avoid having to configure all devices in a network with fixed IP addresses. Oracle VM VirtualBox has a built-in DHCP server that delivers an IP addresses to a virtual machine when networking is configured to NAT. See chapter 6, *Virtual Networking*, page 98.

## E

**EFI** Extensible Firmware Interface, a firmware built into computers which is designed to replace the aging BIOS. Originally designed by Intel, most modern operating systems can now boot on computers which have EFI instead of a BIOS built into them. See chapter 3.14, *Alternative Firmware (EFI)*, page 58.

**EHCI** Enhanced Host Controller Interface, the interface that implements the USB 2.0 standard.

## G

**GUI** Graphical User Interface. Commonly used as an antonym to a "command line interface". In the context of Oracle VM VirtualBox, we sometimes refer to the main graphical `VirtualBox` program as the "GUI", to differentiate it from the `VBoxManage` interface.

**GUID** See UUID.

## I

**IDE** Integrated Drive Electronics, an industry standard for hard disk interfaces. See chapter 5.1, *Hard Disk Controllers: IDE, SATA (AHCI), SCSI, SAS, USB MSD, NVMe*, page 83.

**I/O APIC** See APIC.

**iSCSI** Internet SCSI. See chapter 5.10, *iSCSI Servers*, page 95.

## M

**MAC** Media Access Control, a part of an Ethernet network card. A MAC address is a 6-byte number which identifies a network card. It is typically written in hexadecimal notation where the bytes are separated by colons, such as `00:17:3A:5E:CB:08`.

**MSI** Message Signaled Interrupts, as supported by modern chipsets such as the ICH9. See chapter 3.5.1, *Motherboard Tab*, page 47. As opposed to traditional pin-based interrupts, with MSI, a small amount of data can accompany the actual interrupt message. This reduces the amount of hardware pins required and allows for more interrupts and better performance.

## N

**NAT** Network Address Translation. A technique to share networking interfaces by which an interface modifies the source and/or target IP addresses of network packets according to specific rules. Commonly employed by routers and firewalls to shield an internal network

from the Internet, Oracle VM VirtualBox can use NAT to easily share a host's physical networking hardware with its virtual machines. See chapter 6.3, *Network Address Translation (NAT)*, page 100.

## O

**OVF** Open Virtualization Format, a cross-platform industry standard to exchange virtual appliances between virtualization products. See chapter 1.15, *Importing and Exporting Virtual Machines*, page 21.

## P

**PAE** Physical Address Extension. This enables access to more than 4 GB of RAM, even in 32-bit environments. See chapter 3.4.2, *Advanced Tab*, page 46.

**PIC** See APIC.

**PXE** Preboot Execution Environment, an industry standard for booting PC systems from remote network locations. It includes DHCP for IP configuration and TFTP for file transfer. Using UNDI, a hardware independent driver stack for accessing the network card from bootstrap code is available.

## R

**RDP** Remote Desktop Protocol, a protocol developed by Microsoft as an extension to the ITU T.128 and T.124 video conferencing protocol. With RDP, a PC system can be controlled from a remote location using a network connection over which data is transferred in both directions. Typically graphics updates and audio are sent from the remote machine and keyboard and mouse input events are sent from the client. An Oracle VM VirtualBox extension package by Oracle provides VRDP, an enhanced implementation of the relevant standards which is largely compatible with Microsoft's RDP implementation. See chapter 7.1, *Remote Display (VRDP Support)*, page 110 for details.

## S

**SAS** Serial Attached SCSI, an industry standard for hard disk interfaces. See chapter 5.1, *Hard Disk Controllers: IDE, SATA (AHCI), SCSI, SAS, USB MSD, NVMe*, page 83.

**SATA** Serial ATA, an industry standard for hard disk interfaces. See chapter 5.1, *Hard Disk Controllers: IDE, SATA (AHCI), SCSI, SAS, USB MSD, NVMe*, page 83.

**SCSI** Small Computer System Interface. An industry standard for data transfer between devices, especially for storage. See chapter 5.1, *Hard Disk Controllers: IDE, SATA (AHCI), SCSI, SAS, USB MSD, NVMe*, page 83.

**SMP** Symmetrical Multiprocessing, meaning that the resources of a computer are shared between several processors. These can either be several processor chips or, as is more common with modern hardware, multiple CPU cores in one processor.

**SSD** Solid-state drive, uses microchips for storing data in a computer system. Compared to classical hard-disks they are having no mechanical components like spinning disks.

## T

**TAR** A widely used file format for archiving. Originally, this stood for Tape ARchive and was already supported by very early UNIX versions for backing up data on tape. The file format is still widely used today. For example, with OVF archives using an `.ova` file extension. See chapter 1.15, *Importing and Exporting Virtual Machines*, page 21.

## U

**UUID** A Universally Unique Identifier, often also called GUID (Globally Unique Identifier). A UUID is a string of numbers and letters which can be computed dynamically and is guaranteed to be unique. Generally, it is used as a global handle to identify entities. Oracle VM VirtualBox makes use of UUIDs to identify VMs, Virtual Disk Images (VDI files), and other entities.

## V

**VM** Virtual Machine. A virtual computer that Oracle VM VirtualBox enables you to run on top of your actual hardware. See chapter 1.2, *Some Terminology*, page 2 for details.

**VMM** Virtual Machine Manager. The component of Oracle VM VirtualBox that controls VM execution. See chapter 10.2, *Oracle VM VirtualBox Executables and Components*, page 268 for a list of Oracle VM VirtualBox components.

**VRDE** VirtualBox Remote Desktop Extension. This interface is built into Oracle VM VirtualBox to allow Oracle VM VirtualBox extension packages to supply remote access to virtual machines. An Oracle VM VirtualBox extension package by Oracle provides VRDP support. See chapter 7.1, *Remote Display (VRDP Support)*, page 110.

**VRDP** See RDP.

**VT-x** The hardware virtualization features built into modern Intel processors. See chapter 10.3, *Hardware vs. Software Virtualization*, page 270.

## X

**xHCI** eXtended Host Controller Interface, the interface that implements the USB 3.0 standard.

**XML** The eXtensible Markup Language, a metastandard for all kinds of textual information. XML only specifies how data in the document is organized generally and does not prescribe how to semantically organize content.

**XPCOM** Mozilla Cross Platform Component Object Model, a programming infrastructure developed by the Mozilla browser project which is similar to Microsoft COM and enables applications to provide a modular programming interface. Oracle VM VirtualBox makes use of XPCOM on Linux both internally and externally to provide a comprehensive API to third-party developers.