

Oracle® VM
Server User's Guide
Release 2.2
E15444-04

June 2011

Oracle VM Server User's Guide, Release 2.2

E15444-04

Copyright © 2008, 2011, Oracle and/or its affiliates. All rights reserved.

Primary Author: Alison Holloway

Contributing Authors: John Russell, Kurt Hackel, Herbert van den Bergh, Tatyana Bagerman

Contributor: Chris Barclay, Michael Chan, Adam Hawley, Steve Noyes, Keshav Sharma, Honglin Su, Carol Tian, Junjie Wei, Lisa Vaz

This software and related documentation are provided under a license agreement containing restrictions on use and disclosure and are protected by intellectual property laws. Except as expressly permitted in your license agreement or allowed by law, you may not use, copy, reproduce, translate, broadcast, modify, license, transmit, distribute, exhibit, perform, publish, or display any part, in any form, or by any means. Reverse engineering, disassembly, or decompilation of this software, unless required by law for interoperability, is prohibited.

The information contained herein is subject to change without notice and is not warranted to be error-free. If you find any errors, please report them to us in writing.

If this is software or related documentation that is delivered to the U.S. Government or anyone licensing it on behalf of the U.S. Government, the following notice is applicable:

U.S. GOVERNMENT RIGHTS Programs, software, databases, and related documentation and technical data delivered to U.S. Government customers are "commercial computer software" or "commercial technical data" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, the use, duplication, disclosure, modification, and adaptation shall be subject to the restrictions and license terms set forth in the applicable Government contract, and, to the extent applicable by the terms of the Government contract, the additional rights set forth in FAR 52.227-19, Commercial Computer Software License (December 2007). Oracle America, Inc., 500 Oracle Parkway, Redwood City, CA 94065.

This software or hardware is developed for general use in a variety of information management applications. It is not developed or intended for use in any inherently dangerous applications, including applications that may create a risk of personal injury. If you use this software or hardware in dangerous applications, then you shall be responsible to take all appropriate fail-safe, backup, redundancy, and other measures to ensure its safe use. Oracle Corporation and its affiliates disclaim any liability for any damages caused by use of this software or hardware in dangerous applications.

Oracle and Java are registered trademarks of Oracle and/or its affiliates. Other names may be trademarks of their respective owners.

Intel and Intel Xeon are trademarks or registered trademarks of Intel Corporation. All SPARC trademarks are used under license and are trademarks or registered trademarks of SPARC International, Inc. AMD, Opteron, the AMD logo, and the AMD Opteron logo are trademarks or registered trademarks of Advanced Micro Devices. UNIX is a registered trademark of The Open Group.

This software or hardware and documentation may provide access to or information on content, products, and services from third parties. Oracle Corporation and its affiliates are not responsible for and expressly disclaim all warranties of any kind with respect to third-party content, products, and services. Oracle Corporation and its affiliates will not be responsible for any loss, costs, or damages incurred due to your access to or use of third-party content, products, or services.

Contents

Preface	vii
Audience	vii
Documentation Accessibility	vii
Command Syntax	vii
Related Documents	viii
Conventions	viii
 What's New in Oracle VM Server?	xi
New Features in Release 2.2.2	xi
New Features in Release 2.2.1	xii
New Features in Release 2.2.0	xiii
New Features in Release 2.1.5	xiii
New Features in Release 2.1.2	xiv
New Features in Release 2.1.1	xiv
 1 Introduction to Virtualization	
1.1 What is Virtualization?	1-1
1.2 Why Virtualize?	1-1
1.3 Xen™ Technology	1-2
1.4 Oracle VM	1-2
 2 Oracle VM Server	
2.1 Oracle VM Server	2-1
2.2 Hypervisor	2-2
2.3 Domains, Guests and Virtual Machines	2-2
2.4 Management Domain	2-3
2.5 Domains	2-3
2.6 Paravirtualization, Hardware Virtualization, and Binary Translation	2-3
2.7 Creating Virtual Machines	2-3
2.8 Managing Domains	2-3
2.9 Configuring Oracle VM Server	2-4
2.10 Managing Oracle VM Server Repositories	2-4

3 Oracle VM Agent

3.1	Oracle VM Agent Command-Line Tool	3-1
3.2	Configuring Oracle VM Agent.....	3-1
3.3	Starting Oracle VM Agent	3-3
3.4	Stopping Oracle VM Agent	3-3
3.5	Monitoring Oracle VM Agent	3-3

4 Creating a Guest Virtual Machine

4.1	Supported Guest Operating Systems.....	4-1
4.2	Creating an Installation Tree	4-2
4.3	Creating a Guest Using a Template.....	4-2
4.4	Creating a Guest Using virt-install.....	4-3
4.5	Creating a Paravirtualized Guest Manually	4-6
4.5.1	Creating the Root File System.....	4-6
4.5.2	Populating the Root File System.....	4-7
4.5.3	Configuring the Guest.....	4-8
4.6	Creating a Hardware Virtualized Guest Manually	4-9
4.7	Converting a Hardware Virtualized Guest to a Paravirtualized Guest	4-11
4.8	Installing Paravirtual Drivers.....	4-13

5 Domain Monitoring and Administration

5.1	Domain Lifecycle	5-1
5.2	Using the xm Command-Line Interface	5-1
5.2.1	Monitoring Domains	5-2
5.2.2	Viewing Host Information	5-2

6 Domain Live Migration

6.1	Creating Shared Storage	6-1
6.2	Migrating a Domain	6-1

7 High Availability

7.1	High Availability (HA)	7-1
7.2	Creating Shared Storage	7-4
7.3	Enabling HA	7-4

8 Preparing External Storage and Storage Repositories

8.1	Preparing External Storage.....	8-2
8.1.1	Preparing External Storage Using OCFS2 on iSCSI.....	8-2
8.1.2	Preparing External Storage Using OCFS2 on SAN.....	8-4
8.1.3	Preparing External Storage Using NFS	8-4
8.2	Preparing a Storage Repository	8-5

9 Managing Storage Repositories

9.1	Storage Repository Directory Structure.....	9-1
-----	---	-----

9.2	Managing Storage Repositories	9-2
9.2.1	Listing the Storage Repositories	9-2
9.2.2	Adding a Storage Repository	9-3
9.2.3	Deleting a Storage Repository	9-3
9.2.4	Initializing the Storage Repositories	9-4
9.2.5	Setting the Cluster Root	9-4
10	Converting Hosts and Non-Oracle VM Virtual Machines	
10.1	Converting a Linux or Windows Host	10-1
10.1.1	Using the P2V Utility Interactively	10-2
10.1.2	Using the P2V Utility with a Kickstart File	10-5
10.2	Converting a Non-Oracle VM Virtual Machine	10-6
A	Command-Line Tools	
	ovs-agent	A-2
	virt-install	A-3
	xm	A-7
	P2V	A-12
B	Oracle VM Server Configuration File	
	Oracle VM Server Configuration File	B-2
C	Guest Configuration	
C.1	e100 And e1000 Network Device Emulators	C-1
C.2	Quality of Service (QoS)	C-2
C.2.1	Setting Disk Priority	C-2
C.2.2	Setting Inbound Network Traffic Priority	C-2
C.2.3	Setting Outbound Network Traffic Priority	C-3
C.3	Virtual CPU Configuration File Parameters	C-3
C.4	Simple Configuration File Example	C-4
C.5	Complex Configuration File Example	C-4
C.6	Virtual Iron Migration (VHD) Configuration File Example	C-5
D	Oracle VM Agent Architecture	
D.1	Oracle VM Agent Architecture	D-1
D.2	Oracle VM Agent Deployment	D-2
E	Troubleshooting	
E.1	Debugging Tools	E-1
E.1.1	Oracle VM Server Directories	E-1
E.1.2	Oracle VM Server Log Files	E-2
E.1.3	Oracle VM Server Command-Line Tools	E-2
E.2	Using DHCP	E-2
E.3	Guest Console Access	E-2

E.4	Cannot Display Graphical Installer When Creating Guests.....	E-4
E.5	Hardware Virtualized Guest Console Not Displayed.....	E-4
E.6	Setting the Guest's Clock	E-5
E.7	Wallclock Time Skew Problems.....	E-5
E.8	Mouse Pointer Tracking Problems	E-5
E.9	Hardware Virtualized Guest Stops	E-6
E.10	Hardware Virtualized Guest Devices Not Working as Expected.....	E-6
E.11	CD-ROM Image Not Found	E-6
E.12	Firewall Blocks NFS Access.....	E-6
E.13	Migrating Domains.....	E-6
E.14	Attaching to a Console with the Grub Boot Loader.....	E-7

Glossary

Index

Preface

The preface contains information on how to use the Oracle VM Server User's Guide. The areas discussed are:

- [Audience](#)
- [Documentation Accessibility](#)
- [Command Syntax](#)
- [Related Documents](#)
- [Conventions](#)

Audience

The Oracle VM Server User's Guide is intended for system administrators and end users who want to learn the fundamentals of virtualization and the provision of virtual guest operating systems.

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 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.

Command Syntax

UNIX command syntax appears in monospace font. The dollar character (\$), number sign (#), or percent character (%) are UNIX command prompts. Do not enter them as part of the command. The following command syntax conventions are used in this guide:

Convention	Description
backslash \	A backslash is the UNIX command continuation character. It is used in command examples that are too long to fit on a single line. Enter the command as displayed (with a backslash) or enter it on a single line without a backslash: dd if=/dev/rdisk/c0t1d0s6 of=/dev/rst0 bs=10b \ count=10000
braces { }	Braces indicate required items: .DEFINE {macro1}
brackets []	Brackets indicate optional items: cvtcrtr termname [outfile]
ellipses ...	Ellipses indicate an arbitrary number of similar items: CHKVAL fieldname value1 value2 ... valueN
<i>italics</i>	Italic type indicates a variable. Substitute a value for the variable: library_name
vertical line	A vertical line indicates a choice within braces or brackets: FILE filesize [K M]

Related Documents

For more information, see the following documents in the Oracle VM Release 2.2 documentation set:

- *Oracle VM Server Quick Start Guide*
- *Oracle VM Server Release Notes*
- *Oracle VM Server Installation Guide*
- *Oracle VM Manager Release Notes*
- *Oracle VM Manager Installation Guide*
- *Oracle VM Manager User's Guide*
- *Oracle VM Windows Paravirtual Drivers Installation Guide*
- *Oracle VM Template Builder Installation and User's Guide*
- *Oracle VM Manager Web Services API Reference*

You can also get the latest information on Oracle VM by going to the Oracle virtualization Web site:

<http://www.oracle.com/virtualization>

Conventions

The following text conventions are used in this document:

Convention	Meaning
boldface	Boldface type indicates graphical user interface elements associated with an action, or terms defined in text or the glossary.
<i>italic</i>	Italic type indicates book titles, emphasis, or placeholder variables for which you supply particular values.

Convention	Meaning
monospace	Monospace type indicates commands within a paragraph, URLs, code in examples, text that appears on the screen, or text that you enter.

What's New in Oracle VM Server?

This Preface introduces the new features and enhancements of Oracle VM Server. This information is useful to users who have used previous releases of Oracle VM Server. The releases discussed are:

- [New Features in Release 2.2.2](#)
- [New Features in Release 2.2.1](#)
- [New Features in Release 2.2.0](#)
- [New Features in Release 2.1.5](#)
- [New Features in Release 2.1.2](#)
- [New Features in Release 2.1.1](#)

New Features in Release 2.2.2

The new features and enhancements in Oracle VM Server Release 2.2.2 include:

- The VMPInfo system information and cluster troubleshooting utility is provided with this release. VMPInfo is a tool to gather detailed information from a single machine, group of servers or an integrated cluster and package that information in a concise report archive. Archives from multiple nodes can then be merged into a cluster oriented report that flags problems and expedites the process of verifying best practices and operational correctness for new and existing Oracle VM and Oracle Linux deployments. The VMPInfo package is not installed by default and can be found on the Oracle VM Server CD (or ISO file) as:

`/Server/vmpinfo-2.2-3.noarch.rpm`

To install the VMPInfo package, copy the RPM to the Oracle VM Server(s) and run the following command to install it:

```
# rpm -i vmpinfo-2.2-3.noarch.rpm
```

Running `vmpinfo -k` performs a health check on the Oracle VM Server and displays the results on the console.

Running `vmpinfo` without parameters generates a comprehensive scan and places the report archive in `/tmp/vmpinfo`. These archive can then be sent to a remote analyst, transferred to a local workstation for review, or both.

For more details on using the VMPInfo package, display the inbuilt help by running `vmpinfo -m`.

Additional information on VMPInfo is also available in the following My Oracle Support Notes:

- 1263293.1 Post-installation check list for new Oracle VM Server
- 1290587.1 Performing Site Reviews and Cluster Troubleshooting with VMPInfo
- A new storage repository option to provide NFS mount options when creating a storage repository. The syntax to use is:
`/opt/ovs-agent-2.3/utils/repos.py -n nfs-server:/mount-point -o NFS-options`
For more information on this new parameter, see [Section 9.2.2, "Adding a Storage Repository"](#).
- Updated the OCFS2 cluster file system to OCFS2 Release 1.4.8.
- Updated libdhcp to 1.20.10.
- Updated device drivers for:
 - qla2xxx 8.03.07.00.2.2.1
 - netxen 4.0.75
 - qlcnict 5.0.15.2
 - qla4xxx 5.02.04.02.02.22-d0
 - lpfc 8.2.0.48.2p
 - rdac 09.03.0C02.0331
 - 3w-9xxx driver up to date
- Added the Mellanox 10GbE Release 1.5.1.3 network driver.
- Added Intel network drivers for:
 - igbvf
 - ixgbevf
- Added a 64-bit version of kdump to enable dumping of vmcore on Oracle VM Servers with large memory.
- Added ext4 support for pygrub. This enables a paravirtualized guest with an ext4 boot partition to start up.
- Increased the value of txqueuelen for netback vif devices to increase network performance.

New Features in Release 2.2.1

The new features and enhancements in Oracle VM Server Release 2.2.1 include:

- Improved the serviceability and reliability of managing the server pool with large number of nodes (up to 32 nodes per pool):
 - Serializing the requests.
 - Increasing the default time-out value.
 - Improving the performance for concurrent requests.
- Enhanced pre-check function for agent to ensure successful setup of the server pool and OCFS2 cluster integrity:
 - Before adding a physical server to the server pool, Oracle VM agent checks the Oracle VM Server version, Oracle VM Agent version, CPU compatibility,

storage repositories, NFS, OCFS2 configuration, and so on. It is to ensure successful setup of the server pool and OCFS2 cluster integrity.

- New Oracle VM Agent cleanup utility to perform tasks such as:

- Stop OCFS2 cluster service (o2cb) heartbeat.
- Offline OCFS2 cluster service (o2cb).
- Remove o2cb configuration file.
- Un-mount ovs-agent storage repositories.
- Clean up ovs-agent local database.

Simply run the command from the command line:

```
# /opt/ovs-agent-2.3/utils/cleanup.py
```

- High Availability (HA) Enhancement:
 - In the current HA setup, if the virtual machine is stopped by any method other than Oracle VM Manager (or Enterprise Manager), the virtual machine is always restarted.
 - The HA enhancement is provided for the Oracle Linux 4 and 5 PV guests. The Oracle VM Agent does not restart the virtual machines if they are gracefully shutdown from within the guests or from dom0. For other HVM guests, there is no HA behavior change.
 - To have this HA enhancement, make sure that you have Oracle VM Agent ovs-agent-2.3-38 and the updated Xen 3.4.0-0.1.10 or later.

New Features in Release 2.2.0

The new features and enhancements in Oracle VM Server Release 2.2.0 include:

- Upgrade of the hypervisor to Xen 3.4. Xen 3.4 provides more efficient power management capabilities, broader hardware support, and better performance, scalability and security for both hardware and paravirtualized guests.
- Upgrade of OCFS2 to Release 1.4. Documentation for OCFS2 is available at:
<http://oss.oracle.com/projects/ocfs2/documentation/v1.4/>
- Support for the Intel® Microarchitecture CPU (code named Nehalem), such as Intel Xeon 5500 series processors.
- An improved storage repository and cluster configuration script to reduce external storage set up complexity. The new script is /opt/ovs-agent-2.3/utils/repos.py. This new script replaces the previous scripts, ovs-makerepo and ovs-offlinerepo in the /usr/lib/ovs/ directory, and the /etc/init.d/ovsrepositories script. See [Section 8.2, "Preparing a Storage Repository"](#) and [Chapter 9, "Managing Storage Repositories"](#).

New Features in Release 2.1.5

There are no new features in Oracle VM Server Release 2.1.5. A number of software and documentation errata have been fixed.

New Features in Release 2.1.2

The new features and enhancements in Oracle VM Server Release 2.1.2 include:

- High Availability of server pools and guests. See [Chapter 7, "High Availability"](#), and the *Oracle VM Manager User's Guide*.
- Converting a Linux host to a guest image, and converting a VMware image to an Oracle VM image. See [Chapter 10, "Converting Hosts and Non-Oracle VM Virtual Machines"](#).
- Quality of Service configuration options for guest virtual network interfaces and virtual disks. See [Quality of Service \(QoS\)](#) in [Appendix C, "Guest Configuration"](#), and the *Oracle VM Manager User's Guide*.
- Updated device drivers for:
 - Intel 8255x 10/100 Mbps (e100)
 - Intel 82540EM Gigabit (e1000)
 - Broadcom NetXtreme II (bnx2)
 - Broadcom Everest 10GB (bnx2x)
 - Broadcom NetXtreme and Netlink (tg3)
 - Chelsio T3 Third Generation 10Gb (cxgb3)
 - Qlogic Fibre Channel HBA (qla2xxx)
 - QLogic QLE8000
 - Emulex Fibre Channel HBA (lpfc)
 - Brocade Fibre Channel HBA (bfa)
 - LSI Logic Fusion-MPT SCSI (mptlinux)
 - LSI Logic MegaRAID SCSI (megaraid)
 - Redundant Disk Array Controller (rdac)

See [e100 And e1000 Network Device Emulators](#) in [Appendix C, "Guest Configuration"](#) for information on using the e100 and e1000 controllers.

- **Hypervisor debugger:** Includes an optional kernel-level debugger for the Oracle VM Server hypervisor, allowing debugging of an entire host including all running guests. See the documentation installed with Oracle VM Server in the `/usr/share/doc/xen/README.kdb` file for more information.
Guest debugger: Includes an optional guest debugger which allows individual guests to be debugged using the standard gdb network protocol. Supports both paravirtualized and hardware virtualized guests. See the documentation installed with Oracle VM Server in the `/usr/share/doc/xen/README.gdbsx` file for more information.
- Enhanced file I/O.

New Features in Release 2.1.1

There are no new features in Oracle VM Server Release 2.1.1.

Introduction to Virtualization

This Chapter provides introductory information on virtualization. It discusses why you would want to use virtualization, the technology provided, and features of Oracle VM. It contains the following sections:

- [What is Virtualization?](#)
- [Why Virtualize?](#)
- [Xen™ Technology](#)
- [Oracle VM](#)

1.1 What is Virtualization?

Virtualization is the ability to run multiple *virtual* machines on a single piece of hardware. The hardware runs software which enables you to install multiple operating systems which are able to run simultaneously and independently, in their own secure environment, with minimal reduction in performance. Each virtual machine has its own virtual CPU, network interfaces, storage and operating system.

1.2 Why Virtualize?

With increased server provisioning in the datacenter, several factors play a role in stifling growth. Increased power and cooling costs, physical space constraints, man power and interconnection complexity all contribute significantly to the cost and feasibility of continued expansion.

Commodity hardware manufacturers have begun to address some of these concerns by shifting their design goals. Rather than focus solely on raw gigahertz performance, manufacturers have enhanced the feature sets of CPUs and chip sets to include lower wattage CPUs, multiple cores per CPU die, advanced power management, and a range of virtualization features. By employing appropriate software to enable these features, several advantages are realized:

- **Server Consolidation:** By combining workloads from a number of physical hosts into a single host, a reduction in servers can be achieved and a corresponding decrease in interconnect hardware. Traditionally, these workloads would need to be specially crafted, partially isolated and well behaved, but with new virtualization techniques none of these requirements are necessary.
- **Reduction of Complexity:** Infrastructure costs are massively reduced by removing the need for physical hardware, and networking. Instead of having a large number of physical computers, all networked together, consuming power and

administration costs, fewer computers can be used to achieve the same goal. Administration and physical setup is less time consuming and costly.

- **Isolation:** Virtual machines run in sand-boxed environments. Virtual machines cannot access the resources of other virtual machines. If one virtual machine performs poorly, or crashes, it does not affect any other virtual machine.
- **Platform Uniformity:** In a virtualized environment, a broad, heterogeneous array of hardware components is distilled into a uniform set of virtual devices presented to each guest operating system. This reduces the impact across the IT organization: from support, to documentation, to tools engineering.
- **Legacy Support:** With traditional bare-metal operating system installations, when the hardware vendor replaces a component of a system, the operating system vendor is required to make a corresponding change to enable the new hardware (for example, an Ethernet card). As an operating system ages, the operating system vendor may no longer provide hardware enabling updates. In a virtualized operating system, the hardware remains constant for as long as the virtual environment is in place, regardless of any changes occurring in the real hardware, including full replacement.

1.3 Xen™ Technology

The Xen hypervisor is a small, lightweight, software virtual machine monitor, for x86-compatible computers. The Xen hypervisor securely executes multiple virtual machines on one physical system. Each virtual machine has its own guest operating system with almost native performance. The Xen hypervisor was originally created by researchers at Cambridge University, and derived from work done on the Linux kernel.

The Xen hypervisor has been improved and included with Oracle VM Server.

1.4 Oracle VM

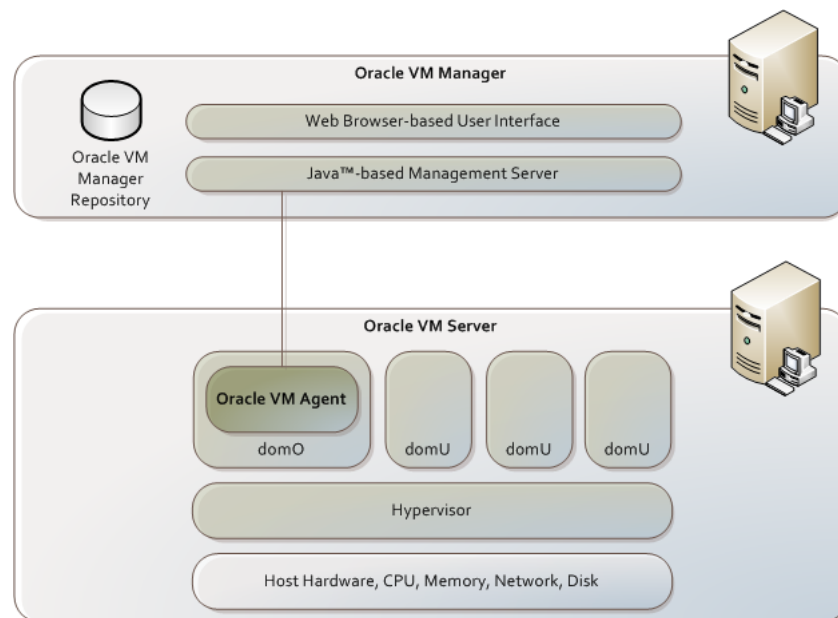
Oracle VM is a platform that provides a fully equipped environment for better leveraging the benefits of virtualization technology. Oracle VM enables you to deploy operating systems and application software within a supported virtualization environment. The components of Oracle VM are:

- **Oracle VM Manager:** Provides the user interface, which is a standard ADF (Application Development Framework) web application, to manage Oracle VM Servers, virtual machines, and resources. Use Oracle VM Manager to:
 - Create virtual machines
 - Create server pools
 - Power on and off virtual machines
 - Pause and unpause live virtual machines
 - Deploy virtual machines
 - Manage virtual NICs (Network Interface Cards), disks and shared disks
 - Create virtual machine templates from virtual machines
 - Import virtual machines and templates
 - Manage high availability of Oracle VM Servers, server pools, and guest virtual machines

- Perform live migration of virtual machines
- Import and manage ISOs
- **Oracle VM Server:** A self-contained virtualization environment designed to provide a lightweight, secure, server-based platform for running virtual machines. Oracle VM Server is based upon an updated version of the underlying Xen hypervisor technology, and includes Oracle VM Agent.
- **Oracle VM Agent:** Installed with Oracle VM Server. Oracle VM Manager communicates with Oracle VM Agent to manage the Oracle VM Servers and virtual machines running on it.

Figure 1–1, "Oracle VM Architecture" shows the components of Oracle VM.

Figure 1–1 Oracle VM Architecture



This book discusses Oracle VM Server, and the Oracle VM Agent. See the *Oracle VM Manager Installation Guide* and the *Oracle VM Manager User's Guide* for information on installing, and using Oracle VM Manager, and managing Oracle VM Servers.

Oracle VM Server

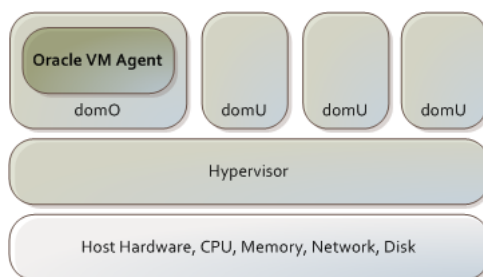
This Chapter contains an overview of Oracle VM Server and the underlying hypervisor, the components of virtual machines and domains, and gets you started with the tools to create and manage guests. This Chapter contains:

- [Oracle VM Server](#)
- [Hypervisor](#)
- [Domains, Guests and Virtual Machines](#)
- [Management Domain](#)
- [Domains](#)
- [Paravirtualization, Hardware Virtualization, and Binary Translation](#)
- [Creating Virtual Machines](#)
- [Managing Domains](#)

2.1 Oracle VM Server

Oracle VM Server includes an updated version of the underlying Xen™ hypervisor technology, and the Oracle VM Agent. It also includes a Linux kernel with support for a broad array of devices, file systems, and software RAID volume management. The Linux kernel is run as dom0 to manage one or more domU virtual machines, each of which could be Linux or Microsoft Windows. [Figure 2–1, "Oracle VM Server"](#) shows the components of Oracle VM Server.

Figure 2–1 Oracle VM Server



2.2 Hypervisor

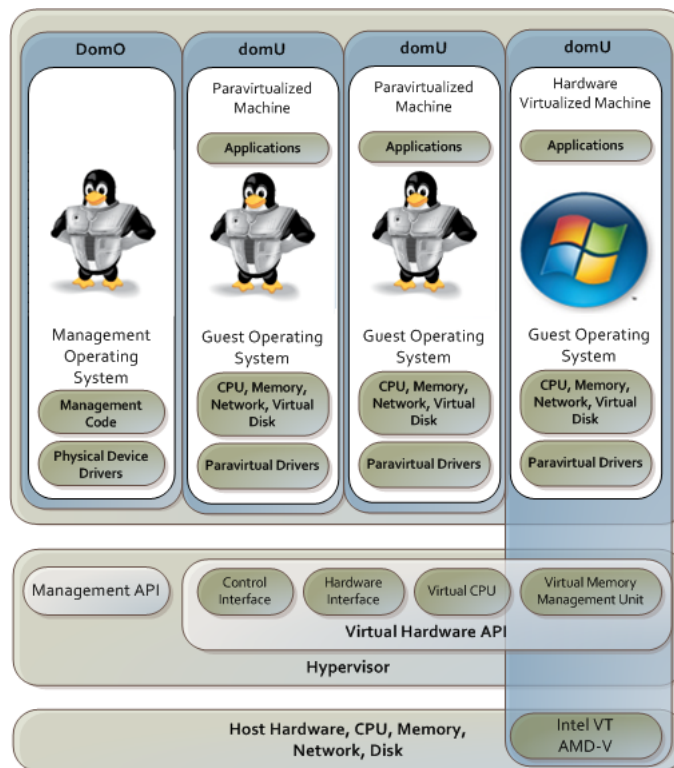
Oracle VM Server is designed so that the hypervisor (also called the Virtual Machine Monitor, or VMM) is the only fully privileged entity in the system, and has an extremely small footprint. It controls only the most basic resources of the system, including CPU and memory usage, privilege checks, and hardware interrupts.

2.3 Domains, Guests and Virtual Machines

The terms domain, guest and virtual machine are often used interchangeably, but they have subtle differences. A *domain* is a configurable set of resources, including memory, virtual CPUs, network devices and disk devices, in which virtual machines run. A domain is granted virtual resources and can be started, stopped and restarted independently. A *guest* is a virtualized operating system running within a domain. A guest operating system may be paravirtualized or hardware virtualized. Multiple guests can run on the same Oracle VM Server. A *virtual machine* is a guest operating system and its associated application software.

Oracle VM Server guest operating systems may run in one of two modes, paravirtualized or hardware virtualized. In paravirtualized mode, the kernel of the guest operating system is recompiled to be made aware of the virtual environment. This allows the paravirtualized guest to run at near native speed, since most memory, disk and network accesses are optimized for maximum performance.

Figure 2–2 Virtual Machine Architecture



If support for hardware virtualization is available (either Intel VT or AMD-V), the guest operating system may run completely unmodified. This hardware virtualized guest is carefully monitored and trapped by Oracle VM Server when any instruction is executed which would violate the isolation with other guests or dom0. In the current

implementation, there may be performance penalty for certain types of guests and access types, but hardware virtualization also allows many Microsoft Windows™ operating systems and legacy operating systems to run unmodified.

2.4 Management Domain

Most of the responsibility of hardware detection in a Oracle VM Server environment is passed to the management domain, referred to as domain zero (or dom0). The dom0 kernel is actually a complete Linux kernel with support for a broad array of devices, file systems, and software RAID and volume management. In Oracle VM Server, the dom0 is tasked with providing access to much of the system hardware, creating, destroying and controlling guest operating systems, and presenting those guests with a set of common virtual hardware.

2.5 Domains

Domains other than the management domain (dom0) are referred to as domU. These domains are unprivileged domains with no direct access to the hardware or device drivers. Each domU is started by Oracle VM Server in dom0.

2.6 Paravirtualization, Hardware Virtualization, and Binary Translation

Oracle VM Server uses paravirtualization, not binary translation. That is, the source code of the operating system is modified to support virtualization.

Binary translation is neither faster, nor slower, than hardware virtualization. Whether binary translation or hardware virtualization is more efficient than paravirtualization depends on the implementation of the binary translation and hardware virtualization, and the applications and operating system running as a guest on the system.

Binary translation and hardware virtualization, is required if you are using an operating system where it is impractical to do paravirtualization, for example, if the source code is not available such as for Microsoft Windows™, or the user base is not large enough to sustain a paravirtualization effort such as for the Linux 2.4.x kernel. In many situations, paravirtualization may perform better than binary translation as operations that cause a hypervisor interaction can be grouped and reused, rather than each event requiring its own hypervisor interaction.

2.7 Creating Virtual Machines

Create virtual machines (guests) using the Oracle VM Server virt-install command-line tool, or using a Virtual Machine Template in Oracle VM Manager. See [Chapter 4, "Creating a Guest Virtual Machine"](#) and the *Oracle VM Manager User's Guide* for more information.

2.8 Managing Domains

Manage domains using the Oracle VM Server xm command-line tool, or using Oracle VM Manager. See [Chapter 5, "Domain Monitoring and Administration"](#) and the *Oracle VM Manager User's Guide* for more information.

Migrate domains using the xm migrate command. See [Chapter 6, "Domain Live Migration"](#) for more information.

2.9 Configuring Oracle VM Server

You can configure Oracle VM Server using the configuration file. The configuration file options are available in the `/etc/xen/xend-config.sxp` file. When you make changes to this file, you must restart Oracle VM Server for the changes to take effect. See [Appendix B, "Oracle VM Server Configuration File"](#) for more information on the configuration options.

2.10 Managing Oracle VM Server Repositories

You can create new Oracle VM Server repositories for storage of ISOs, guest, and live migration. See [Chapter 9, "Managing Storage Repositories"](#) for more information.

Oracle VM Agent

Oracle VM Manager communicates with Oracle VM Agent to create and manage guests on an Oracle VM Server. Oracle VM Agent is installed and configured during the installation of Oracle VM Server. You do not need to install Oracle VM Agent separately. Oracle VM Agent is installed into:

`/opt/ovs-agent-2.3`

Oracle VM Agent logs are located in:

`/var/log/ovs-agent/`

This Chapter discusses the configuration and control of Oracle VM Agent. It contains:

- [Oracle VM Agent Command-Line Tool](#)
- [Configuring Oracle VM Agent](#)
- [Starting Oracle VM Agent](#)
- [Stopping Oracle VM Agent](#)
- [Monitoring Oracle VM Agent](#)

See [Appendix D, "Oracle VM Agent Architecture"](#) for more detailed information.

3.1 Oracle VM Agent Command-Line Tool

The `ovs-agent` command-line tool enables you to configure and control Oracle VM Agent. The following sections discuss using the `ovs-agent` command-line tool. See "[ovs-agent](#)" in [Appendix A, "Command-Line Tools"](#) for more details on the `ovs-agent` command-line tool options.

3.2 Configuring Oracle VM Agent

Oracle VM Agent is configured during installation. A default user is created with the username *admin*, and the password you set during installation.

You can change the default configuration with the Oracle VM Agent configuration script, `ovs-agent`. To configure Oracle VM Agent:

1. As the *root* user, run the Oracle VM Agent configuration script:

```
# service ovs-agent configure
```

2. You are prompted to enter the IP addresses that are allowed to access to the computer.

```
;network access control by ip --
```

```
;rules := if addr.match(allow) and not addr.match(deny): return True
;pattern items delimited by comma and could be
;10.1.1.1      #single ip
;10.1.1.*      #range
;10.1.1.1/24   #range in CIDR format
;default to allow all, deny none
allow=*
allow=
```

Enter * (asterisk) to allow all IP addresses access to the computer. Alternatively, enter a list of IP addresses for computers for which you want to allow access, while restricting all others. The default is to allow all computers to access the computer (*). For example, to allow access to all IP addresses in the domain 192.168.2.x, enter

```
192.168.2.*
```

Press **Enter**.

3. You are prompted to enter the IP addresses that are denied access to the computer.

```
now allow=192.168.2.*
```

```
deny=
deny=
```

Leave this field empty (does not deny any IP addresses), or enter * to deny all IP addresses access to the computer. Alternatively, enter a list of IP addresses for which you want to deny access to the computer, while allowing all others. The default is to deny no computers access to the computer.

Press **Enter**.

4. You are prompted to enter the shared disk search path.

```
now deny=
```

```
;share_disk_pat --
;set the directories for searching sharable block devices
;directories should be seperated by ':'
;if not set, /dev/mpath/* will be used
share_disk_pat=/dev/mpath/*
share_disk_pat=
```

Enter the shared disk search path. You should not use the path /dev/mapper/x for multipath devices, only use the path /dev/mpath/x.

Press **Enter**.

5. You are prompted whether you want to change the Oracle VM Agent password.

```
would you like to modify password to communicate with agent?[y/N]
```

Enter y to change the Oracle VM Agent password, or N to continue without changing the password. Press **Enter**.

6. Restart Oracle VM Agent for the configuration changes to take effect.

```
# service ovs-agent restart
```


3.3 Starting Oracle VM Agent

Oracle VM Agent is started automatically when the computer starts. To manually start Oracle VM Agent, enter

```
# service ovs-agent start
```

Oracle VM Agent is started.

Alternatively, if Oracle VM Agent is already running, you can stop and restart it with the command

```
# service ovs-agent restart
```

Oracle VM Agent is stopped, and restarted.

3.4 Stopping Oracle VM Agent

To stop Oracle VM Agent, enter

```
# service ovs-agent stop
```

Oracle VM Agent is stopped.

Note: When Oracle VM Agent is stopped, Oracle VM Manager cannot manage the Oracle VM Server or the guests running on it.

If you shut down or restart the Oracle VM Agent on an HA-enabled Oracle VM Server on which guests are running you are prompted to:

- Migrate or Power Off the guest(s) using Oracle VM Manager. When the guests have been migrated or Powered Off, the Oracle VM Agent shuts down.
- Shut down the guest(s) and shut down Oracle VM Agent.
- Cancel the shutdown operation.

3.5 Monitoring Oracle VM Agent

To get information on the Oracle VM Agent daemon, enter

```
# service ovs-agent status
```

Information on the Oracle VM Agent daemon is displayed.

Creating a Guest Virtual Machine

This Chapter contains information on how to create a guest virtual machine using Oracle VM Server. You can create paravirtualized guests and hardware virtualized guests using a template, or using the command-line tool `virt-install`. The `virt-install` tool can be used as an interactive shell, or all parameters can be given at the same time from the command-line. You can enter multiple parameters to the `virt-install` tool in the format:

`virt-install [options]`

[Appendix A, "Command-Line Tools"](#) lists the `virt-install` command-line tool parameters.

You can also create guest virtual machines using Oracle VM Manager, and is the recommended method for creating guests. See the *Oracle VM Manager User's Guide* for information on creating guests with Oracle VM Manager.

Oracle recommends you create paravirtualized guests if possible, as the performance of a paravirtualized guest is superior to that of a hardware virtualized guest.

Before you create a guest virtual machine, you should have access to an installation tree, or a template. You may also need a host IP address, and a hostname.

This Chapter contains:

- [Supported Guest Operating Systems](#)
- [Creating an Installation Tree](#)
- [Creating a Guest Using a Template](#)
- [Creating a Guest Using `virt-install`](#)
- [Creating a Paravirtualized Guest Manually](#)
- [Creating a Hardware Virtualized Guest Manually](#)
- [Converting a Hardware Virtualized Guest to a Paravirtualized Guest](#)
- [Installing Paravirtual Drivers](#)

4.1 Supported Guest Operating Systems

See the *Oracle VM Server Release Notes* for a list of the supported guest operating systems you can create.

4.2 Creating an Installation Tree

You cannot create a paravirtualized guest virtual machine from a local hard disk or CD-ROM using either the virt-install command-line tool, or Oracle VM Manager. You can, however, create an installation tree, and mount it as an NFS share. or make it available via HTTP or FTP. For example, to create mount an ISO file and make it available via NFS:

```
# mkdir -p /el/EL5-x86
# mount -o ro,loop /path/to/Enterprise-R5-x86-dvd.iso /el/EL5-x86
# exportfs */el/EL5-x86/
```

When you create the guest virtual machine, enter the installation location as:

```
nfs:example.com:/el/EL5-x86
```

Similarly, to set up an installation tree that can be accessed via HTTP on a computer named example.com, enter

```
# cd /var/www/html
# mkdir EL5-x86
# mount -o ro,loop /path/to/Enterprise-R5-x86-dvd.iso EL5-x86
```

When you create the guest virtual machine, enter the installation location as:

```
http://example.com/EL5-x86
```

Note: If you have multiple ISO files (CDs), you can mount each ISO file (CD) and copy the contents into a single directory. All the ISO files are then available from the same location.

4.3 Creating a Guest Using a Template

You can create a guest using a template. You can also register a template in Oracle VM Manager and use it to create guests. See the *Oracle VM Manager User's Guide* for information about using templates with Oracle VM Manager.

A template is compressed as a .tgz file. A template must contain the basic guest configuration files, vm.cfg and system.img. Templates are often hosted on an FTP or HTTP server. The following example shows you how to download an Oracle Linux template from an HTTP server, and use it to create a guest.

To create a guest virtual machine from a template:

1. Log in to the Oracle VM Server as the *root* user.
2. Download the Oracle VM template .zip file to an Oracle VM Server and place it in the /OVS/seed_pool directory.
3. Use the unzip tool to uncompress the file. The unzip tool can be downloaded from:

<http://oss.oracle.com/el4/unzip/unzip.html>

4. Extract the .tgz file. This step creates a directory with the name of the template. This directory contains the files for the template. For example:

```
# cd /OVS/seed_pool
# tar xzf OVM_template_name.tgz
```

This creates the following directory structure:

```

/OVS/seed_pool/OVM_template_name
|
|- System.img           (operating system image file)
|- extra.img            (optional additional image file(s))
|- vm.cfg               (virtual machine configuration file)
|- README

```

5. Change directory to where the virtual machine files are located

```
# cd /OVS/seed_pool/OVM_template_name
```

6. Run the following commands to generate a new MAC address:

```
# export PYTHONPATH=/opt/ovs-agent-2.3; python -c "from OVSCommons import
randomMAC; print randomMAC()"
```

7. Edit the vm.cfg file and change the line starting with vif to:

```
vif = [ 'mac=00:16:3E:xx:xx:xx', ] # for PVM
```

or

```
vif = [ 'type=netfront, mac=00:16:3E:xx:xx:xx', ] # for PVHVM
```

Where 00:16:3E:xx:xx:xx is the MAC address generated in 6.

8. Create and start up the guest virtual machine from the Oracle VM Server command line:

```
# xm create vm.cfg
```

9. Connect to the guest virtual machine's console using VNC:

a. Find out the VNC port number by running the following:

```
# xm list -l OVM_template_name |grep location
```

This displays the port number assigned to the guest virtual machine's VNC console. There may be more than one line, but there should be a line that looks similar to:

```
(location 0.0.0.0:5901)
```

Use the port number (5901 in this example) when connecting to the guest virtual machine using VNC Viewer.

b. Connect to the guest virtual machine using any VNC Viewer from another computer. For example, on Oracle Linux, use the command:

```
# vncviewer hostname:vnc_port
```

Where *hostname* is the IP address or host name of the Oracle VM Server, and *vnc_port* is the port number found out in a.

c. Complete any guest virtual machine configuration required by the template.

The guest is created and started.

4.4 Creating a Guest Using virt-install

The following example shows how to create a paravirtualized or hardware virtualized guest using the virt-install command-line tool. This procedure uses an interactive

session. You can also pass virt-install parameters at the same time as command-line options. In particular, kickstart options can be passed with:

```
virt-install -x ks=options
```

To create a paravirtualized or hardware virtualized guest interactively:

1. Open a command-line shell as *root*, and start the interactive install process by running the virt-install command-line tool:

```
# virt-install
```

2. If the host is capable of creating a hardware virtualized guest, the following question is displayed:

```
Would you like a fully virtualized guest (yes or no)?
```

Creating a hardware (fully) virtualized guest enables you to run unmodified operating systems, such as Microsoft Windows. Enter *no* to create a paravirtualized guest, or enter *yes* to create a hardware virtualized guest. Press **Enter**.

3. The following question is displayed:

```
What is the name of your virtual machine?
```

This is the label that identifies the guest. It is used as the guest's configuration file name and stored as */etc/xen/name*. This label is also used with a number of *xm* commands. Enter the name of the guest, for example enter

```
myguest
```

The configuration file is created. Press **Enter**.

4. The following question is displayed:

```
How much RAM should be allocated (in megabytes)? Setting the RAM to a value less than 256 megabytes is not recommended.
```

You are prompted to enter the RAM size to be allocated to the guest. RAM is allocated solely to the guest, and not taken from dom0. To check the amount of RAM available on your computer, run the *xm info* command and review the *free_memory* column. Free memory is displayed in Megabytes. This is the total amount of RAM that can be allocated to guests. Enter the amount of RAM to be allocated for the guest in Megabytes, for example, enter

```
1024
```

Press **Enter**.

5. The following question is displayed:

```
What would you like to use as the disk (path)?
```

The guest sees the disk storage allocated in virt-install as a single virtual hard disk, analogous to a physical hard disk. This appears as *hda* and can be partitioned and managed in the guest exactly as if it were regular physical hardware. Enter the absolute local path and file name of the file to serve as the disk image for the guest, for example, enter

```
/OVS/running_pool/myguest/System.img
```

This is exported as a full disk to your guest. Press **Enter**.

6. If the file specified in the previous step does not exist, the following question is displayed:

How large would you like the disk to be (in gigabytes)?

Enter the size of the virtual disk for the guest in Gigabytes. For the purpose of this example, enter 8 Gigabytes. For example, enter

8

Press **Enter**.

7. The following question is displayed:

Would you like to enable graphics support (yes or no)?

Graphics support determines whether a virtual graphics card is available to the guest. If you are creating a hardware virtualized guest, you should always answer yes to this question. If you are creating a paravirtualized guest, you can answer yes, or no. Press **Enter**.

8. The virt-install utility requests different install locations for paravirtualized and hardware virtualized guests. Paravirtualized guests can be installed from an installation tree using NFS, FTP, and HTTP. Hardware virtualized guests can be installed from an ISO file on dom0.

- If you are creating a paravirtualized guest, the following question is displayed:

What is the install location?

This is the path to an Oracle Linux installation tree, in the format used by Anaconda. NFS, FTP, and HTTP locations are supported. For example:

```
nfs:example.com:/path/to/tree/  
http://example.com/path/to/tree/  
ftp://example.com/path/to/tree/
```

Enter the path to an installation tree, for example

```
http://example.com/EL5-x86
```

Press **Enter**.

Note: The installation location must be a location on the network. It is not possible to perform the installation from a local disk or CD-ROM. See [Section 4.2, "Creating an Installation Tree"](#).

- If you are creating a hardware virtualized guest, the following question is displayed:

What would you like to use for the virtual CD image?

This is the path to an ISO file on the file system on dom0. For example:

```
/mypath/myISO.iso
```

Press **Enter**.

Note: NFS, FTP and HTTP locations are not supported. Do not mount the ISO file, virt-install will mount the ISO file and begin the installation from it.

The guest operating system installer starts. If you enabled graphics support in 7, a VNC window is displayed and the graphical installer is displayed. If you did not enable graphics support, a text-based installer is displayed. For example, a text-based installation of Oracle Linux displays:

Figure 4–1 Text-based Installer Screen

```

Welcome to Enterprise Linux

+-----+ Choose a Language +-----+
|
| What language would you like to use
| during the installation process?
|
| Catalan ^
| Chinese(Simplified) :
| Chinese(Traditional) #
| Croatian :
| Czech :
| Danish :
| Dutch :
| English v
|
| +----+
| | OK |
| +----+
|
+-----+

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

```

Follow the prompts to complete the guest operating system installation.

4.5 Creating a Paravirtualized Guest Manually

To manually create a paravirtualized guest:

1. Create the root file system.
2. Populate the root file system.
3. Configure the guest.

4.5.1 Creating the Root File System

To create the root file system:

1. Create a root partition for the guest. The root partition may be a:
 - Physical partition
 - Logical Volume Manager-backed Virtual Block Device
 - File-backed Virtual Block Device

Select one of the following options to create the root file system.

a. Using a physical disk partition

Create a disk partition for the guest root.

Make a file system on the partition.

b. Using a Logical Volume Manager-backed Virtual Block Device

A particularly appealing solution is to use a Logical Volume Manager (LVM) volume as backing for a guest file system, as this allows dynamic growing and shrinking of volumes, snapshots, and other features.

To initialize a partition to support LVM volumes, enter

```
# pvcreate /dev/sda10
```

Create a volume group named **vg** on the physical partition:

```
# vgcreate vg /dev/sda10
```

Create a logical volume of 4 Gigabytes named **myvm disk1**:

```
# lvcreate -L4096M -n myvm disk1 vg
```

You now have a `/dev/vg/myvm disk1`. Make a file system on the partition:

```
# mkfs -t ext3 /dev/vg/myvm disk1
```

c. Using a file-backed Virtual Block Device

To create a 4 Gigabyte file-backed virtual block device, enter

```
# dd if=/dev/zero of=vm1disk bs=1k seek=4096k count=1
```

Make a file system in the disk file:

```
# mkfs -t ext3 vm1disk
```

The tool requests that you confirm the creation of the file system. Enter **y** to confirm the creation of the file system.

4.5.2 Populating the Root File System

The root file system for the guest may be populated in a number of ways:

- Copying the root file system of dom0
 - Installing an operating system
1. To copy the root file system of dom0, mount the guest root partition to `/mnt`:

```
# mount -t <File system type> <Guest Root Partition> /mnt
```

Copy the root file system from dom0 to domU:

```
# rsync -avH /boot /mnt
# rsync -avH /root /mnt
# rsync -avH /dev /mnt
# rsync -avH /var /mnt
# rsync -avH /etc /mnt
# rsync -avH /usr /mnt
# rsync -avH /bin /mnt
# rsync -avH /sbin /mnt
# rsync -avH /lib /mnt
```

If your computer is a 64 bit computer, enter

```
# rsync -avH /lib64 /mnt
```

Then continue for all computers:

```
# rsync -avH /selinux /mnt
# mkdir /mnt/{proc,sys,home,tmp}
# chmod 777 /mnt/tmp
# umount /mnt
```

2. Install an operating system. This may be done a number of ways.
 - Install an Oracle VM Server-enabled operating system from CD-ROMs.
 - Install an Oracle VM Server-enabled operating system from a network drive, or PXE (Preboot Execution Environment) install.

After you create the root file system for the guest, modify the guest configuration files to reflect its configuration. For example, update `/etc/hosts`, `/etc/fstab` and any network configuration files.

4.5.3 Configuring the Guest

Modify the following guest configuration files to configure the guest:

1. Edit `/mnt/etc/fstab` to reflect the mounted file system in the guest.

```
/dev/sda1 / ext3 defaults 1 1
none /dev/pts devpts gid=5,mode=620 0 0
none /dev/shm tmpfs defaults 0 0
none /proc proc defaults 0 0
none /sys sysfs defaults 0 0
```

`/dev/hda1` is the root of domU as set up in the configuration file.

2. Edit `/mnt/etc/sysconfig/network` to include a valid host name.

GATEWAY is the same value as dom0.

Hostname is the name of the virtual machine, for example, `mycomputer.example.com`. Make sure the name you use is unique and not being used by another machine.

```
NETWORKING=yes
HOSTNAME=mycomputer.example.com
GATEWAY=139.185.48.1
```

3. Edit the `/mnt/etc/hosts` file to include the IP address and hostname. Make sure the IP address you use is unique and not being used by another computer.

```
127.0.0.1 localhost.localdomain localhost
139.185.48.212 mycomputer.example.com hostname
```

4. Edit the `/mnt/etc/sysconfig/network-scripts/ifcfg-eth0` file. Use the same MAC address as the vif. If more than one MAC address is exported to the guest operating system, you must to configure more network interfaces, for example, `eth1`, `eth2` in dom0.

The NETMASK and BROADCAST address must match the corresponding network interface in dom0.

HWADDR is same as the MAC address vif.

IPADDR is the same as in the `/mnt/etc/hosts` file.

```
DEVICE=eth0
BOOTPROTO=static
```

```
HWADDR=00:50:56:02:ff:d3
IPADDR=10.1.1.1
NETMASK=255.255.254.0
BROADCAST=10.1.1.255
ONBOOT=yes
TYPE=Ethernet
```

5. Move `/lib/tls` to `/lib/tls.disabled`.

```
# mv /mnt/lib/tls /mnt/lib/tls.disabled
```

6. Unmount `/mnt`.

```
# umount /mnt
```

7. Create the guest.

```
# xm create -c /etc/xen/domain-config-file
```

8. Get the console of the guest.

```
#xm console [Domainname|DomainID]
```

The guest virtual machine is created and running.

4.6 Creating a Hardware Virtualized Guest Manually

To create a hardware virtualized guest manually:

1. Install the operating system on a disk by CD-ROM pack or network install method (PXE install).
2. Create the guest configuration file, `/etc/xen/domain.cfg`. This is the minimum (without advanced options) hardware virtualized guest configuration file. Modify this file to suit your configuration.

```
#Config File for Full virtualization
import os, re
arch = os.uname()[4]
if re.search('64', arch):
    arch_libdir = 'lib64'
else:
    arch_libdir = 'lib'
# Kernel for hvm domain will be hvmloader
kernel="/usr/lib/xen/boot/hvmloader"
builder='hvm'
# Memory in MB for HVM guest domU
memory=3000
# Name of domain
name="hvm-dom"
# No of virtual cpus
vcpus=4
# Mac address and corresponding bridge
vif=[ 'mac=00:50:56:1e:34:b5 , bridge=xenbr0' ]
# Disk in which Guest OS is installed
disk=[ 'phy:/dev/cciss/c0d1,hda,w' ]
# Here /dev/cciss/c0d1 is the disk onwhich OS is installed.
device_model='/usr/' + arch_libdir + '/xen/bin/qemu-dm'
# Enable vnc library
sdl=0
vnc=1
# Vncviewer no is 1
```

```
vncviewer=1
# Password to access the vnc for this guest
vncpasswd="welcome"
vnclisten="0.0.0.0"
ne2000=1
serial='pty'
# Enable USB
usb=1
usbdevice='mouse'
```

See [Appendix C, "Guest Configuration"](#) for a more configuration file examples.

3. Mount the guest root file system to /mnt to enable you to modify the configuration files.
4. Edit /mnt/etc/sysconfig/network to specify the host name.

GATEWAY is same as dom0.

Hostname is the name of the virtual machine, for example, mycomputer.example.com. Make sure the name you use is unique and not being used by another computer.

```
NETWORKING=yes
HOSTNAME=mycomputer.example.com
GATEWAY=10.1.1.1
```

5. Edit the /mnt/etc/hosts file to include the hostname and IP address. Make sure that the IP address you use is unique and not being used by another machine.

```
127.0.0.1 localhost.localdomain localhost
10.1.1.1 mycomputer.example.com hostname
```

6. Edit /mnt/etc/sysconfig/network-scripts/ifcfg-eth0.

Use the same MAC address as you use for the vif. If more than one MAC address is exported to the guest operating system, you must configure more network interfaces, for example, eth1, eth2.

The NETMASK and BROADCAST address must match the corresponding network interface in dom0.

HWADDR is same as the MAC address in vif.

IPADDR is as in the /mnt/etc/hosts file.

```
DEVICE=eth0
BOOTPROTO=static
HWADDR=00:50:56:02:ff:d3
IPADDR=10.1.1.1
NETMASK=255.255.254.0
BROADCAST=10.1.1.255
ONBOOT=yes
TYPE=Ethernet
```

The /etc/fstab file does not require modification with a hardware virtualized guest. Hardware virtualized guests boot as a normal operating system. It reads the partition table as usual at boot time.

7. Unmount /mnt.

```
# umount /mnt
```

8. Create the guest.

```
# xm create -c /etc/xen/domain-config-file
```

9. Use VNCViewer to display the guest.

```
# vncviewer hostname_of_dom0
# password : welcome
```

The guest is displayed.

4.7 Converting a Hardware Virtualized Guest to a Paravirtualized Guest

You may want to convert a hardware virtualized guest to a paravirtualized guest. This example uses Oracle Enterprise Linux 4 Update 5 as it does not support a direct installation as a paravirtualized guest, and paravirtual drivers are available for this operating system. This procedure gives an example of installing Oracle Enterprise Linux as a hardware virtualized guest, then converting it to a paravirtualized guest.

To create an Oracle Enterprise Linux 4, Update 5, paravirtualized guest:

1. Copy the Oracle Enterprise-R4-U5-x86_64-dvd.iso image to the local file system of the Oracle VM Server computer

```
# ls -l /root/Enterprise-R4-U5-x86_64-dvd.iso
-rw-r--r-- 1 root root 2530611200 Aug  2 13:03 /root/Enterprise-R4-U5-x86_
64-dvd.iso
```

2. Create a logical volume, which will be used as the guest disk image.

```
# lvcreate -L8G -n el4u5 VolGroup00
```

3. Run the virt-install command-line tool to create a hardware virtualized machine domU, and install the Oracle Enterprise Linux operating system.

```
# virt-install -n el4u5 -f /dev/VolGroup00/el4u5 -v -c
/root/Enterprise-R4-U5-x86_64-dvd.iso -r 512 --vnc
```

If you see this error message:

```
main: unable to connect to host: Connection refused (111)
```

You must run VNCViewer to reconnect to the guest console:

```
# vncviewer :0
```

4. The Oracle Enterprise Linux install begins. From the installation choices, select the following:

Installation type: Server

Package selection: Default.

Partition layout type: Make sure the guest has a single root partition. Do not configure the virtual disk using LVM. Do not create a swap partition, or other partitions mounted at other locations such as /usr or /boot.

Firewall: Disable.

SELinux: Disable.

Network: Configure the network settings for either DHCP or a fixed IP address.

5. When the Oracle Enterprise Linux installation is complete, restart the guest. If the guest does not restart automatically, use the xm command-line tool to restart it, for example:

```
# xm list
Name      ID    Mem VCPUs      State   Time(s)
Domain-0  0     944    2    r-----  5670.8
# xm create el4u5
```

```
Using config file "/etc/xen/el4u5".
Started domain el4u5
# vncviewer :0
```

6. Take note of the guest IP address, or hostname if assigned via DHCP.

Copy the domU kernel for Oracle Enterprise Linux 4 Update 5, which can be found in the directory `extra_kernels/EL4U5PV_64` on the Oracle VM Server installation CD-ROM, to the guest:

```
# cd extra_kernels/EL4U5PV_64/
# scp kernel-xenU-version.EL.x86_64.rpm 10.1.1.1:
```

7. Log in to the guest as root and replace the contents of the `/etc/modprobe.conf` file with:

```
alias scsi_hostadapter xenblk
alias eth0 xennet
```

8. Install the kernel-xenU RPM:

```
# rpm -ivh kernel-xenU-version.EL.x86_64.rpm
warning: kernel-xenU-version.EL.x86_64.rpm: V3 DSA signature: NOKEY, key ID
b38a8516
Preparing... ##### [100%]
 1:kernel-xenU ##### [100%]
WARNING: No module xenblk found for kernel version.ELxenU, continuing anyway
```

9. Edit the `/boot/grub/grub.conf` file in the guest and change the default to point to this entry:

```
title Enterprise Linux Enterprise Linux AS (version.ELxenU)
    root (hd0,0)
    kernel /boot/vmlinuz-version.ELxenU ro root=LABEL=/
    initrd /boot/initrd-version.ELxenU.img
```

10. Shut down the guest. Modify the host configuration file `/etc/xen/el4u5` to an entry similar to:

```
name = "el4u5"
memory = "512"
disk = [ 'phy:/dev/VolGroup00/el4u5,hda,w', ]
bootloader="/usr/bin/pygrub"
vcpus=1
on_reboot = 'restart'
on_crash = 'restart'
```

11. Restart the guest, using the `xm` command-line tool:

```
# xm create -c el4u5
```

The hardware configuration is displayed.

12. Remove the configuration for the network adapter and keyboard.

13. Log in to the guest, and delete the `/etc/sysconfig/hwconf` file.

Shut down the guest.

14. Modify the guest configuration file `/etc/xen/el4u5` to add a `vif` entry similar to:

```
name = "el4u5"
memory = "512"
disk = [ 'phy:/dev/VolGroup00/el4u5,hda,w', ]
vif = [ 'bridge=xenbr0', ]
bootloader="/usr/bin/pygrub"
vcpus=1
on_reboot = 'restart'
on_crash = 'restart'
```

15. Start the guest and log in as root. Run the command:

```
# ifconfig eth0
```

Take note of the `HWaddr` (MAC address).

16. Replace the contents of the `/etc/sysconfig/network-scripts/ifcfg-eth0` file with:

```
TYPE=Ethernet
DEVICE=eth0
BOOTPROTO=dhcp
ONBOOT=yes
HWADDR=xx:xx:xx:xx:xx:xx
```

Replace `xx:xx:xx:xx:xx:xx` with the actual MAC address reported by the `ifconfig` command in the guest.

17. On the host, edit the domU configuration file `/etc/xen/el4u5`, and add the MAC address to the `vif` entry:

```
vif = [ 'mac=xx:xx:xx:xx:xx:xx, bridge=xenbr0', ]
```

Replace `xx:xx:xx:xx:xx:xx` with the actual MAC address reported by the `ifconfig` command in the guest.

18. Start the guest with the `xm create` command, for example

```
# xm create /etc/xen/el4u5
```

The guest is now ready for use.

4.8 Installing Paravirtual Drivers

For optimized performance, you can install paravirtualized drivers on hardware virtualized guests. Paravirtual drivers are optimized and improve the performance of the operating system in a guest virtual machine.

Installing Oracle Linux Release 4 Update 4 as a hardware virtualized guest may require that you install paravirtual drivers for your hardware. This section lists the steps for installing these paravirtual drivers.

To install the paravirtual drivers for Windows operating systems, see the *Oracle VM Windows Paravirtual Drivers Installation Guide*.

To install paravirtual drivers for Oracle Linux guest operating systems:

1. Download and install the paravirtual drivers from the Oracle Unbreakable Linux Network (ULN):

<http://linux.oracle.com/>

Search for package name `kmod-xenpv-kernel_type`. For example, if you are running the `hugemem` kernel in the guest, install the drivers with:

```
# up2date kmod-xenpv-hugemem
```

2. Modify the `/etc/modprobe.conf` file to comment out existing `eth0` line and add the following lines:

```
alias scsi_hostadapter xen_vbd
alias eth0 xen_vnif
```

3. Run `depmod`.
4. Edit the `/etc/xen/vm.cfg` file to replace the `vif` entry to:

```
vif = [ '', ] # for PVM
```

or

```
vif = [ 'type=netfront, ', ] # for PVHVM
```

5. Shut down the domain:

```
# xm shutdown mydomain
```

6. Start the domain:

```
# xm create /OVS/running_pool/myguest/vm.cfg
```

7. When prompted by `kudzu`, remove the old network configuration.

8. In the newly booted guest operating system, run the following command to find the new MAC address for `eth0`:

```
# ifconfig eth0
```

9. Edit the `/etc/xen/vm.cfg` file to add the new MAC address:

```
vif = [ 'mac=xx:xx:xx:xx:xx:xx, bridge=xenbr0', ] # for HVM
```

or

```
vif = [ 'type=netfront, mac=xx:xx:xx:xx:xx:xx, bridge=xenbr0', ] # for PVHVM
```

10. Create or edit the `/etc/sysconfig/network-scripts/ifcfg-eth0` file with the following contents:

```
TYPE=Ethernet
DEVICE=eth0
BOOTPROTO=dhcp
ONBOOT=yes
HWADDR=xx:xx:xx:xx:xx:xx
```

11. Create a new `initrd` image. Use the kernel version for your guest operating system.

```
# mkinitrd -f /boot/initrd-version.ELsmp.img version.ELsmp --omit-scsi-modules
--with=xen-vbd --with=xen-vnif --preload xen-vbd --preload xen-vnif
```

12. Reboot the domain.

13. Edit the `/boot/grub/grub.conf` file and add the following to the end of the *kernel* line.

```
kernel /vmlinuz-version.el5 ro root=LABEL=/ clock=pit nohpet nopmtimer
hda=noprobe hdb=noprobe ide0=noprobe
```

Domain Monitoring and Administration

This Chapter contains information on the Oracle VM Server domain lifecycle, monitoring and administration. It contains:

- [Domain Lifecycle](#)
- [Using the xm Command-Line Interface](#)

You can use Oracle VM Manager to monitor domains running on Oracle VM Server, or you can use the `xm` command. Using Oracle VM Manager is the recommended method of managing domains. See the *Oracle VM Manager User's Guide* for information on using Oracle VM Manager to manage domains (virtual machines).

5.1 Domain Lifecycle

There are a number of states in which a domain may exist. They are:

- Starting (initializing)
- Running
- Paused
- Suspended
- Stopping (shutting down)
- Powered off (stopped)

A **start** operation can take the domain from the stopped (powered down) state to the paused state, or the running state. From the running state, a **suspend** action takes the domain to the suspended state, and a **resume** operation takes it back to the running state. The transition to and from the suspended state could also happen from the paused state.

A domain in the running state could go to the paused state through the **pause** command, and return to the running state by the **resume** command. A domain in the running state could transition into the stopped state through a clean, or hard shut down.

5.2 Using the xm Command-Line Interface

You can create, destroy, manage and migrate domains using the `xm` command-line interface. You can enter parameters to the `xm` command-line tool in the format:

```
xm [option] [argument]
```

For example, to pause a domain called `mydomain`, enter

```
# xm pause mydomain
```

See "xm" in [Appendix A, "Command-Line Tools"](#) for detailed information on the xm command-line interface.

5.2.1 Monitoring Domains

The `xm top` command performs real time monitoring of domain loads on a host. The `xm top` command displays the following information:

- The state of each domain.
- The number of domains on the host.
- Memory statistics of the host, such as the total available memory, the memory in use, and free memory.
- The CPU statistics of the host, such as the number of CPUs and CPU speed.
- Information on each domain, such as domain name, domain state, CPU usage in seconds, percentage of CPU, memory in Kilobytes, and so on.

For example, an `xm top` command displays output similar to:

Figure 5–1 Example xm top Command Output

```
xentop - 19:31:55 Xen 3.1.1
4 domains: 1 running, 3 blocked, 0 paused, 0 crashed, 0 dying, 0 shutdown
Mem: 4193568k total, 4193492k used, 76k free CPUs: 4 @ 2992MHz
```

NAME	STATE	CPU(sec)	CPU(%)	MEM(k)	MEM(%)	MAXMEM(k)	MAXMEM(%)	VCPU
NETS	NETTX(k)	NETRX(k)	VBDS	UVD_00	UVD_RD	UVD_WR	SSID	
Domain-0	-----r	3647	0.7	414720	9.9	no limit	n/a	4
0	646023	3412987	0	0	0	0	0	
XEN_EL4U5_X86_PARA	--b---		6	0.0	524288	12.5	524288	12
.5	2	1	1	11	1	0	2568	1504
XEN_EL5_X86_64_HVM	--b---		108	3.8	2105216	50.2	2113536	50
.4	2	1	0	0	1	0	0	0
XEN_EL5_X86_PVM	--b---		124	0.1	1048576	25.0	1048576	25.0
1	1	16	748	1	0	32836	51768	0

```
Delay Networks VBds VCPUs Repeat header Sort order Quit _
```

Note that the format of each line of output wraps over two lines.

5.2.2 Viewing Host Information

Use the `xm info`, `xm log`, and `xm dmesg` commands to display information about the host computer. For example, the `xm info` command displays output similar to the following:

Figure 5–2 Example xm info Command Output

```

host          : ca-ostest224.us.oracle.com
release       : 2.6.18-8.1.6.0.15.el5xen
version       : #1 SMP Tue Oct 30 21:08:27 EDT 2007
machine       : i686
nr_cpus       : 4
nr_nodes      : 1
sockets_per_node : 2
cores_per_socket : 2
threads_per_core : 1
cpu_mhz       : 2992
hw_caps       : bfebfbff:20100800:00000000:00000140:0004e3bd:00000000:0
00000001
total_memory  : 4095
free_memory   : 0
xen_major     : 3
xen_minor     : 1
xen_extra     : .1
xen_caps      : xen-3.0-x86_64 xen-3.0-x86_32p hvm-3.0-x86_32 hvm-3.0-x
3.0-x86_32p hvm-3.0-x86_64
xen_scheduler  : credit
xen_pagesize   : 4096
platform_params : virt_start=0xff800000
xen_changeset  : unavailable
cc_compiler    : gcc version 4.1.1 20070105 (Red Hat 4.1.1-52)
--More--

```

Domain Live Migration

This Chapter discusses live migration of domains to other, identical computers. You must use identical computers to perform live migrations, that is, the computer make and model number must be identical.

To perform live migration of domains, you must attach the same shared storage at identical mount points to all hosts within the server pool before you can complete the migration. You can use OCFS2- or NFS-based external storage as shared storage, and create a storage repository for this purpose. This Chapter contains:

- [Creating Shared Storage](#)
- [Migrating a Domain](#)

6.1 Creating Shared Storage

If you want to perform live migration of domains to other, identical, computers, you must create external storage to be used as shared storage during the live migration, then create a storage repository. See [Section 8.1, "Preparing External Storage"](#) and [Section 8.2, "Preparing a Storage Repository"](#) for information on creating the shared storage and storage repository.

6.2 Migrating a Domain

You can use Oracle VM Manager to migrate a domain. See the *Oracle VM Manager User's Guide* for information on migrating domains using Oracle VM Manager. Alternatively, you can migrate a domain from the command line on an Oracle VM Server.

To migrate a domain from the Oracle VM Server command line, on the Oracle VM Server that contains the domain, migrate the domain to the remote computer with the following command:

```
# xm migrate mydomain myremotecomputer
```

The domain is migrated to the remote computer.

To perform live migration of the domain, use the command:

```
# xm migrate -l mydomain myremotecomputer
```

High Availability

This Chapter discusses implementing High Availability (HA) fail over for server pools and guests in Oracle VM. This Chapter contains:

- [High Availability \(HA\)](#)
- [Creating Shared Storage](#)
- [Enabling HA](#)

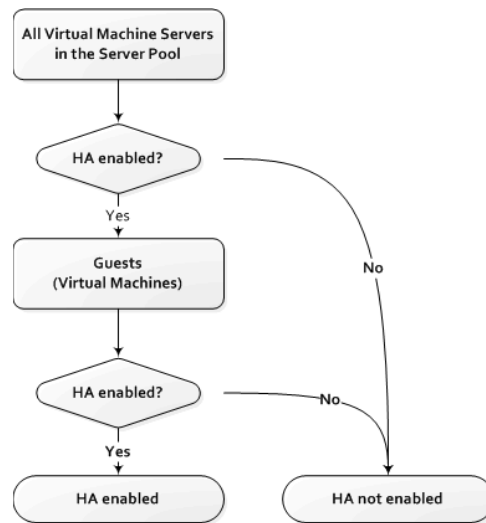
You can manage HA of server pools and guests using Oracle VM Manager. See the *Oracle VM Manager User's Guide* for information on managing HA using Oracle VM Manager.

7.1 High Availability (HA)

You can set up HA in Oracle VM to guarantee the availability of guests if the Virtual Machine Server they are running on fails or restarts. When a Virtual Machine Server is restarted or shut down, the guests running on it are either restarted on, or migrated to, another Virtual Machine Server.

You can manage HA with Oracle VM Manager. To implement HA, you must create a server pool (a cluster of Virtual Machine Servers) and have them managed by Oracle VM Manager. HA cannot be implemented with a stand-alone Oracle VM Server.

To use HA, you must first enable HA on the server pool, then on all guests, as shown in [Figure 7-1, "Enabling HA"](#). If you enable HA in the server pool and then for guests, when a Virtual Machine Server is shut down or fails, the guests are migrated or restarted on another available Virtual Machine Server. HA must be enabled for **both** the server pool **and** for guests. If HA is not enabled for both, HA is disabled.

Figure 7–1 Enabling HA

If HA is enabled, when you restart, shut down, or delete the Virtual Machine Server in Oracle VM Manager, you are prompted to migrate the running guests to another available Virtual Machine Server. If you do not migrate the running guests, Oracle VM Agent attempts to find an available Virtual Machine Server on which to restart the guests. The Virtual Machine Server is selected using the Preferred Server setting for the server pool when you create a guest in Oracle VM Manager:

- **Auto** selects an available Virtual Machine Server.
- **Manual** selects an available preferred Virtual Machine Server.

If you do not select a preferred Oracle VM Server when creating a guest in Oracle VM Manager, **Auto** is set as the default.

If there is no preferred Virtual Machine Server or Virtual Machine Server available, the guests shut down (Power Off) and are restarted when a Virtual Machine Server becomes available.

If the Server Pool Master fails, another Oracle VM Server is selected from the server pool to act as the Server Pool Master. The Oracle VM Server chosen to take over the Server Pool Master role is the first Oracle VM Server available to take the lock. To use the Server Pool Master fail over feature, you should make sure the Oracle VM Agent password is identical on all Oracle VM Servers in the server pool.

You can also dynamically change the Oracle VM Server which acts as the Server Pool Master without causing any outages.

If the Server Pool Master also performs the Utility Server role, and it fails, the Utility Server role is not moved to another Oracle VM Server. If you want fail over for the Utility Server role, make sure you set up more than one Utility Server in the server pool.

The possible HA scenarios are:

- If you shut down or restart a Virtual Machine Server in Oracle VM Manager, you are prompted which guests to migrate to another available Virtual Machine Server. Any guests which are not migrated, are restarted on an available Virtual Machine Server.

- If you shut down or restart a Virtual Machine Server at the Oracle VM Server command-line, Oracle VM Agent restarts the guests on an available Virtual Machine Server.
- If a Virtual Machine Server fails, all running guests are restarted automatically on another available Virtual Machine Server.
- If a Virtual Machine Server fails and no other Virtual Machine Servers are available, all running guests are restarted when a Virtual Machine Server becomes available.

Figure 7–2, "HA in effect for a Virtual Machine Server failure" shows a Virtual Machine Server failing and the guests restarting on other Virtual Machine Servers in the server pool.

Figure 7–2 HA in effect for a Virtual Machine Server failure

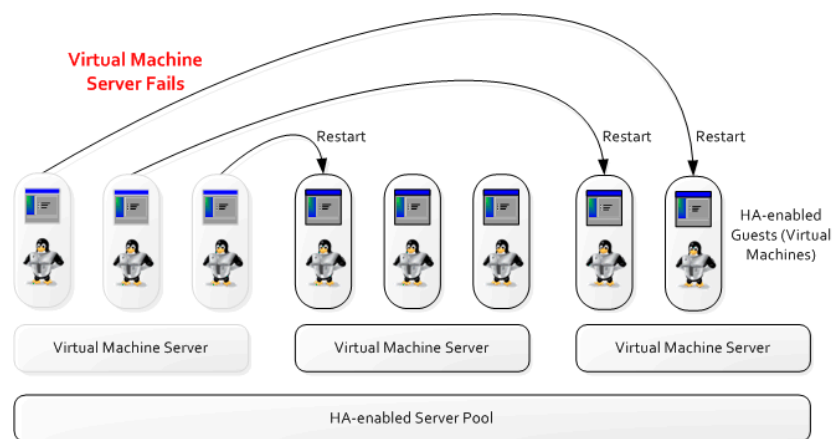
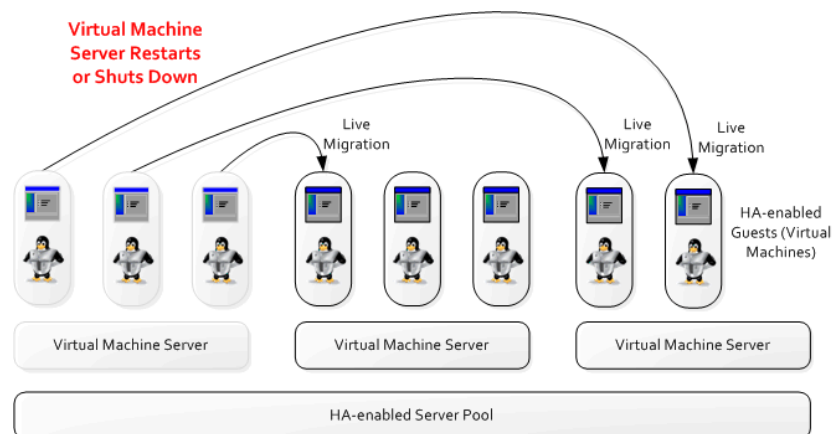


Figure 7–3, "HA in effect for a Virtual Machine Server restart or shut down" shows a Virtual Machine Server restarting or shutting down and the guests migrating to other Virtual Machine Servers in the server pool.

Figure 7–3 HA in effect for a Virtual Machine Server restart or shut down



To enable HA, you must first make sure all Virtual Machine Servers in the server pool:

- Use the same storage repository. This means all Oracle VM Servers in the server pool must have the same storage view. If the shared storage is OCFS2-based, the

OCFS2 device `/dev/sdd1` must be visible on all Virtual Machine Servers. If the shared storage is NFS-based, then all Virtual Machine Servers in the server pool must have client access to the same shared NFS source.

- Are in the same cluster.
- Are Oracle VM Server Release 2.1.2 or above.
- Have the cluster root configured from the Server Pool Master. The cluster root is specially designated shared storage used for *heartbeating* in the server pool (cluster). For example, the cluster root might be `example.com:/OVS` for an NFS cluster, or `/dev/sdd6` for an OCFS2 cluster. The heartbeat file would then be located at `example.com:/OVS/.server_pool_hb` for an NFS cluster. There is no heartbeat file for an OCFS2 cluster as one is in-built in the file system.
- Have the cluster root mounted at `/var/ovs/mount/root_sr_uuid` and linked to the `/OVS` directory. All other storage may be mounted at `/var/ovs/mount/other_sr_uuid`.

If you have created multiple storage repositories and set the cluster root from the Server Pool Master (see [Section 9.2.5, "Setting the Cluster Root"](#)), when you create a server pool from Oracle VM Manager, the cluster root storage repository is automatically mounted to `/var/ovs/mount/root_sr_uuid` and linked to the `/OVS` directory. Non-cluster root storage repositories are mounted to the `/var/ovs/mount/other_non_cluster_root_uuid/` directory. All storage repositories are automatically propagated and mounted to other nodes in the server pool (cluster).

- Have a shared cluster root and link to the `/OVS` directory (not local), using clustered OCFS2 on SAN, or ISCSI storage, or NFS on NAS. If the default local storage is a local OCFS2 partition, it cannot be HA-enabled.

The following sections describe how to perform the configuration required to enable HA.

7.2 Creating Shared Storage

If you want to enable HA, you must create external storage to be used as shared storage for the server pool, then create a storage repository. See [Section 8.1, "Preparing External Storage"](#) and [Section 8.2, "Preparing a Storage Repository"](#) for information on creating the shared storage and storage repository.

7.3 Enabling HA

After you have created the storage repository, you must enable HA in the server pool and on guests. To enable HA:

1. Log in to Oracle VM Manager and enable HA in the server pool.

If the server pool already exists, enable HA on the server pool. If the server pool does not exist, create the server pool and enable HA when you create it. See the *Oracle VM Manager User's Guide* for information on setting up HA using Oracle VM Manager.

The server pool set up is verified and any errors are displayed in Oracle VM Manager. If there are no errors, the Oracle VM Agent on the Server Pool Master automatically mounts the storage repository set as the cluster root, and any other storage repositories, on each Oracle VM Server in the server pool (the cluster). When an Oracle VM Agent is started in the server pool, this propagation is repeated for that Oracle VM Server.

The shared storage and cluster is configured, and the server pool is HA enabled.

- 2.** When creating guests (virtual machines) in Oracle VM Manager, enable HA.

HA is enabled on the server pool and guests.

Preparing External Storage and Storage Repositories

Oracle VM uses the concept of storage repositories to define where Oracle VM resources may reside in a server pool. Resources include guest virtual machines, virtual machines templates (guest seed images), ISO images, shared virtual disks, and so on.

A storage repository is required for High Availability (HA) fail over, and for live migration of domains.

A storage repository should be set up using external storage based on any of the following technologies:

- OCFS2 (Oracle Cluster File System) using the iSCSI (Internet SCSI) network protocol
- OCFS2 using SAN (Storage Area Network)
- NFS (Network File System)

To enable HA or live migration, you must make sure all Oracle VM Servers:

- Use the same storage repository
- Are in the same server pool

If you used the default /OVS partition during installation, and you have only one Oracle VM Server in your server pool, you do not need to create a storage repository. If you want to add more Oracle VM Servers to your server pool, you must create a storage repository and remove the local /OVS partition created during installation.

If you have only one Oracle VM Server in your server pool, but want to use iSCSI, SAN or NFS-based storage for your guest domains, you must create a storage repository, even though it will not be shared.

You prepare the external storage and storage repository at the Oracle VM Server command line, and then use Oracle VM Manager to create a server pool, and propagate the storage repository to all Oracle VM Servers in the server pool.

You can also manage the HA of server pools and guests, and live migration of domains, using Oracle VM Manager. See the *Oracle VM Manager User's Guide* for more information.

This Chapter discusses preparing external storage, and storage repositories to be used in Oracle VM Manager to create server pools, and contains:

- [Preparing External Storage](#)
- [Preparing a Storage Repository](#)

8.1 Preparing External Storage

If you want to enable HA, or perform live migration of domains to other, identical, computers, you must first create external storage, and then create a storage repository to be used in the server pool.

When you create a server pool in Oracle VM Manager, the external storage is automatically mounted on each Oracle VM Server in the server pool (the *cluster*). Similarly, when an Oracle VM Server in the server pool is restarted, the external storage is automatically mounted. Make sure the external storage is available to the Oracle VM Servers in the server pool so it can be automatically mounted when an Oracle VM Server is restarted.

This section discusses:

- [Preparing External Storage Using OCFS2 on iSCSI](#)
- [Preparing External Storage Using OCFS2 on SAN](#)
- [Preparing External Storage Using NFS](#)

When you have prepared the external storage, prepare the storage repository. See [Section 8.2, "Preparing a Storage Repository"](#) for information on preparing a storage repository.

8.1.1 Preparing External Storage Using OCFS2 on iSCSI

To prepare external storage to be used in a storage repository using OCFS2 on iSCSI:

1. Start the iSCSI service:

```
# service iscsi start
```

2. Run discovery on the iSCSI target. In this example, the target is 10.1.1.1:

```
# iscsiadm -m discovery -t sendtargets -p 10.1.1.1
```

This command returns output similar to:

```
10.1.1.1:3260,5 iqn.1992-04.com.emc:cx.apm00070202838.a2
10.1.1.1:3260,6 iqn.1992-04.com.emc:cx.apm00070202838.a3
10.2.1.250:3260,4 iqn.1992-04.com.emc:cx.apm00070202838.b1
10.1.0.249:3260,1 iqn.1992-04.com.emc:cx.apm00070202838.a0
10.1.1.249:3260,2 iqn.1992-04.com.emc:cx.apm00070202838.a1
10.2.0.250:3260,3 iqn.1992-04.com.emc:cx.apm00070202838.b0
```

3. Delete entries that you do not want to use, for example:

```
# iscsiadm -m node -p 10.2.0.250:3260,3 -T
iqn.1992-04.com.emc:cx.apm00070202838.b0 -o delete
# iscsiadm -m node -p 10.1.0.249:3260,1 -T
iqn.1992-04.com.emc:cx.apm00070202838.a0 -o delete
# iscsiadm -m node -p 10.2.1.250:3260,4 -T
iqn.1992-04.com.emc:cx.apm00070202838.b1 -o delete
# iscsiadm -m node -p 10.1.1.249:3260,2 -T
iqn.1992-04.com.emc:cx.apm00070202838.a1 -o delete
# iscsiadm -m node -p 10.0.1.249:3260,5 -T
iqn.1992-04.com.emc:cx.apm00070202838.a2 -o delete
```

4. Verify that only the iSCSI targets you want to use for the server pool are visible:

```
# iscsiadm -m node
```

5. Review the partitions by checking `/proc/partitions`:

```
# cat /proc/partitions
major minor #blocks name
 8      0  71687372 sda
 8      1   104391 sda1
 8      2  71577607 sda2
253     0  70516736 dm-0
253     1   1048576 dm-1
```

6. Restart the iSCSI service:

```
# service iscsi restart
```

7. Review the partitions by checking `/proc/partitions`. A new device is listed.

```
# cat /proc/partitions
major minor #blocks name
 8      0  71687372 sda
 8      1   104391 sda1
 8      2  71577607 sda2
253     0  70516736 dm-0
253     1   1048576 dm-1
 8     16  1048576 sdb
```

8. The new device can now be used. Use the `fdisk` utility to create at least one partition:

```
# fdisk /dev/sdb
```

Alternatively, if the external storage is more than 2 TB, use the `gparted` utility to create a BIOS boot partition (GPT):

```
# gparted /dev/sdb
```

9. Format the partition from any of the Oracle VM Servers in the (intended) server pool with a command similar to the following:

```
# mkfs.ocfs2 -Tdatafiles -N8 /dev/sdb1
```

The `-Tdatafiles` parameter makes `mkfs.ocfs2` use a large cluster size. The size chosen depends on the device size.

Add an additional parameter, `--fs-features=local`, if you are not mounting the volume in the server pool (cluster).

The `-N8` parameter allocates 8 slots to allow that many nodes to mount the file system concurrently. Increase the number if the server pool has (or will have) greater than 8 nodes. It is recommended that you have at least 8 slots, even for a local file system. Creating slots for a local file system allows users to easily cluster-enable the shared volume later.

For more information on using OCFS2, see the OCFS2 documentation online at:

<http://oss.oracle.com/projects/ocfs2/documentation/v1.4/>

<http://oss.oracle.com/projects/ocfs2-tools/documentation/v1.4/>

When you have prepared the external storage, prepare the storage repository. See [Section 8.2, "Preparing a Storage Repository"](#) for information on preparing a storage repository.

8.1.2 Preparing External Storage Using OCFS2 on SAN

To prepare external storage to be used in a storage repository using OCFS2 on SAN:

1. Review the partitions by checking `/proc/partitions`:

```
# cat /proc/partitions
major minor #blocks name
 8      0  71687372 sda
 8      1   104391 sda1
 8      2  71577607 sda2
253     0  70516736 dm-0
253     1   1048576 dm-1
 8     16   1048576 sdb
```

2. Determine the share disk volume you want to use. You may need to use the `fdisk` utility to create at least one partition on the shared disk volume:

```
# fdisk /dev/sdb
```

Alternatively, if the external storage is more than 2 TB, use the `gparted` utility to create a GPT partition:

```
# gparted /dev/sdb
```

3. Format the partition from any of the Oracle VM Servers in the (intended) server pool with a command similar to the following:

```
# mkfs.ocfs2 -Tdatafiles -N8 /dev/sdb1
```

The `-Tdatafiles` parameter makes `mkfs.ocfs2` use a large cluster size. The size chosen depends on the device size.

Add an additional parameter, `--fs-features=local`, if you are not mounting the volume in the server pool.

The `-N8` parameter allocates 8 slots to allow that many nodes to mount the file system concurrently. Increase the number if the server pool has (or will have) greater than 8 nodes. It is recommended that you have at least 8 slots, even for a local file system. Creating slots for a local file system allows users to easily cluster-enable the shared volume later.

For more information on using OCFS2, see the OCFS2 documentation online at:

<http://oss.oracle.com/projects/ocfs2/documentation/v1.4/>

<http://oss.oracle.com/projects/ocfs2-tools/documentation/v1.4/>

When you have prepared the external storage, prepare the storage repository. See [Section 8.2, "Preparing a Storage Repository"](#) for information preparing a storage repository.

8.1.3 Preparing External Storage Using NFS

To prepare external storage to be used in a storage repository using NFS, find an NFS mount point to use. This example uses the mount point:

```
mynfsserver:/vol/vol1/data/ovs
```

When you have identified an NFS mount point to use for the external storage, prepare the storage repository. See [Section 8.2, "Preparing a Storage Repository"](#) for information on preparing a storage repository.

8.2 Preparing a Storage Repository

In addition to preparing external storage for a server pool, you must also prepare a storage repository. The storage repository is used to host the resources used in the server pool, such as guest virtual machines and shared virtual disks.

The storage repository is used by Oracle VM Manager when you create a server pool and is automatically propagated to all Virtual Machine Servers in the server pool.

Creating a server pool ensures the safety of guest data, and protects from runaway nodes which may become unreachable. Oracle VM Servers in a server pool are in a cluster and have built-in rules and restrictions which are more stringent than unclustered Oracle VM Servers, for example, Quorum is needed and may require restarting one or more Oracle VM Servers to preserve the cluster under rare circumstances.

You can create a server pool using NFS- or OCFS2-based external storage. You must have at least one storage repository created and ready to use before you create a server pool. To prepare external storage for use in a storage repository, see [Section 8.1, "Preparing External Storage"](#).

Before you create a storage repository, make sure that every Oracle VM Server in the server pool has been correctly configured:

- The host name in the `/etc/hosts` file is associated with the public IP address, and not with 127.0.0.1. To make sure the hostname is correctly set in the `/etc/hosts` file, ping the host name of all Oracle VM Servers in the server pool to display their public IP addresses.
- You have all the Oracle VM Server entries in your DNS server. If you do not use DNS, make sure the correct setting is in the `/etc/hosts` file for all the Oracle VM Servers in the server pool. If you plan to use DNS for all Oracle VM Servers, but DNS was not specified during the Oracle VM Server installation, update the `/etc/resolv.conf` file and add your domain name to it.
- All the Oracle VM Servers in the server pool must use consistent name resolution, either using a DNS server, or using the `/etc/hosts` file. You should not have mixed name services for the Oracle VM Servers a server pool. For example, some Oracle VM Servers use DNS, while others use the `/etc/hosts` file to resolve host names.

To prepare a storage repository for use in Oracle VM Manager when a server pool is created:

1. On the Server Pool Master, create a storage repository with the script:

```
/opt/ovs-agent-2.3/utils/repos.py -n storage_location
```

For example, for an NFS set up, you might use something similar to:

```
# /opt/ovs-agent-2.3/utils/repos.py -n mynfsserver:/vol/vol1/data/ovs
```

Or for an OCFS2 using iSCSI or SAN set up, you might use something similar to:

```
# /opt/ovs-agent-2.3/utils/repos.py -n /dev/sdb1
```

2. The `repos.py -n` command from the previous step displays the UUID of the storage repository you created. You need this UUID to set the cluster root. Copy the UUID for the storage repository you want to use as the cluster root.
3. Paste the UUID for the storage repository and use it to set the cluster root with the command:

```
# /opt/ovs-agent-2.3/utils/repos.py -r UUID
```

For more detailed information on using the `repos.py` script, see [Section 9.2, "Managing Storage Repositories"](#).

The external storage and storage repository are configured and ready to use.

Log in to Oracle VM Manager and create a server pool. The storage repository is automatically propagated and the external storage mounted on all Virtual Machine Servers in the server pool.

Note: If you use Oracle VM Manager to create a server pool, you do not need to use the `repos.py -i` command to initialize the storage repository. Oracle VM Agent automatically mounts the shared storage on each Virtual Machine Server in the server pool.

Managing Storage Repositories

This Chapter contains information on managing Oracle VM Server storage repositories. It contains:

- [Storage Repository Directory Structure](#)
- [Managing Storage Repositories](#)

9.1 Storage Repository Directory Structure

When you create a server pool in Oracle VM Manager, the storage repository is automatically mounted with the source type, for example, NFS or OCFS2. The storage repository directory structure is also automatically created under the /OVS directory on the storage repository. These directories are listed in [Table 9–1, "/OVS Directory Contents"](#).

Table 9–1 /OVS Directory Contents

Directory Name	Description
iso_pool	Contains ISO files imported using Oracle VM Manager.
publish_pool	Contains guest virtual machines deployed as public.
seed_pool	Contains guest virtual machine templates.
sharedDisk	Contains shared virtual disks.
running_pool	Contains virtual machine images and configuration files.

The /OVS directory is the cluster root and is a symbolic link mounted to the /var/ovs/mount/*uuid* directory. For example, the mount command might display something similar to:

```
# mount
example.com:/OVS on /var/ovs/mount/F4135C096045458195057412169071E5 type nfs
(rw,addr=192.168.2.20)
```

And the ls command might display something similar to:

```
# ls -l /OVS
lrwxrwxrwx 1 root root 47 Sep 18 16:15 /OVS ->
/var/ovs/mount/F4135C096045458195057412169071E5
```

9.2 Managing Storage Repositories

Storage repositories are managed by Oracle VM Agent. When you create a storage repository, Oracle VM Agent saves the configuration information in the Oracle VM Agent database. You can have multiple storage repositories for Oracle VM resources.

If you create a server pool using Oracle VM Manager, the storage repository is automatically propagated to the other nodes (Virtual Machine Servers) in the server pool (cluster).

You can manage storage repositories using the `/opt/ovs-agent-2.3/utils/repos.py` script. The `repos.py` script should be run on the Server Pool Master.

The `repos.py` script options are discussed in this section. The script has commands for:

- [Listing the Storage Repositories](#)
- [Adding a Storage Repository](#)
- [Deleting a Storage Repository](#)
- [Initializing the Storage Repositories](#)
- [Setting the Cluster Root](#)

The `repos.py` script also contains a function to display help on the script parameters. To display the full list of options for the `repos.py` script, enter

```
/opt/ovs-agent-2.3/utils/repos.py [-h | --help]
```

9.2.1 Listing the Storage Repositories

You can list all the storage repositories with the `repos.py` script. The script takes the following parameters to list the storage repositories:

```
/opt/ovs-agent-2.3/utils/repos.py [-l | --list]
```

[-l | --list]

Lists the storage repositories.

The output format used to list the repositories is:

```
[* | Del | New | R] uuid => source
```

[* | Del | New | R]

The status of the storage repository. The status be may empty, or one of:

- *****: The cluster root for the storage repository. The cluster root is linked to the `/OVS` directory and stores all the key meta data for Oracle VM Agent and the storage repositories.
- **Del**: The storage repository is deleted.
- **New**: The storage repository is created.
- **R**: The storage repository is newly assigned as the root repository, but is not initialized.

uuid

The UUID (Universally Unique Identifier) of the storage repository.

source

The block device or path to the file system used for the storage repository.

For example, to list the storage repositories, enter

```
# /opt/ovs-agent-2.3/utils/repos.py -l
```

In this example, the output contains two storage repositories, the second being the default storage repository which is set as the cluster root:

```
[ ] e9253250-7955-4773-9e26-5954a5e95e57 => /dev/mpath/mpath1
[ * ] f4135c09-6045-4581-9505-7412169071e5 => example.com:/OVS
```

9.2.2 Adding a Storage Repository

You can add a file system or shared virtual disk as a storage repository using the `repos.py` script. The script identifies the file system or shared virtual disk as a storage repository and updates the storage repository configuration in the Oracle VM Agent database to enable it. You can use this command to create one or more storage repositories. The script takes the following parameters to add a storage repository:

```
/opt/ovs-agent-2.3/utils/repos.py [-n | --new] storage [-o NFS-options]
```

[-n | --new]

Adds the storage repository.

storage

The block device path to the file system to be added.

-o *NFS-options*

The NFS mount options to use. You should only use this parameter for NFS-based storage.

If you add an additional storage repository to an existing server pool, you must initialize it to create the storage repository directories, and mount it on all the nodes in the server pool (cluster). To initialize the storage repositories, see [Section 9.2.4, "Initializing the Storage Repositories"](#).

For example, to create a storage repository, on the Server Pool Master enter

```
# /opt/ovs-agent-2.3/utils/repos.py -n example.com:/test
```

In this example, the output displays the storage repository, which is marked as new:

```
[ NEW ] af60bcea-c588-4b51-8bec-d89d1a490e2e => example.com:/test
```

In the following example, an NFS mount is used to create the repository and the NFS mount options passed in.

```
# /opt/ovs-agent-2.3/utils/repos.py -n example.com:/test1 -o
rw,fg,hard,nointr,rsize=32768,wsz=32768,tcp,actimeo=0,nfsvers=3,timeo=600
```

9.2.3 Deleting a Storage Repository

You can remove a storage repository using the `repos.py` script. If the storage repository is mounted, it is automatically unmounted before it is deleted. Deleting a storage repository that is set as the cluster root causes the cluster configuration to be removed. The script takes the following parameters to delete a storage repository:

```
/opt/ovs-agent-2.3/utils/repos.py [-d | --delete] uuid
```

[-d | --delete]

Removes the storage repository.

uuid

The UUID (Universally Unique Identifier) of the storage repository.

If you delete a storage repository from an existing server pool, you must initialize any remaining storage repositories. To initialize the storage repositories, see [Section 9.2.4, "Initializing the Storage Repositories"](#).

For example, to delete a storage repository, on the Server Pool Master enter

```
# /opt/ovs-agent-2.3/utils/repos.py -d af60bcea-c588-4b51-8bec-d89d1a490e2e
```

In this example, the output displays the storage repository, which is marked as deleted:

```
[ DEL ] af60bcea-c588-4b51-8bec-d89d1a490e2e => example.com:/test
```

9.2.4 Initializing the Storage Repositories

Initializing the storage repositories creates the storage repository directories, and mounts them on all the nodes in the server pool (cluster).

You should initialize the storage repositories if you make any changes to existing storage repositories, such as adding, deleting or changing the cluster root.

Note: If you use Oracle VM Manager to create a server pool, you do not need to initialize the storage repositories. Oracle VM Agent automatically mounts the shared storage on each Virtual Machine Server in the server pool.

The script takes the following parameters to initialize the storage repositories:

```
/opt/ovs-agent-2.3/utils/repos.py [-i | --init]
```

[-i | --init]

Initializes the storage repositories by creating the storage repository directories, if required, and mounting them on all nodes in the server pool (cluster).

For example, to initialize the storage repositories, on the Server Pool Master enter

```
# /opt/ovs-agent-2.3/ovs-agent/ovs-agent.py -i
```

9.2.5 Setting the Cluster Root

If you have multiple storage repositories, you can select one storage repository as the cluster root in a server pool (cluster). Alternatively, if you have only one Oracle VM Server in the server pool, you can select your local disk as the storage repository and set it as the cluster root. The script takes the following parameters to set a storage repository as the cluster root:

```
/opt/ovs-agent-2.3/utils/repos.py [-r | --root] uuid
```

[-r | --root]

Sets a storage repository as the cluster root.

uuid

The UUID (Universally Unique Identifier) of the storage repository.

If you change the cluster root from one storage repository to another in an existing server pool, you must initialize the storage repositories. To initialize the storage repositories, see [Section 9.2.4, "Initializing the Storage Repositories"](#).

For example, to set a storage repository as the cluster root, on the Server Pool Master enter

```
# /opt/ovs-agent-2.3/utils/repos.py -r e9253250-7955-4773-9e26-5954a5e95e57
```

In this example, the output displays the storage repository, which is marked as the cluster root:

```
[ * ] e9253250-7955-4773-9e26-5954a5e95e57 => /dev/mpath/mpath1
```

Converting Hosts and Non-Oracle VM Virtual Machines

This Chapter discusses creating hardware virtualized guest images from existing physical computers running Linux or Windows, and converting non-Oracle VM virtual machines to Oracle VM guest images. This Chapter contains:

- [Converting a Linux or Windows Host](#)
- [Converting a Non-Oracle VM Virtual Machine](#)

You can import and manage guest images in Oracle VM Manager. See the *Oracle VM Manager User's Guide* for information on importing and managing guest images in Oracle VM Manager.

10.1 Converting a Linux or Windows Host

You can convert a Linux or Windows computer to an Oracle VM hardware virtualized guest image using the Physical to Virtual (P2V) conversion utility. The P2V utility is included on the Oracle VM Server CD. The operating system must be one of the Oracle VM supported guest operating systems. See the *Oracle VM Server Release Notes* for a list of the supported guest operating systems. The host computer must also have a CPU that supports PAE (Physical Address Extension).

The P2V conversion process creates a virtual machine configuration file (vm.cfg), allows you to make some modifications in terms of sizing of the virtual machine hardware, and then replicates the physical image and transfers it over the network to the server pool's repository using Oracle VM Manager. The image on your physical computer is not changed in any way.

The P2V utility converts disks on the computer to virtual disk images. The first four virtual disk images are created as IDE disks (hda, hdb, hdc, and hdd) on the guest, using the original disk names. Up to seven additional disks are created as SCSI devices (sda, sdb, sdc, and so on). The disk entries in the vm.cfg file look similar to:

```
disk = ['file:System-sda.img,hda,w',  
       'file:System-sdb.img,hdb,w',  
       'file:System-sdc.img,hdc,w',  
       'file:System-sdd.img,hdd,w',  
       'file:System-sde.img,sda,w',  
       'file:System-sdf.img,sdb,w',  
       'file:System-sdg.img,sdc,w',  
       'file:System-sdh.img,sdd,w',  
       'file:System-sdi.img,sde,w',  
       'file:System-sdj.img,sdf,w',  
       'file:System-sdk.img,sdg,w',
```

]

The hardware virtualized guest created by the P2V utility must have its own network configuration. If you use the same network configuration as the original computer, a network clash may occur as two computers on the network may have the same IP and MAC address. When the guest is started, make sure Kudzu detects the network device and configures the new network device.

You can run the P2V utility interactively, or as an automated process using a kickstart configuration file. When you use the P2V utility with a kickstart file, no user intervention is required.

10.1.1 Using the P2V Utility Interactively

When you use the P2V utility interactively, you are prompted for all required information. To create an Oracle VM guest image of a computer using the P2V utility interactively:

1. Insert the Oracle VM Server CDROM into your CDROM drive.
2. Start the computer with the Oracle VM Server CDROM.
3. The **Oracle VM Server** screen is displayed.

Figure 10–1 Oracle VM Server Installation Screen

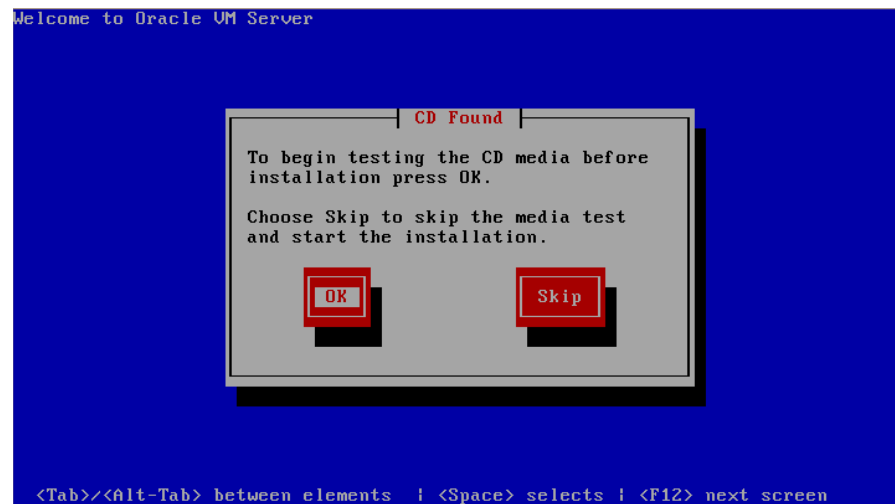


At the boot: prompt, enter:

```
linux p2v
```

Press **Enter**.

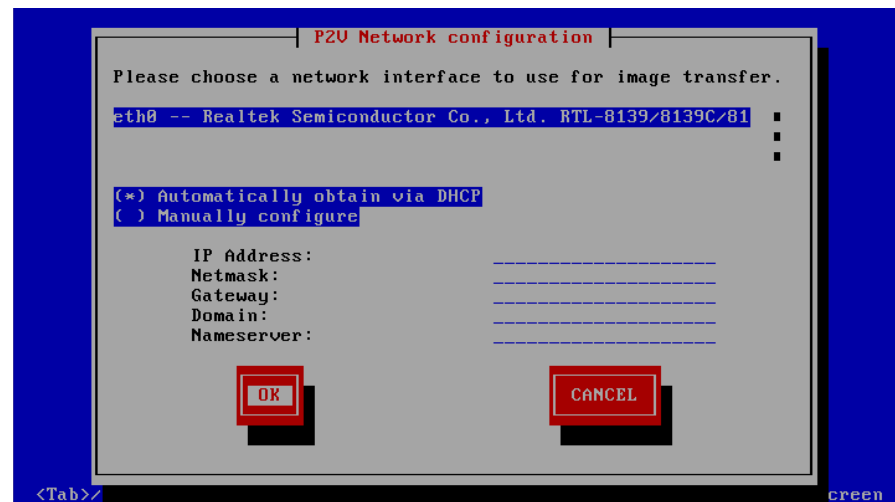
4. The **CD Found** screen is displayed.

Figure 10–2 CD Found Screen

If you want to make sure the CDROM is error free, you can have the installer test it for errors. To test the CDROM, select **OK** and press **Enter**. The CDROM is tested and any errors are reported.

To skip media testing and continue with the installation, select **Skip** and press **Enter**.

5. The **P2V Network Configuration** screen is displayed.

Figure 10–3 P2V Network Configuration Screen

Select your ethernet driver from the list displayed.

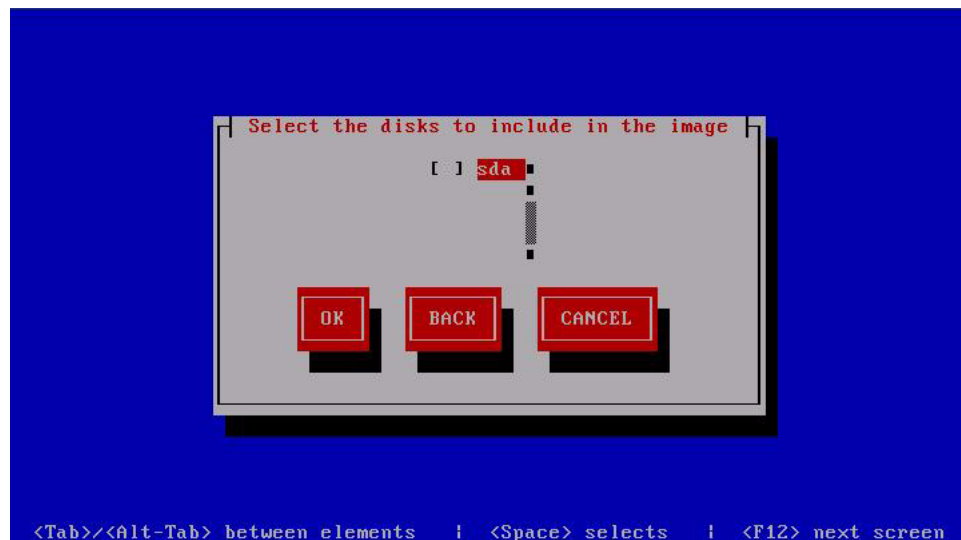
If your computer uses DHCP to assign its IP address, select **Automatically obtain via DHCP**.

If your computer uses a static IP address, select **Manually configure**, and enter the IP address and netmask, gateway, domain and nameserver for your computer.

Select **OK** and press **Enter**.

6. The disk selection screen is displayed.

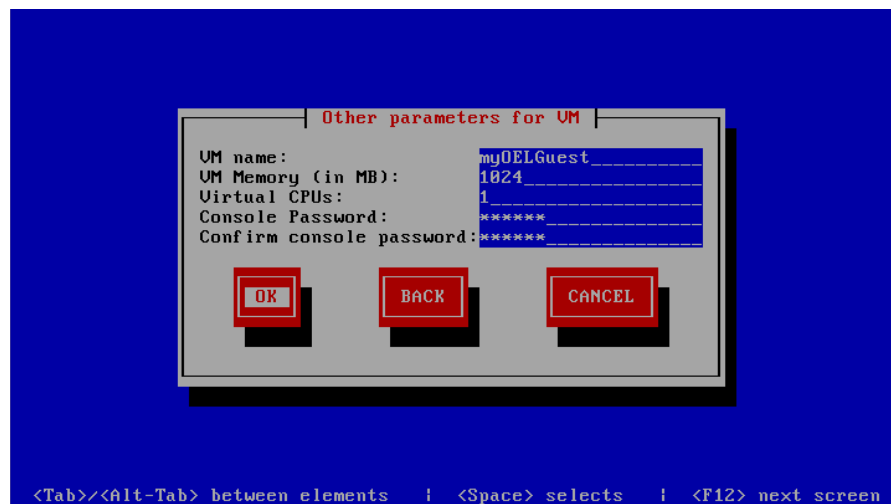
Figure 10–4 Disk Selection Screen



Select the disk partition(s) on the computer to include in the guest image. Select **OK** and press **Enter**.

7. The **Other parameters for VM** screen is displayed.

Figure 10–5 Other Parameters for VM Screen

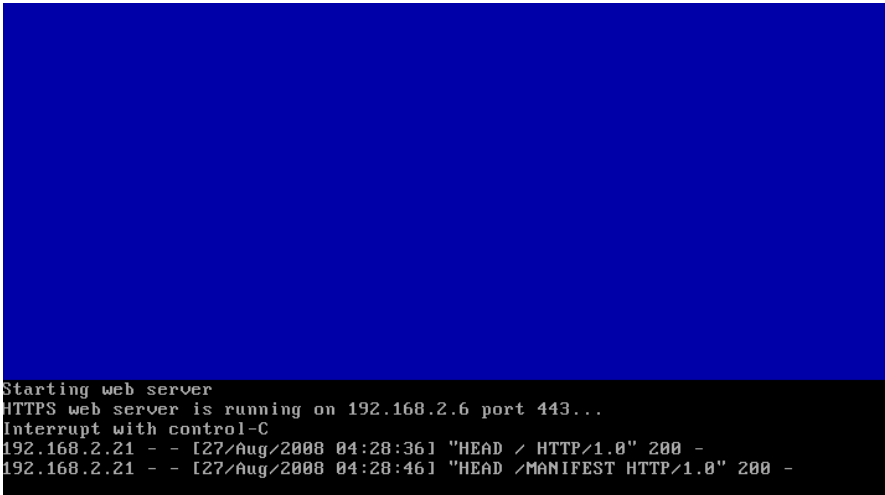


Enter information about the guest image for:

- VM (guest) name
- VM (guest) memory
- Number of virtual CPUs
- Console password

Select **OK** and press **Enter**.

8. A secure web server (HTTPS) is started. The IP address of the computer, and port number the web server is available on is displayed.

Figure 10–6 Web Server Screen


```
Starting web server
HTTPS web server is running on 192.168.2.6 port 443...
Interrupt with control-C
192.168.2.21 - - [27/Aug/2008 04:28:36] "HEAD / HTTP/1.0" 200 -
192.168.2.21 - - [27/Aug/2008 04:28:46] "HEAD /MANIFEST HTTP/1.0" 200 -
```

Log in to Oracle VM Manager and import the guest using the P2V feature. See the *Oracle VM Manager User's Guide* for information on importing P2V guest images.

9. The guest image is created and transferred to the server pool's repository. To cancel the transfer at any time, enter **Control+C**. When the file transfer is complete, Oracle VM Manager sets the status of the imported guest template as Pending.

Press **Control+Alt+Delete** to terminate the P2V utility on the computer. Remove the Oracle VM Server CDROM from your CDROM drive. Restart the computer.

The guest image is created and transferred to the server pool's repository as a hardware virtualized guest template.

10.1.2 Using the P2V Utility with a Kickstart File

You can use a kickstart file to automate the creation of a guest image of a physical computer using the P2V utility. When you use the P2V utility with a kickstart file, no user intervention is required. If there are any missing parameters in the kickstart file, you are prompted to enter them.

To use a P2V kickstart file, you must create a file with the P2V configuration options and parameters and place it on a kickstart server. The kickstart server can be made available using NFS, FTP, or HTTP. The kickstart server is set up in the same way as a standard Oracle Linux or Red Hat kickstart server.

The following example P2V kickstart file starts sends the guest image to an instance of Oracle VM Manager via network device eth0, which obtained an IP address via DHCP:

```
p2v
cdrom
lang en_US.UTF-8
keyboard us
target --ovmmanager
network --device eth0 --bootproto dhcp
diskimage --device /dev/sda --type IDE
vm_options --name myGuest --mem 1024 --vcpus 1 --consolepasswd mypassword
```

See "P2V" in [Appendix A, "Command-Line Tools"](#) for detailed information on P2V kickstart file options and parameters.

To create an Oracle VM guest image of a computer using the P2V utility with a kickstart file:

1. Create a P2V kickstart file and copy it to your kickstart server.
2. Insert the Oracle VM Server CDROM into your CDROM drive.
3. Restart the computer with the Oracle VM Server CDROM.
4. The **Oracle VM Server** screen is displayed. At the `boot:` prompt, enter `linux p2v` and the protocol and location for the kickstart file. For example, to use a kickstart file called `ks.cfg` on an HTTP server named `http://example.com`, enter:

```
linux p2v ks=http://example.com/mypath/ks.cfg
```

Press **Enter**.

5. If there are any missing parameters in the kickstart file, you are prompted to enter them.
6. If the kickstart file includes the directive to import the guest image to Oracle VM Manager, a secure web server (HTTPS) is started. A screen is displayed giving the IP address of the computer, and port number the web server is available on. Log in to Oracle VM Manager and import the guest using the P2V feature. See the *Oracle VM Manager User's Guide* for information on importing P2V guest images.
7. Remove the Oracle VM Server CDROM from your CDROM drive. Restart the computer.

The guest image is created and transferred to the server pool's repository as a hardware virtualized guest template.

10.2 Converting a Non-Oracle VM Virtual Machine

Oracle VM Manager automatically converts a VMware virtual machines to an Oracle VM guest image when you import it into Oracle VM Manager. You can also import virtual machines in VHD (Virtual Hard Disk) format. See the *Oracle VM Manager User's Guide* for information on importing non-Oracle VM guest virtual machines.

Command-Line Tools

This Appendix contains references for the Oracle VM Server and Oracle VM Agent command-line tools. The command-line interfaces in this Appendix are:

- [ovs-agent](#)
- [virt-install](#)
- [xm](#)
- [P2V](#)

ovs-agent

The ovs-agent command-line tool enables you to configure, and control Oracle VM Agent. Enter parameters to the ovs-agent command-line tool in the format

`service ovs-agent {option}`

If you shut down or restart the Oracle VM Agent on an HA-enabled Oracle VM Server on which guests are running you are prompted to:

- Migrate or Power Off the guest(s) using Oracle VM Manager. When the guests have been migrated or Powered Off, the Oracle VM Agent shuts down.
- Shut down the guest(s) and shut down Oracle VM Agent.
- Cancel the shutdown operation.

See [Chapter 3, "Oracle VM Agent"](#) for examples on using the ovs-agent command-line tool.

Options

start

Starts Oracle VM Agent.

```
# service ovs-agent start
```

stop

Stops Oracle VM Agent.

```
# service ovs-agent stop
```

restart

Stops and restarts Oracle VM Agent.

```
# service ovs-agent restart
```

status

Displays information on the status of the Oracle VM Agent daemon.

```
# service ovs-agent status
```

configure

Starts the Oracle VM Agent interactive configuration script.

```
# service ovs-agent configure
```


virt-install

The virt-install command-line tool creates paravirtualized guests and hardware virtualized guests. virt-install can be used as an interactive shell, or all parameters can be given at the same time. Enter multiple parameters to the virt-install command-line tool in the format:

```
virt-install [option ...]
```

This section contains a brief explanation of some of the more common virt-install options. For full documentation, use the `virt-install -h` command.

[Chapter 4, "Creating a Guest Virtual Machine"](#) discusses using the virt-install tool.

Options

[-h | --help]

Displays the virt-install command parameters and their purpose.

```
# virt-install -h
```

[-nname | --name=name]

Sets the name of the guest instance.

```
# virt-install -nMyGuest
```

[-rRAM | --ram=RAM]

Sets the memory to allocate for a guest instance in Megabytes.

```
# virt-install --ram=256
```

[-uUUID | --uuid=UUID]

Sets the UUID (Universally Unique Identifier) for the guest. If none is given, a random UUID is generated.

```
# virt-install -u
```

[--vcpus=number]

Sets the number of virtual CPUs to configure for the guest.

```
# virt-install --vcpus=2
```

[-fdiskfile | --file=diskfile]

Sets the file to use as the disk image.

```
# virt-install --file=/home/myhome/myimage
```

[-sfilesize | --file-size=filesize]

Sets the size of the disk image (if it does not exist) in Gigabytes.

```
# virt-install -s2
```

[-nonsparse]

Do not use sparse files for disks. This option may be significantly slower when creating guests.

[-mvalue | --mac=value]

Sets the fixed MAC address for the guest; if none or RANDOM is given, a random address is used.

```
# virt-install --mac=RANDOM
```

[-bvalue | --bridge=value]

Sets the bridge to connect guest NIC to. If none is given, attempts to determine the default.

[--vnc]

Use VNC (Virtual Network Computing) for graphics support.

```
# virt-install --vnc
```

[--vncport=port]

Sets the port to use for VNC connections.

```
# virt-install --vncport=5900
```

[--sdl]

Use SDL (Simple DirectMedia Layer) for graphics support.

```
# virt-install --sdl
```

[--nographics]

Do not use a graphical console for the guest.

```
# virt-install --nographics
```

[--noautoconsole]

Do not automatically connect to the guest console.

```
# virt-install --noautoconsole
```

[-kvalue | --keymap=value]

Set up keyboard mapping for the graphical console. If none is given, the keymap is automatically set to the local keymap.

```
# virt-install --de
```

[--accelerate]

Use kernel acceleration capabilities.

```
# virt-install --accelerate
```

[--connect=URI]

Connect to hypervisor with URI.

```
# virt-install --connect=test:///default
```

[-v | --hvm]

Sets the guest as being a fully virtualized guest.

```
# virt-install -v
```

[-cCD-ROM | --CD-ROM=CD-ROM]

Sets the file to use a virtual CD-ROM device for fully a virtualized guest.

[--os-type=type]

Sets the operating system type for a fully virtualized guest. Possible values are windows, unix, other, and linux.

```
# virt-install --os-type=windows
```

[--os-variant=variant]

Sets the operating system variant for a fully virtualized guest, for example, rhel5, win2k, or vista. This parameter should be used with the os-type parameter.

The following table lists the possible values available for `os-variant` for each `os-type` option.

Possible values for <code>os-type=windows</code>	Possible values for <code>os-type=unix</code>	Possible values for <code>os-type=other</code>	Possible values for <code>os-type=linux</code>
win2k3	solaris9	netware6	generic24
win2k	solaris10	generic	generic26
vista	freebsd6	netware4	rhel2.1
winxp	openbsd4	msdos	fedora7_64
		netware5	el5_64
			fedora6
			fedora7
			fedora5
			centos5_64
			generic26_64
			centos5
			sles10
			sles10_64
			el4_64
			rhel4
			rhel5
			rhel4_64
			rhel3
			fedora6_64
			rhel5_64
			fedora5_64
			el4
			el5

Note: Not all operating system variants are supported by Oracle for use with Oracle products, but are made available for your convenience.

```
# virt-install --os-type=windows --os-variant=winxp
```

[--noapic]

Disables APIC (Advanced Programmable Interrupt Controller) for a fully virtualized guest. Overrides the value set in `--os-type` and `--os-variant`.

[--arch=*arch*]

Sets the CPU architecture to simulate.

```
# virt-install --arch=x86
```

[-p | --paravirt]

Sets the guest as being a paravirtualized guest.

[-l *location* | --location=*location*]

Sets the installation source for a paravirtualized guest, for example, `nfs:host:/path`, `http://host/path`, or `ftp://host/path`.

```
# virt-install -lhttp://example.com/path
```

[--vif-type=*type*]

Sets the virtual network interface type for hardware virtualized guests. The netfront driver is a paravirtualized driver which can be used with a paravirtualized guest, or with a hardware virtualized guest with the proper paravirtualized drivers installed. The ioemu driver is a hardware virtualized driver, and can only be used with a hardware virtualized guest. Both drivers contain the device emulation code to support hardware virtualized guests.

For hardware virtualized guests, `type` can be either `ioemu` or `netfront`. The default is `ioemu`.

You cannot use this parameter for paravirtualized guests. For paravirtualized guests, the default is `netfront` and cannot be changed.

```
# virt-install --vif-type=ioemu
```

[[-xargs | --extra-args=*args*] ...]

Any additional arguments to pass to the installer with a paravirtualized guest.

[-d | --debug]

Prints debugging information.

xm

The Oracle VM Server management command-line management tool `xm`, creates, destroys, manages and migrates guests.

This section contains a brief explanation of some of the more common `xm` commands. For full documentation, use the `xm help --long` command.

The `xm` command-line tool requires the `xend` daemon to be started.

Enter parameters to the `xm` command-line tool in the format:

```
xm [option] [argument]
```

See [Chapter 5, "Domain Monitoring and Administration"](#) for examples on using the `xm` command-line tool.

Options

block-attach {domain-id} {be-dev} {fe-dev} {mode} [bedomain-id]

Creates a new virtual block device. This triggers a hotplug event for the guest.

The *domain-id* parameter is the domain identifier.

The *be-dev* parameter is the device in the back-end domain (usually `dom0`) to be exported. Can be specified as a physical partition (`phy:sda7`), or as a file mounted as loopback (`file://path/to/loop.iso`).

The *fe-dev* parameter specifies how the device is presented to the guest domain. Can be specified as either a symbolic name, such as `/dev/hdc` for common devices, or by a device identifier, such as `0x1400` (`/dev/hdc` device identifier in hexadecimal).

The *mode* parameter sets the access mode for the device from the guest domain. Supported modes are `w` (read/write) or `r` (read-only).

The *bedomain-id* parameter is the back-end domain hosting the device. Defaults to `dom0`.

For example, to mount an ISO as a disk:

```
# xm block-attach 02 file://path/to/dsl-2.0RC2.iso /dev/hdc ro
```

This mounts the ISO as `/dev/hdc` in the guest domain as a read only device. This may not be detected as a CD-ROM by the guest, but mounting `/dev/hdc` gives you access to the ISO.

block-detach {domain-id} {device-id} [--force]

Detaches a virtual block devices from a domain.

The *device-id* parameters may be the symbolic name or the numeric device identifier given to the device by `dom0`. Run the `xm block-list` command to determine the device identifier.

Detaching the device requires the co-operation of the domain. If the domain fails to release the device (perhaps because the domain is hung or is still using the device), the detach fails. The `--force` parameter forces the device to detach, but may cause IO errors in the domain.

```
# xm block-detach 02 51776
```

block-list {*domain-id*} [-l | --long]

Lists the virtual block devices for a domain.

```
# xm block-list 02 --long
```

console {*domain-id*}

Attaches to a domain's console.

```
# xm console 02
```

create [-c] {*config-file* [*name=value ...*]}

Creates a domain based on the entries in the *config-file*.

Entering the *-c* parameter attaches to the domain's console when the domain is created and started. You can also enter name value pairs to override variables in the *config-file* using the *name=value* parameter.

```
# xm -c /home/myhome/myconfig
```

destroy {*domain-id*}

Immediately terminates a domain.

```
# xm destroy 02
```

dmesg [--clear]

Displays message buffer logs similar in format to the equivalent to the dmesg command in the Linux kernel.

The *--clear* parameter clears the message buffer.

help [--long] [*option*]

Displays help on the xm command, and its options.

The *--long* option displays full help on xm commands, grouped by function.

Enter a command name as an option to the xm command to get help only on that command.

```
# xm help --long create
```

info

Displays information about the host computer.

```
# xm info
```

list [--long | --label] [*domain-id*, ...]

Displays information on all the running domains.

The *--long* option displays full information on running domains.

Enter the *domain-id* as an option to the xm command to get information on only that domain, or a set of domains.

```
# xm list --long 02
```

log

Displays logs similar in format to the equivalent for the Linux kernel. The log file is located at */var/log/xend.log*.

```
# xm log
```

mem-max {*domain-id*} {*MBs*}

Allocates the maximum memory available to the domain in MBs. Overrides the *maxmem* setting in the configuration file.

```
# xm mem-max 02 2048
```

mem-set {domain-id} {MBs}

Allocates domain memory up to the value of the `maxmem` setting in the guest configuration file or the `mem-max {domain-id} {MBs}` setting. Overrides the `memory` setting in the guest configuration file.

```
# xm mem-set 02 1024
```

migrate {domain-id} {host} [-l | --live] [-r=MB | --resource=MB]

Migrates a domain to another computer.

The `domain-id` parameter is the domain to migrate.

The `host` parameter is the target computer.

The `--live` parameter migrates the domain without shutting down the domain.

The `--resource` parameter sets the maximum amount of Megabytes to be used.

```
# xm migrate 02 example.com --live
```

new [config-file] [option ...] [name=value ...]

Adds a domain to Oracle VM Server domain management.

You can set domain creation parameters with a number of command-line options, a Python script (with the `--defconfig` parameter), or an SXP configuration file (the `--config` parameter).

You can set configuration variables with `name=value` pairs, for example `vmid=3` sets `vmid` to 3.

The `config-file` parameter is the location of the domain configuration file.

The `option` parameter is one or more of the following:

`[-h | --help]`

Displays help on the command.

`[--help-config]`

Prints the available configuration variables for the configuration script.

`[-q | --quiet]`

Quiet.

`[--path=path]`

Searches the location given in `path` for configuration scripts. The value of `path` is a colon-separated directory list.

`[-f=file | --defconfig=file]`

Uses the given Python configuration script. The script is loaded after arguments have been processed. Each command-line option sets a configuration variable named after its long option name, and these variables are placed in the environment of the script before it is loaded. Variables for options that may be repeated have list values. Other variables can be set using `name=value` on the command-line. After the script is loaded, values that were not set on the command-line are replaced by the values set in the script.

`[-F=file | --config=file]`

Sets the domain configuration to use SXP. SXP is the underlying configuration format used by Xen. SXP configurations can be hand-written or generated from Python configuration scripts, using the `--dryrun` option to print the configuration.

`[-n | --dryrun]`

Prints the resulting configuration in SXP, but does not create the domain.

`[-x | --xmldryrun]`

Prints the resulting configuration in XML, but does not create the domain.

`[-s | --skipdtd]`

Skips DTD checking and XML checks before domain creation. This option is experimental and may slow down the creation of domains.

`[-p | --paused]`

Leaves the domain paused after it is created.

`[-c | --console_autoconnect]`

Connects to the console after the domain is created.

```
# xm new /home/myhome/myconfig
```

pause {domain-id}

Pauses the execution of a domain.

```
# xm pause 02
```

reboot [--all] [--wait] [domain-id]

Reboots a domain.

The `--all` parameter reboots all domains.

The `--wait` parameter waits for the domain to reboot before returning control to the console.

```
# xm reboot --wait 02
```

restore {statefile}

Restores a domain from a saved state.

```
# xm restore /home/myhome/statefile
```

save {domain-id} {statefile}

Saves a domain state so it can be restored at a later date.

```
# xm save 02 /home/myhome/statefile
```

shutdown [-a] [-w] [domain-id]

Shuts down a domain gracefully.

The `-a` parameter shuts down all domains.

The `-w` parameter waits for the domain to shut down before returning control to the console.

```
# xm shutdown -w 02
```

top

Displays real time monitoring information of the host and domains.

```
# xm top
```


unpause {domain-id}

Unpauses a paused domain.

```
# xm unpause 02
```

vcpu-set {domain-id} {vCPUs}

Sets the number of virtual CPUs for the domain, up to the value of the `vcpus` setting in the guest configuration file.

```
# xm vcpu-set 02 8
```

You can set which virtual CPUs are used with the `vcpu_avail` parameter in the configuration file. See [Section C.3, "Virtual CPU Configuration File Parameters"](#).

P2V

The Physical to Virtual (P2V) conversion utility enables you to convert a computer's operating system (Linux and Windows) and applications to an Oracle VM hardware virtualized guest image. The P2V utility is included on the Oracle VM Server CD. You can access the P2V utility by restarting a computer with the Oracle VM Server CD. The Oracle VM Server startup screen is displayed. At the `boot :` prompt, enter:

```
linux p2v
```

You can use a P2V kickstart file to automate creation of hardware virtualized guest images from physical computers. This section discusses the options and parameters of the P2V kickstart file.

The P2V utility converts disks on the computer to virtual disk images. The virtual disk images are created as IDE disks (`hda`, `hdb`, `hdc`, `hdd`, and so on) on the guest, using the original disk names. When you use a P2V kickstart file, up to four disks are automatically deployed in the guest. Any extra disks are converted and added to the guest configuration file (`vm.cfg`), although they are not deployed. To deploy the additional disks in the guest, edit the guest configuration file, remove the comments from the disk entries, and map the additional disks to SCSI device names, for example, `sda`, `sdb`, and `sdc`. The boot disk must always be mapped to device `hda`. Any files on the guest which contain references to these devices must also be changed, for example, the `/etc/fstab` file may contain references to `/dev/hda1`, `/dev/sda1`, and so on.

When you use a P2V kickstart file, at least one network interface must use DHCP. This is required for the computer running the P2V utility to read the kickstart file over the network. The network configuration for this network interface cannot be modified from the kickstart file.

If you want the P2V utility's web server to listen using a network interface other than the one used to initiate the kickstart session, the network configuration (DHCP or static IP address) for that network interface can be specified in the kickstart file.

A number of screens may be displayed prior to the P2V utility starting with a kickstart file. You can suppress these screens to fully automate the P2V utility. Prior to the P2V utility starting, you may see up to four screens:

- P2V Network Configuration screen
- Language selection screen
- Keyboard selection screen
- Installation source screen

The following sections give examples on how to suppress these screens.

Suppressing the P2V Network Configuration Screen

To suppress the P2V Network Configuration screen, supply the ethernet device on the command line, for example:

```
linux p2v ks=http://example.com/ks.cfg ksdevice=eth0
```

Suppressing the Language Selection Screen

To suppress the Language selection screen, supply the language kickstart parameter, for example:

```
lang en_US.UTF-8
```

Suppressing the Keyboard Selection Screen

To suppress the Keyboard selection screen, supply the keyboard kickstart parameter, for example:

```
keyboard us
```

Suppressing the Installation Source Screen

To suppress the Installation source screen, supply the source kickstart parameter, for example:

```
cdrom
```

Example P2V Kickstart File

An example P2V kickstart file follows:

```
p2v
cdrom
lang en_US.UTF-8
keyboard us
target --ovmmanager
network --device eth0 --bootproto dhcp
diskimage --device /dev/sda --type IDE
vm_options --name myGuest --mem 1024 --vcpus 1 --consolepasswd mypassword
```

For more examples and information on using P2V kickstart files, see ["Converting a Linux or Windows Host"](#) in [Chapter 10, "Converting Hosts and Non-Oracle VM Virtual Machines"](#).

Options

The following parameters are accepted in a P2V kickstart file.

p2v

Indicates the kickstart file is intended to automate a P2V conversion. This parameter is required in order to perform an automated P2V conversion and should be supplied at the Oracle VM Server boot: prompt instead of install, update, or rescue. It accepts no parameters.

target [option]

Sets the end destination for the guest image.

The *option* parameter can only contain the following:

```
--ovmmanager
```

Sets the P2V utility to operate in HTTPS server mode to transfer the guest image to a running instance of Oracle VM Manager.

diskimage [option...]

Denotes a disk to be included in the guest image. The P2V utility uses device mapper-based snapshotting to copy the disk as a *system-*.img* file on the target computer. There may be multiple *diskimage* directives in a P2V kickstart file, each resulting in a disk image in the guest image. The *--device* parameter must always be used with the *diskimage* directive to indicate which device should be imaged.

The *option* parameter is one or more of the following:

```
--device path
```

The device to image. *path* must be the full path to the device. For example:

```
diskimage --device /dev/sda
```

--type [IDE | SCSI | LVM | MDRAID]

Sets the type of disk. Must be one of IDE, SCSI, LVM, or MDRAID. Devices /dev/hda, /dev/hdb, /dev/hdc, and /dev/hdd should be IDE. Devices /dev/sd[a-zz] should be SCSI. A logical volume should be LVM. Devices /dev/md[a-zz] should be MDRAID. For example:

```
diskimage --device /dev/hda --type IDE
```

network [option...]

Configures network information for the computer.

The *option* parameter is one or more of the following:

--bootproto [dhcp | bootp | static]

Sets the method by which the network configuration is determined. Must be dhcp, bootp, or static. The default is dhcp. bootp and dhcp are treated as the same.

dhcp uses a DHCP server to obtain the networking configuration, for example:

```
network --bootproto dhcp
```

static requires all the necessary networking information. As the name implies, this information is static and is used during and after the installation. The entry for static networking is more complex, as you must include all network configuration information on one line. You must specify the IP address, netmask, gateway, and nameserver, for example:

```
network --bootproto static --ip 10.0.2.15 --netmask 255.255.255.0 --gateway  
10.0.2.254 --nameserver 10.0.2.1
```

The static method has the following restrictions:

- All static networking configuration information must be specified on one line; you cannot wrap lines using a backslash.
- You can only specify one nameserver.

--ip *ipaddress*

The IP address for the computer.

--gateway *ipaddress*

The IP address for the default gateway.

--nameserver *ipaddress*

The IP address for the primary nameserver.

--netmask *netmask*

The netmask for the computer.

vm_options [option...]

Sets the configuration options for the guest.

--name *name*

The name of the guest.

--mem *size*

The memory allocation for the guest in Mb.

`--vcpus number`

The number of VCPUs for the guest.

`--consolepasswd password`

The console password for the guest. For example:

```
vm_options --name myGuest --mem 1024 --vcpus 1 --consolepasswd mypassword
```

Oracle VM Server Configuration File

This Appendix contains information on the entries in the Oracle VM Server configuration file. It contains:

- [Oracle VM Server Configuration File](#)

Oracle VM Server Configuration File

This section contains information on configuring Oracle VM Server using the configuration file. The configuration file options are available in the `/etc/xen/xend-config.sxp` file. When you make changes to this file, you must restart Oracle VM Server for the changes to take effect.

Logfile Options

logfile {*location*}

Specifies the location of the Oracle VM Server log which contains detailed information on guest start up, shut down, configuration, and error conditions. The default location is `/var/log/xen/xend.log`.

```
(logfile /var/log/xen/xend.log)
```

loglevel {CRITICAL or FATAL | ERROR | WARN or WARNING | INFO | DEBUG}

Sets the level of verbosity for the logfile parameter. The default is `DEBUG`.

```
(loglevel DEBUG)
```

Oracle VM Server API Options

xen-api-server {(*access-method*) ...}

***access-method* {(for local access): [(unix [*authtype*])]}**

***access-method* {(for remote access): [(*ipaddress*):*port* [*authtype* [*host-access* [*ssl-key* [*ssl-cert*]]]]]}**

Sets the configuration of the Oracle VM Server API which uses an XML-RPC interface to control and monitor guests and the dom0 host.

A list of access method entries should be provided, each entry in the list enclosed in parentheses, and the list itself enclosed in its own parentheses.

If dom0 local access is required, the access method entry should begin with the `unix` parameter. This creates a unix socket in a directory on the dom0 file system. An *authtype* parameter may also be supplied as a second argument.

If remote access is required, the access method entry should take a TCP port number as its first argument, or an *ipaddress:port* number pair. This TCP port is used to listen for incoming Oracle VM Server API requests on all dom0 network interfaces, or only on one specific interface if an IP address of a dom0 interface is given.

The optional *authtype* parameter can be set to `none` or `pam` (Pluggable Authentication Model).

The optional *host-access* parameter can be a list of space separated regular expressions to list the host IP addresses or host names to allow access. This parameter is only enabled for remote access.

The optional *ssl-key* is the private key for SSL communication. This parameter is only enabled for remote access.

The optional *ssl-cert* is the SSL certificate for SSL communication. This parameter is only enabled for remote access.

The default is `unix`.

```
(xen-api-server ((10.1.1.1:9363 none) (unix none)))  
(xen-api-server ((9363 pam '^localhost$ example\\.com$') (unix none)))
```



```
(xen-api-server ((9367 pam '' /etc/xen/xen-api.key /etc/xen/xen-api.crt)))
(xen-api-server ((unix)))
```

Oracle VM Server Options

xend-http-server {yes | no}

Sets the original Xen remote interface. This setting may be needed for some legacy applications that use HTTP. The default is no.

```
(xend-http-server yes)
```

xend-unix-server {yes | no}

Sets the original Xen remote interface. This setting may be needed for some legacy applications that use a local unix socket. The default is no.

```
(xend-unix-server yes)
```

xend-tcp-xmlrpc-server {yes | no}

Sets the legacy XML-RPC interface. This setting may be needed for some applications that use XML-RPC over TCP. The default is no.

```
(xend-tcp-xmlrpc-server no)
```

xend-unix-xmlrpc-server {yes | no}

Sets the legacy XML-RPC interface. This setting may be needed for some applications that use XML-RPC over unix sockets. The default is yes.

```
(xend-unix-xmlrpc-server yes)
```

xend-relocation-server {yes | no}

Sets the Oracle VM Server used for the live migration of domains. The default is no.

```
(xend-relocation-server no)
```

xend-unix-path {*path*}

Sets the path for the xend-unix-server socket parameter. The default is /var/lib/xend/xend-socket.

```
(xend-unix-path /var/lib/xend/xend-socket)
```

xen-tcp-xmlrpc-server-address {*IPAddress*}

Sets the IP address Oracle VM Server should use for the legacy TCP XMLRPC interface. This setting is used if xen-tcp-xmlrpc-server is set. The default is localhost.

```
(xen-tcp-xmlrpc-server-address 'localhost')
```

xen-tcp-xmlrpc-server-port {*port*}

Sets the port Oracle VM Server should use for the legacy TCP XMLRPC interface. This setting is used if xen-tcp-xmlrpc-server is set. The default is 8006.

```
(xen-tcp-xmlrpc-server-port 8006)
```

xend-tcp-xmlrpc-server-ssl-key-file {*key*}

Sets the SSL key file for the legacy XML-RPC interface if SSL is to be used. The default is none.

```
(xend-tcp-xmlrpc-server-ssl-key-file /etc/xen/xmlrpc.key)
```

xend-tcp-xmlrpc-server-ssl-cert-file {*file*}

Sets the SSL certificate file for the legacy XML-RPC interface if SSL is to be used. The default is none.

```
(xend-tcp-xmlrpc-server-ssl-cert-file /etc/xen/xmlrpc.crt)
```

xend-port {port}

Sets the port Oracle VM Server should use for the HTTP interface if xend-http-server is set. The default is 8000.

```
(xend-port 8000)
```

xend-relocation-port {port}

Sets the port Oracle VM Server should use for the relocation interface if xend-relocation-server is set. The default is 8002.

```
(xend-relocation-port 8002)
```

xend-address {IPAddress}

Sets the IP address Oracle VM Server should listen on for HTTP connections if xend-http-server is set. Setting to localhost prevents remote connections. Setting to an empty string allows all connections. The default is an empty string.

```
(xend-address "localhost")
```

xend-relocation-address {IPAddress}

Sets the IP address Oracle VM Server should listen on for relocation-socket connections if xend-relocation-server is set. Setting to localhost prevents remote connections. Setting to an empty string allows all connections. The default is an empty string.

```
(xend-relocation-address "localhost")
```

xend-relocation-hosts-allow {[IPAddress | regular_expression] ...}

Sets the hosts allowed to talk to the relocation port. Setting to an empty string allows all connections. Setting to a space separated series of regular expressions allows any host with the domain name or IP address that matches any of the regular expressions. The default is an empty string.

```
(xend-relocation-hosts-allow '^localhost$ ^.*\\.example\\.org$')
(xend-relocation-hosts-allow '')
(xend-relocation-hosts-allow '^localhost$')
```

console-limit {size}

Sets the limit in kilobytes of the console buffer. The default is 1024.

```
(console-limit 2048)
```

network-script 'network-bridge {[netdev={name}] | [bridge={name}]}'

Sets the network bridge to use. The default is to use the default Ethernet device as the outgoing interface.

```
(network-script 'network-bridge netdev=eth1')
(network-script 'network-bridge bridge=xenbr0')
(network-script 'network-bridge netdev=eth1 bridge=xenbr0')
(network-script my-network-bridge)
(network-script network-bridge)
```

vif-script {vif-bridge | vif-route | vif-nat}

Sets the script used to control virtual interfaces. The default is to use the value of vif-bridge.

```
(vif-script vif-bridge)
(vif-script vif-route)
(vif-script vif-nat)
```

dom0-min-mem {memory}

Sets the minimum memory level in Megabytes that dom0 can use. The default is 196.

```
(dom0-min-mem 256)
```

dom0-cpus {CPUs}

Sets the number of CPUs that dom0 can use. Setting it to 0 allows dom0 to use all CPUs on the computer. The default is 0.

```
(dom0-cpus 0)
```

enable-dump {yes | no}

Sets whether to core dump when a domain crashes. The default is no.

```
(enable-dump no)
```

external-migration-tool {tool}

Sets the tool to be used for initiating virtual TPM (Trusted Platform Module) migration. The default is an empty string.

```
(external-migration-tool '')
```

VNC Server Options

vnc-listen {port}

The port on which to listen for the hardware virtualized VNC Server. Setting the port to 0.0.0.0 allows access from all hosts. Setting the port to localhost restricts access to only the local host. The default is 127.0.0.1.

```
(vnc-listen '0.0.0.0')
```

vncpasswd {passwd}

Sets the password to use for hardware virtualized VNC Server connections. The password is the global default for all hardware virtualized guests. You can set to use no password with an empty string as the value. The default is an empty string (no password).

```
(vncpasswd 'mypassword')
```

Guest Configuration

This Appendix contains information about guest network driver installation, guest configuration file options and parameters, and examples of guest configuration files you can modify and use to create guests. A detailed explanation of the configuration parameters and common values is available in the `/etc/xen/xmexample.hvm` file in Oracle VM Server.

Create the guest configuration file as `/OVS/running_pool/domain/vm.cfg` and use the following command to create the guest:

```
# xm create /OVS/running_pool/domain/vm.cfg
```

This Appendix contains:

- [e100 And e1000 Network Device Emulators](#)
- [Quality of Service \(QoS\)](#)
- [Virtual CPU Configuration File Parameters](#)
- [Simple Configuration File Example](#)
- [Complex Configuration File Example](#)
- [Virtual Iron Migration \(VHD\) Configuration File Example](#)

C.1 e100 And e1000 Network Device Emulators

You can use the network device emulators for the Intel 8255x 10/100 Mbps Ethernet controller (the *e100* controller) and the Intel 82540EM Gigabit Ethernet controller (the *e1000* controller) for hardware virtualized guests. The *e1000* controller is a Gigabit Ethernet controller and increases the network throughput when compared to the default Ethernet controller.

To use these network device emulators, install the network device driver on the guest, then modify the guest configuration file to specify the controller model type: either *e100*; or *e1000*. For example, to use the *e1000* controller, set `model=e1000` in the `vif` entry in the guest configuration file:

```
vif = [ 'type=ioemu, mac=00:16:3e:09:bb:c6, bridge=xenbr2, model=e1000']
```

Create the guest again using the `xm create` command. The guest now uses the faster *e1000* controller.

C.2 Quality of Service (QoS)

Quality of Service (QoS) is the ability to provide varying priority to different applications, users, or data flows, or to guarantee a level of performance to a data flow. You can set virtual network interface and virtual disk QoS parameters for guests running on an Oracle VM Server. Guest virtual network interfaces share a physical network interface card (NIC) and you can control how much bandwidth is available to the virtual network interface. You can also control the I/O priority of a guest's virtual disk(s).

This section contains:

- [Setting Disk Priority](#)
- [Setting Inbound Network Traffic Priority](#)
- [Setting Outbound Network Traffic Priority](#)

You can set QoS parameters in Oracle VM Server, and in Oracle VM Manager. See the *Oracle VM Manager User's Guide* for information on setting QoS parameters in Oracle VM Manager.

C.2.1 Setting Disk Priority

You can set the priority of a guest's virtual disk(s). Eight priority levels are available to set the time slice a process receives in each scheduling window. The priority argument is from 0 to 7; the lower the number, the higher the priority. Virtual disks running at the same priority are served in a round-robin fashion.

The virtual disk priority is controlled with the `disk_other_config` parameter in the guest's configuration file (`vm.cfg`). The `disk_other_config` parameter is entered as a list; each list item represents a QoS setting. The syntax to use for the `disk_other_config` parameter is:

```
disk_other_config = [[ 'front_end', 'qos_algorithm_type', 'qos_algorithm_params' ]]
```

`front_end` is the front end name of the virtual disk device to which you want to apply QoS. For example, `hda`, `hdb`, `xvda`, and so on.

`qos_algorithm_type` is the QoS algorithm. Only `ionice` is currently supported.

`qos_algorithm_params` are the parameters for the `qos_algorithm_type`. For the `ionice` algorithm, this may be the schedule class and the priority, for example `sched=best-effort,prio=5`.

For example:

```
disk_other_config = [['hda', 'ionice', 'sched=best-effort,prio=5'], ['hdb', 'ionice', 'sched=best-effort,prio=6']]
```

If you make a change to a running guest's configuration file, you must shut down the guest, then start it again with the `xm create vm.cfg` command for the change to take effect. The `xm reboot` command does not restart the guest with the new configuration.

C.2.2 Setting Inbound Network Traffic Priority

You can set the priority of inbound network traffic for a guest. The inbound network traffic priority is controlled with the `vif_other_config` parameter in the guest's configuration file (`vm.cfg`). The `vif_other_config` parameter is entered as a list; each list item represents a QoS setting. The syntax to use for the `vif_other_config` parameter is:

```
vif_other_config = [[ 'mac', 'qos_algorithm_type', 'qos_algorithm_params' ]]
```

mac is the MAC address of the virtual network device to which you want to apply QoS.

qos_algorithm_type is the QoS algorithm. Only *tbfbf* is currently supported.

qos_algorithm_params are the parameters for the *qos_algorithm_type*. For the *tbfbf* algorithm, this may be the rate limit and latency, for example,
`rate=8mbit,latency=50ms`.

For example:

```
vif_other_config = [['00:16:3e:31:d5:4b', 'tbfbf', 'rate=8mbit,latency=50ms'],  
['00:16:3e:52:c4:03', 'tbfbf', 'rate=10mbit']]
```

If you make a change to a running guest's configuration file, you must shut down the guest, then start it again with the `xm create vm.cfg` command for the change to take effect. The `xm reboot` command does not restart the guest with the new configuration.

C.2.3 Setting Outbound Network Traffic Priority

You can set the priority of outbound network traffic for a guest. The outbound network traffic priority is controlled with the *rate* parameter of the *vif* option in the guest's configuration file (*vm.cfg*). The *rate* parameter supports an optional time window parameter for specifying the granularity of credit replenishment. The default window is 50ms. For example, you could set *rate* as `rate=10Mb/s`, `rate=250KB/s`, or `rate=1MB/s@20ms`. An example *vif* option to set the network traffic priority for a guest might be:

```
vif = ['mac=00:16:3e:31:d5:4b,bridge=xenbr0,rate=10Mb/s@50ms']
```

If you make a change to a running guest's configuration file, you must shut down the guest, then start it again with the `xm create vm.cfg` command for the change to take effect. The `xm reboot` command does not restart the guest with the new configuration.

C.3 Virtual CPU Configuration File Parameters

You can set the number of virtual CPUs to be used by a guest domain using the *vcpus* parameter in the configuration file. You can also set the *vcpu_avail* parameter to activate specific virtual CPUs when the domain starts up with the syntax:

```
vcpu_avail {bitmap}
```

The *bitmap* tells the domain whether it may use each of its virtual CPUs. *vcpu0* is always activated when the domain starts up, so the last bit of *vcpu_avail* is ignored. By default, all virtual CPUs are activated.

For example, to activate only *vcpu0*:

```
vcpu_avail = 1
```

And the following activates *vcpu0*, *vcpu3* and *vcpu4*:

```
vcpu_avail = 25 # 11001
```

This is equal to the following as *vcpu0* is always active:

```
vcpu_avail = 24 # 11000
```

C.4 Simple Configuration File Example

A simple example of a configuration file to create a guest follows:

```
disk = [ 'file:/mnt/el4u5_64_hvm/system.img,hda,w' ]
memory=4096
vcpus=2
name="el4u5_64_hvm"
vif = [ ' ' ] #By default no n/w interfaces are configured. E.g: A default hvm
install will have the line as vif=[ 'type=ioemu,bridge=xenbr0' ]
builder = "hvm"
device_model = "/usr/lib/xen/bin/qemu-dm"

vnc=1
vncunused=1

apic=1
acpi=1
pae=1
serial = "pty" # enable serial console

on_reboot = 'restart'
on_crash = 'restart'
```

C.5 Complex Configuration File Example

A more complex example of a configuration file to create a guest follows:

```
# An example of setting up the install time loopback mount
# using nfs shared directory with iso images
# to create "pseudo cdrom device" on /dev/loop*:
#
# mount ca-fileserver2:/vol/export /srv/
# mount -o loop,ro /srv/osinstall/RedHat/FC6/F-6-x86_64-DVD.iso /mnt
#
# You can tell what loop device to use by looking at /etc/mtab after the mount
# The first set of disk parameters commented out below are
# "install time disk parameters" with the "pseudo" cdrom.
# Your new domU HVM install will see "/dev/sda" just like a usual hardware
# machine.
#disk = [ 'phy:/dev/vgxen/lvol0,hda,w', 'phy:/dev/loop0,hdc:cdrom,r' ]
# Example of after-setup "HVM up and running" disk parameters below;
# the last three devices were added later
# and last two are shared, writeable.
# Note, for HVM you must use "whole" device.
# Do not try to get domU to see a partition on a device...
# For example, in a HVM this will not work : 'phy:/dev/vgxen/tls4-swap,hdb1,w'
# Best that you fdisk any extra or added devices within one of your domUs
disk = [ 'phy:/dev/vgxen/lvol0,hda,w',
        'phy:/dev/vgxen/tls4-swap,hdb,w',
        'phy:/dev/vgxen/sharedvol1,hdc,w!',
        'phy:/dev/vgxen/sharedvol2,hdd,w!' ]
# Result of this config file from within the new domU:
# [root@ca-DomU ~]# sfdisk -s
# /dev/sda: 10485760
# /dev/sdb: 8388608
# /dev/sdc: 104857600
# /dev/sdd: 104857600
# For vnc setup try:
```



```

vfb = [ "type=vnc,vncunused=1,vnclisten=0.0.0.0" ]
# Example with a passwd of "foo".
#vfb = [ "type=vnc,vncunused=1,vnclisten=0.0.0.0,vncpasswd=foo" ]
# Remember, this file is "per individual" domU
# during install you will need to change
# /etc/xen/xend-config.sxp
# (vnc-listen '127.0.0.1')
# to: (vnc-listen '0.0.0.0')
#
# then from any machine do:
# "vncviewer <your dom0 ip or hostname>"
# to see vnc console

```

C.6 Virtual Iron Migration (VHD) Configuration File Example

If you migrate a Virtual Iron virtual machine (or other VHD file), you must place all the files in the `/OVS/running_pool/domain` directory, together with a guest configuration file (`vm.cfg`). You must manually create the `vm.cfg` file. Import the virtual machine using Oracle VM Manager.

Note: Before you migrate a Virtual Iron virtual machine, export any disks from Virtual Iron's VI-Center Storage view to VHD files, and uninstall any Virtual Iron VSTools.

A sample configuration file to migrate a Virtual Iron virtual machine guest follows:

```

acpi = 1
apic = 1
builder = 'hvm'
device_model = '/usr/lib/xen/bin/qemu-dm'
disk = ['file:/OVS/running_pool/domain/vm_name.vhd,sda,w',]
kernel = '/usr/lib/xen/boot/hvmloader'
keymap = 'en-us'
memory = '2048'
on_crash = 'restart'
on_reboot = 'restart'
pae = 1
timer_mode = 1
vcpus = 2

```

Oracle VM Agent Architecture

This Appendix contains more detailed information on the architecture and deployment options for the Oracle VM Agent. It contains:

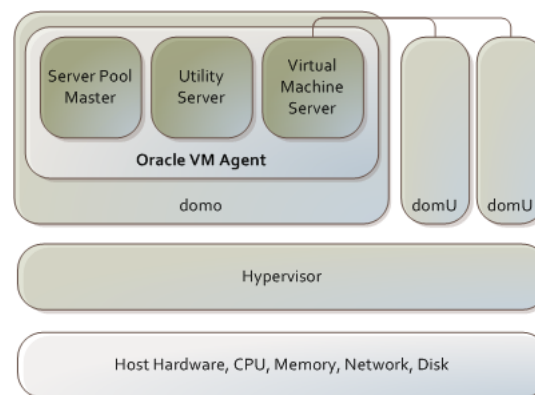
- [Oracle VM Agent Architecture](#)
- [Oracle VM Agent Deployment](#)

D.1 Oracle VM Agent Architecture

Oracle VM Agent is installed with Oracle VM Server. Oracle VM Manager manages the virtual machines running on Oracle VM Server through the Oracle VM Agent. Three types of agents are implemented:

- **Server Pool Master:** This acts as the contact point to the outside world of Oracle VM Server and dispatches to other Oracle VM Agents. It also provides virtual machine host load-balance, and local persistency of Oracle VM Server information.
- **Utility Server:** This mainly focuses on creating, removing, migrating, and so on of I/O intensive operations.
- **Virtual Machine Server:** The virtual machine server. This is the daemon for Oracle VM Server virtual machines. Virtual Machine Server can start and stop virtual guests. It also collects performance data for the host and guest operating systems. Acts as a hypervisor for domUs.

Figure D–1 Oracle VM Agent



D.2 Oracle VM Agent Deployment

Oracle VM's deployment architecture utilizes server pools, with shared access to storage across Oracle VM Servers in the server pool. Guest virtual machines are stored on the shared storage and placed on one of the Oracle VM Servers in one of two ways to balance the workloads of the server pool:

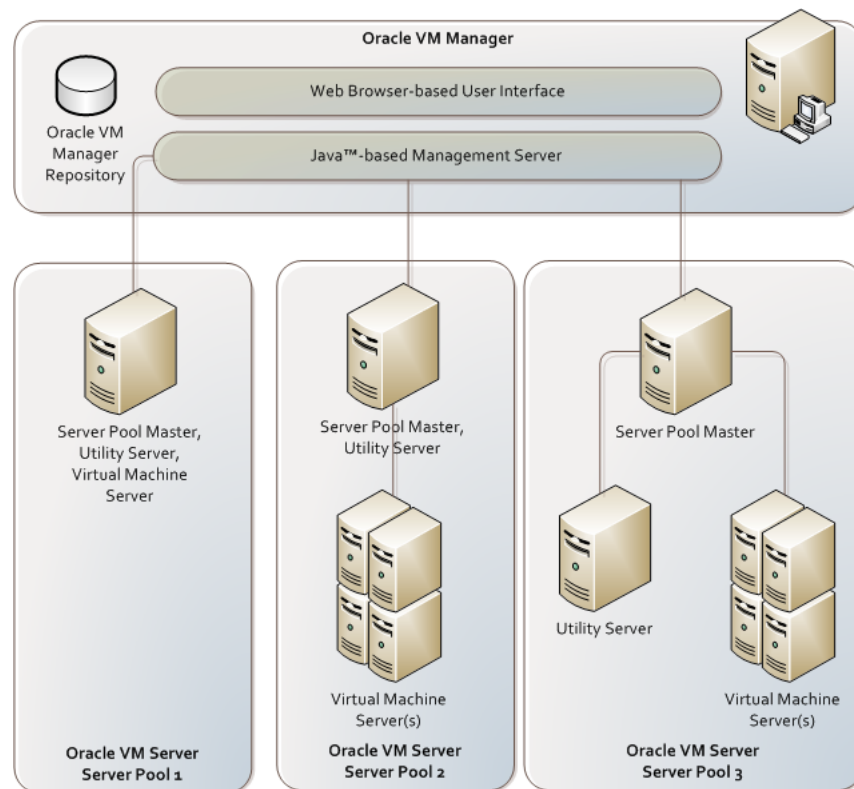
- **Auto:** Selects the Oracle VM Server in the pool with most available resources (for example, memory) to start the guest virtual machine.
- **Manual:** Allows the user to designate a pool of Oracle VM Servers as preferred servers where the guest virtual machine is allowed to start and run.

Since the guest virtual machines are not bound to any specific physical Oracle VM Server in the server pool, unless preferred servers are explicitly specified in Oracle VM Manager, guest virtual machines are not prevented from powering-on simply because an individual Oracle VM Server happens to be down for maintenance or otherwise unavailable at the time. Further, since the load-balancing algorithm assures that a guest virtual machine is placed on the Oracle VM Server with the most resources available, it also helps assure the maximum aggregate performance from the server pool.

Oracle VM Manager communicates with Oracle VM Agent to perform the management of guest virtual machines and Oracle VM Servers in server pools. There are a number of configuration options you can use when deploying Oracle VM.

- The Server Pool Master, Utility Server and Virtual Machine Server can be on the same computer.
- The Server Pool Master and Utility Server can be on the same computer, and the Virtual Machine Server can be on another computer.
- All three Oracle VM Server agent components can be on separate computers.

[Figure D-2, "Oracle VM deployment"](#) shows the deployment options for managing Oracle VM Servers.

Figure D–2 Oracle VM deployment

A server pool consists of one or more physical Oracle VM Servers, and represents a logical view of the storage where the guests reside.

In [Figure D–2, "Oracle VM deployment"](#), Server Pool 3 is deployed on individual Oracle VM Servers, while in Server Pool 1, all of the three agents are deployed on a single Oracle VM Server. Server Pool 2 shows a more typical deployment. The Server Pool Master and Utility Server are on one computer, and the Virtual Machine Server(s) are on another, or multiple other computers. This improves overall performance as guests running on the Virtual Machine Server(s) may consume a large proportion of resources, and dedicated computers are often set aside for this.

In medium- to large-scale environments with more than just a few guests in a server pool, it is recommended that the Server Pool Master and Utility Server functions reside together, or individually, on a separate and dedicated Oracle VM Server that does not host any guests, as illustrated in Server Pool 3. This is to prevent any significant Server Pool Master and Utility Server usage from impacting the performance of the workloads hosted in the guests.

The storage in [Figure D–2, "Oracle VM deployment"](#) is a mounted file system dedicated to the server pool, and stores the guests, external resources and other data files shared by Oracle VM Servers on the server pool.

The decision on how to deploy Oracle VM Agent components is made using Oracle VM Manager. You cannot configure this using Oracle VM Server. You can set up Oracle VM Manager to manage your virtual machines, in any of the configurations in [Figure D–2, "Oracle VM deployment"](#). See the *Oracle VM Manager User's Guide* for information on setting up the management of virtual machines and server pools.

Troubleshooting

This Appendix contains information on troubleshooting Oracle VM Server. It contains:

- [Debugging Tools](#)
- [Using DHCP](#)
- [Guest Console Access](#)
- [Cannot Display Graphical Installer When Creating Guests](#)
- [Hardware Virtualized Guest Devices Not Working as Expected](#)
- [Setting the Guest's Clock](#)
- [Wallclock Time Skew Problems](#)
- [Mouse Pointer Tracking Problems](#)
- [Hardware Virtualized Guest Stops](#)
- [Hardware Virtualized Guest Devices Not Working as Expected](#)
- [CD-ROM Image Not Found](#)
- [Migrating Domains](#)
- [Attaching to a Console with the Grub Boot Loader](#)

E.1 Debugging Tools

If domain creation fails, check the Oracle VM Server log files and use the command-line tools to help you find the cause of a problem. There are a number of useful command-line tools, important directories, and log files that you should check when troubleshooting problems with Oracle VM Server. This section discusses:

- Oracle VM Server directories
- Oracle VM Server log files
- Oracle VM Server command-line tools

E.1.1 Oracle VM Server Directories

The important Oracle VM Server directories you should check when troubleshooting problems with Oracle VM Server are listed in [Table E-1, "Oracle VM Server directories"](#)

Table E–1 Oracle VM Server directories

Directory	Purpose
/etc/xen	Contains Oracle VM Server configuration files for the Oracle VM Server daemon and virtualized guests.
/etc/xen/scripts	Contains networking related scripts
/var/log/xen	Contains Oracle VM Server log files.

E.1.2 Oracle VM Server Log Files

The Oracle VM Server log files you should check when troubleshooting problems with Oracle VM Server are listed in [Table E–2, "Oracle VM Server log files"](#)

Table E–2 Oracle VM Server log files

Log File	Purpose
xend.log	Contains a log of all the actions of the Oracle VM Server daemon. Actions are normal or error conditions. This log contains the same information as output using the <code>xm log</code> command.
xend-debug.log	Contains more detailed logs of the actions of the Oracle VM Server daemon.
xen-hotplug.log	Contains a log of hotplug events. Hotplug events are logged if a device or network script does not start up or become available.
qemu-dm.pid.log	Contains a log for each hardware virtualized guest. This log is created by the <code>qemu-dm</code> process. Use the <code>ps</code> command to find the <i>pid</i> (process identifier) and replace this in the file name.

E.1.3 Oracle VM Server Command-Line Tools

The Oracle VM Server command-line tools you should use when troubleshooting problems with Oracle VM Server are listed in [Table E–2, "Oracle VM Server log files"](#).

Table E–3 Oracle VM Server command-line tools

Command-Line Tool	Purpose
<code>xen top</code>	Displays real-time information about Oracle VM Server and domains.
<code>xm dmesg</code>	Displays log information on the hypervisor.
<code>xm log</code>	Displays log information of the Oracle VM Server daemon.

E.2 Using DHCP

It is recommended that you install Oracle VM Server on a computer with a static IP address. If your computers uses DHCP you should configure your DHCP server to assign static DHCP addresses. This makes sure your host always receives the same IP address. The behavior of the Oracle VM Server host is undefined if used in an environment where your IP address may change due to DHCP lease expiry.

E.3 Guest Console Access

You can connect to a guest's console using Oracle VM Manager. If you do not have access to Oracle VM Manager, you can configure access to a guest's console with VNC (Virtual Network Computing). VNC access to guests requires that VNC access is

enabled in the guest's configuration file, `vm.cfg`. Some VNC parameters (for example, the listening address and password) can be configured in one, either, or both of the following locations:

- The Oracle VM Server configuration file:
`/etc/xen/xend-config.sxp`
- The guest configuration file in either of the following locations:
`/etc/xen/name`
`/OVS/running_pool/name/vm.cfg`

Hardware virtualized guests use the `vnc=1` parameter in the guest configuration file, for example

```
vnc=1
vnclisten '0.0.0.0'
```

Paravirtualized guests use the VNC virtual frame buffer in the guest configuration file, for example

```
vfb = ['type=vnc,vncunused=1,vnclisten=0.0.0.0,vncpasswd=mypassword']
```

VNC settings defined in the guest configuration file override the settings in the Oracle VM Server configuration file. For example, if the following is specified in a hardware virtualized guest configuration file:

```
vnc=1
vnclisten '0.0.0.0'
vncpasswd 'mypassword'
```

The values set in the guest configuration file are used for VNC access, rather than any corresponding values set in the Oracle VM Server configuration.

Note: Setting `vnclisten` to `0.0.0.0` sets VNC to allow access to any computer. This may compromise security on the host computer.

If the following is specified in a hardware virtualized guest configuration file:

```
vnc=1
```

VNC is enabled in the guest, and the `vnclisten` parameter is used from the Oracle VM Server configuration file. If `vnclisten` is not specified in the Oracle VM Server configuration file, a default value of `127.0.0.1` is used. If the following is specified in the hardware virtualized guest configuration file:

```
vnc=0
```

VNC access to the guest is disabled.

Setting the default configuration options for VNC access in the Oracle VM Server configuration file enables you to configure access for all guests, and then individually override VNC access by setting the VNC parameters in the guest configuration file.

The following example is a VNC configuration entry in a paravirtualized guest configuration file:

```
vfb = ['type=vnc,vncunused=1,vnclisten=0.0.0.0,vncpasswd=mypassword']
```

The following example is a VNC configuration entry in a hardware virtualized guest configuration file:

```
vnc = 1                # vnc=1 enabled, 0=disabled
vncconsole = 1         # vncconsole=1 enables spawning VNC viewer for domain's
                        # console. Default=0
vnclisten = 0.0.0.0    # Address that should be listened on for the VNC server
                        # if VNC is set. Default (if vnc=0) is to use
                        # 'vnc-listen' setting from /etc/xen/xend-config.sxp
vncpasswd = 'mypasswd' # VNC password
vncunused = 1          # vncunused=1 - find an unused port for the VNC server
                        # to listen on. Default=1
```

In this example, the `vncunused=1` parameter allocates a new VNC port number each time a guest is created and assigns it to the guest. Port numbers are allocated starting at the default VNC port number of 5900, so `dom1` is allocated port 5900, `dom2` is allocated port 5901, `dom3` port 5902, and so on.

Connect to the guest on the host computer with the command

```
# vncviewer -Shared ipaddress:port
```

The `-Shared` parameter enables you to share the VNC connection. If you do not include this parameter, another user may destroy your VNC session if they connect at the same time. Connect from a remote computer with a VNC viewer using the connection string:

```
ipaddress:port
```

In both examples, *ipaddress* is the IP address or hostname of the Oracle VM Server, and *port* is the VNC port number of the guest.

E.4 Cannot Display Graphical Installer When Creating Guests

If the graphical installer does not start when creating a guest using the `virt-install` command-line tool, you should check your X11 configuration. If you are using a console through an `ssh` (Secure Shell) connection, connect to the console and set the `DISPLAY` environment variable, for example

```
# ssh root@example
# export DISPLAY=example:0.0
```

Alternatively, you can enable connect to a console and enable `ssh` forwarding using the `ssh -X` command, for example

```
# ssh -X root@example
```

If you use Putty to connect to a console, you must connect from an X11 capable operating system.

E.5 Hardware Virtualized Guest Console Not Displayed

If a console is not displayed after you create a hardware virtualized guest, your disk device specification may be incorrect. When you create a hardware virtualized guest, you must specify the VNC console setup. This is not required for a paravirtualized guest.

E.6 Setting the Guest's Clock

Paravirtualized guests may perform their own system clock management, for example, using the NTPD (Network Time Protocol daemon), or the hypervisor may perform system clock management for all guests.

You can set paravirtualized guests to manage their own system clocks by setting the `xen.independent_wallclock` parameter to 1 in the `/etc/sysctl.conf` file. For example

```
"xen.independent_wallclock = 1"
```

If you want to set the hypervisor to manage paravirtualized guest system clocks, set `xen.independent_wallclock` to 0. Any attempts to set or modify the time in a guest will fail.

You can temporarily override the setting in the `/proc` file. For example

```
"echo 1 > /proc/sys/xen/independent_wallclock"
```

Note: This setting does not apply to hardware virtualized guests.

E.7 Wallclock Time Skew Problems

Oracle VM Release 2.1.1 introduces the use of the `timer_mode` parameter for hardware virtualized guests. This parameter, when properly applied, can reduce or even eliminate problems with wallclock time skew in most hardware virtualized guests. Wallclock time skew problems do not occur in paravirtualized guests.

Since the application of the correct value of the `timer_mode` parameter can be difficult to determine, you can pass the `os-type` and `os-variant` command-line switches to `virt-install` to select the best `timer_mode` value for the guest operating system. When you use these `virt-install` parameters, the correct `timer_mode` value is automatically added to the guest configuration file. For example, to create an Oracle Enterprise Linux 5 64-bit guest, add the following to the `virt-install` command-line:

```
# virt-install --hvm ... --os-type=linux --os-variant=el5_64 ...
```

For best results, additional parameters may be needed in the boot loader (`grub.conf`) configuration file for certain operating system variants after the guest is installed. Specifically, for optimal clock accuracy, Linux guest boot parameters should be specified to ensure that the *pit* clock source is utilized. Adding `clock=pit nohpet nopmtimer` for most guests will result in the selection of *pit* as the clock source for the guest. Published templates for Oracle VM will include these additional parameters.

Proper maintenance of virtual time can be tricky. The various parameters provide tuning for virtual time management and supplement, but do not replace, the need for an *ntp* time service running within guest. Ensure that the *ntpd* service is running and that the `/etc/ntpd.conf` configuration file is pointing to valid time servers.

E.8 Mouse Pointer Tracking Problems

If your mouse pointer fails to track your cursor in a VNC Viewer session in a hardware virtualized guest, add the following to the Oracle VM Server configuration file located at `/etc/xen/xend-config.sxp` to force the device model to use absolute (tablet) coordinates:

```
usbdevice='tablet'
```

Restart Oracle VM Server for the changes to take effect.

E.9 Hardware Virtualized Guest Stops

When running hardware virtualized guests, the QEMU process (`qemu-dm`) may have its memory usage grow substantially, especially under heavy I/O loads. This may cause the hardware virtualized guest to stop as it runs out of memory. If the guest is stopped, increase the memory allocation for `dom0`, for example from 512MB to 768MB.

E.10 Hardware Virtualized Guest Devices Not Working as Expected

Some devices, such as sound cards, may not work as expected in hardware virtualized guests. In a hardware virtualized guest, a device that requires physical memory addresses instead uses virtualized memory addresses, so incorrect memory location values may be set. This is because DMA (Direct Memory Access) is virtualized in hardware virtualized guests.

Hardware virtualized guest operating systems expect to be loaded in memory starting somewhere around address 0 and upwards. This is only possible for the first hardware virtualized guest loaded. Oracle VM Server virtualizes the memory address to be 0 to the size of allocated memory, but the guest operating system is actually loaded at another memory location. The difference is fixed up in the shadow page table, but the operating system is unaware of this.

For example, a sound is loaded into memory in a hardware virtualized guest running Windows at an address of 100MB may produce garbage through the sound card, instead of the intended audio. This is because the sound is actually loaded at 100MB *plus* 256MB. The sound card receives the address of 100MB, but it is actually at 256MB.

An IOMMU (Input/Output Memory Management Unit) in the computer's memory management unit would remove this problem as it would take care of mapping virtual addresses to physical addresses, and enable hardware virtualized guests direct access to the hardware.

E.11 CD-ROM Image Not Found

If you create a paravirtualized or hardware virtualized guest using a configuration file, and the CDROM image cannot be found during the installation, you may have the IDE devices in the incorrect order. Putting the IDE devices in order fixes this problem. Check that the `disk = [...]` parameter is defined as `hdc:cdrom` and is included *before* `hda`, otherwise the usual `boot='dc'` configuration fails to find the CDROM image.

E.12 Firewall Blocks NFS Access

Oracle VM Server blocks NFS access from any external computer (or guest) by default. This may cause problems when trying to create a guest using an NFS connection. To resolve this, disable the firewall with the following command:

```
# service iptables stop
```

E.13 Migrating Domains

You cannot migrate domains on computers with hardware that is not identical. To migrate a domain, you must have hardware that is the same make and model. You must also have the same Oracle VM Server release.

E.14 Attaching to a Console with the Grub Boot Loader

Tracking down startup problems with a hardware virtualized guest may be difficult because you may not be able to attach a console using the `xm console` command. To workaroud this problem, you can include a console in the guest's Grub boot loader, and connect to a console during boot.

To include a console in the Grub boot loader, add the following lines before the first "title ..." line in the `/etc/grub.conf` file:

```
serial --unit=0 --speed=9600 --word=8 --parity=no --stop=1
terminal --timeout=10 serial console
```

Glossary

Domain

A configurable set of resources, including memory, virtual CPUs, network devices and disk devices, in which virtual machines run. A domain is granted virtual resources and can be started, stopped and restarted independently.

See also [dom0](#) and [domU](#).

dom0

An abbreviation for *domain zero*. The management domain with privileged access to the hardware and device drivers. Dom0 is the first domain started by the Oracle VM Server at boot time. Dom0 has more privileges than domU. It can access the hardware directly and can manage the device drivers for other domains. It can also start new domains.

domU

An unprivileged domain with no direct access to the hardware or device drivers. Each domU is started by Oracle VM Server in dom0. The xm command-line tool is used to interact with each domU.

Guest

A guest operating system that runs within a domain in Oracle VM Server. A guest may be paravirtualized or hardware virtualized. Multiple guests can run on the same Oracle VM Server.

Hardware virtualized machine

A virtual machine with an unmodified guest operating system. It is not recompiled for the virtual environment. There may be substantial performance penalties running as a hardware virtualized guest. Enables Microsoft Windows™ operating system to be run, and legacy operating systems. Hardware virtualization is only available on Intel VT or AMD SVM CPUs.

Host computer

The physical computer on which Oracle VM Server is installed.

Hypervisor

The hypervisor, monitor, or Virtual Machine Manager (VMM). It is the only fully privileged entity in the system. It controls only the most basic resources of the system, including CPU and memory usage, privilege checks, and hardware interrupts.

Management domain

See [dom0](#).

Oracle VM Agent

An application installed with Oracle VM Server. It communicates with Oracle VM Manager for management of virtual machines. Oracle VM Manager manages the virtual machines running on Oracle VM Server by communicating with Oracle VM Agent. It contains three components: Server Pool Master, Utility Server, and Virtual Machine Server.

Oracle VM Server

A self-contained virtualization environment designed to provide a lightweight, secure, server-based platform for running virtual machines. Oracle VM Server is based upon an updated version of the Xen hypervisor technology. Includes Oracle VM Agent to enable communication with Oracle VM Manager.

Oracle VM Manager

Provides the user interface, which is a standard ADF (Application Development Framework) web application, to manage Oracle VM Server pools. Manages virtual machine lifecycle, including creating virtual machines from templates or from installation media, deleting, powering off, uploading, deployment and live migration of virtual machines. Manages resources including ISO files, templates and shared virtual disks. Also provides an API via a web service to Oracle VM Server.

Paravirtualized machine

A virtual machine with a kernel that is recompiled to be made aware of the virtual environment. Runs at near native speed, with memory, disk and network access optimized for maximum performance.

Preferred Server

A Virtual Machine Server that provides resources such as memory, CPU, network interface cards (NICs), and disk to the virtual machine. If you select only one Virtual Machine Server as the preferred server, the virtual machine always starts from and runs on this server. If you select multiple preferred servers, each time the virtual machine starts, it runs on the machine with the maximum available resources.

QEMU

Also referred to as qemu-dm, which is the process name. The virtualization process which allows full virtualization of a PC system within another PC system.

Server Pool

Logically an autonomous region that contains one or more physical Oracle VM Servers. Presents a unified view of the storage where the virtual machines reside, and groups the users of these virtual machines into a single community called a *group*, in which each user is a server pool member.

Server Pool Master

A component of Oracle VM Agent. An application that acts as the contact point to Oracle VM Manager, and to other Oracle VM Agents. Provides virtual machine host load-balancing, and local persistency for Oracle VM Server.

There is only one Server Pool Master in a server pool. A physical server can perform as the Server Pool Master, Utility Server and Virtual Machine Server simultaneously.

Utility Server

A component of Oracle VM Agent. An application that handles I/O intensive operations for virtual machines, server pools and servers, for example, copying, moving and renaming files.

There can be more than one Utility Server in a server pool. A physical server can perform as the Server Pool Master, Utility Server and Virtual Machine Server simultaneously.

vif

A virtual network interface for bridging network interfaces between domUs and dom0. When a domU is started it is assigned a number. This number is used to bridge the network interface from `ethn` to `vifn.0`.

Virtual disk

A file or set of files, usually on the host file system although it may also be a remote file system, that appears as a physical disk drive to the guest operating system.

Virtual Machine (VM)

A guest operating system and the associated application software that runs within Oracle VM Server. May be paravirtualized or hardware virtualized machines. Multiple virtual machines can run on the same Oracle VM Server.

Virtual Machine Manager (VMM)

See [Hypervisor](#).

Virtual Machine Server

A component of Oracle VM Agent. An application which runs Oracle VM Server virtual machines. It can start and stop virtual machines, and collect performance data for the host and guest operating systems. Enables communication between the Server Pool Master, Utility Server and Virtual Machine Servers.

There can be more than one Virtual Machine Server in a server pool. A physical server can perform as the Server Pool Master, Utility Server and Virtual Machine Server simultaneously.

Virtual Machine Template

A template of a virtual machine. Contains basic configuration information such as the number of CPUs, memory size, hard disk size, and network interface card (NIC). Create virtual machines based on a virtual machine template using Oracle VM Manager.

VMM

See [Virtual Machine Manager \(VMM\)](#).

Xen™

The Xen hypervisor is a small, lightweight, software virtual machine monitor, for x86-compatible computers. The Xen hypervisor securely executes multiple virtual machines on one physical system. Each virtual machine has its own guest operating system with almost native performance. The Xen hypervisor was originally created by researchers at Cambridge University, and derived from work done on the Linux kernel.

Index

A

Agent
 VM Server, D-1
Anaconda installation tree, 4-5

C

Clock, E-5
 Setting, E-5
Command line tools, E-2
Convert host to guest, A-12
Converting hardware virtualized guest to
 paravirtualized guest, 4-11
Converting Linux hosts, 10-1
Converting VMWare Virtual Machines, 10-1

D

DHCP, E-2
Disk Priority, C-2
dom0, Glossary-1
 Explanation, 2-3
Domain, Glossary-1
domU, Glossary-1

E

e100/e1000 network driver, C-1
External storage, 8-2

F

Firewall
 Stopping, E-6

G

Gigabit network driver, C-1
Guest, Glossary-1
 Configuration, 4-8
 Lifecycle, 5-1
 Management, 5-1, D-2
 Monitoring, 5-1
 Supported operating systems, 4-1
Guest clock
 Setting, E-5

Guest operating system, Glossary-1

H

HA, 7-1, 8-1
Hardware virtualization
 Overview, 2-2
Hardware virtualized guest
 Creating, 4-9
 Creating using virt-install, 4-3
Hardware virtualized machine, Glossary-1
High Availability, 7-1, 8-1
Host
 Monitoring, 5-2
Host computer, Glossary-1
Hypervisor, 2-2, Glossary-1

I

Import a virtual machine, C-5
Installation source screen, A-12
Installation tree, 4-5
Intel e100/e1000 network drivers, C-1
iso_pool, 9-1

K

Keyboard selection screen, A-12
Kickstart file, A-12

L

Language selection screen, A-12
Linux host conversion, 10-1
linux p2v, 10-1
Log
 Oracle VM Agent, 3-1
 Oracle VM Server, E-2

M

Management domain, 2-3, Glossary-1

N

Network driver, C-1
Network Traffic Priority, C-2

NFS access blocked, E-6
NIC, C-1

O

OCFS2, 8-1
/opt/ovs-agent-2.3/utls/repos.py script, 9-2
Oracle Enterprise Linux 4 Update 4
 Converting hardware virtualized guest to
 paravirtualized guest, 4-11
Oracle VM, 1-2
 Overview, 1-3
Oracle VM Agent, 1-3, 3-1, A-2, D-1, Glossary-2
 Configuration, 3-1
 default user, 3-1
 Install location, 3-1
 Install log, 3-1
 Log, 3-1
 Monitoring, 3-3
 Server Pool Master, D-1
 Starting and stopping, 3-3
 Virtual Machine Server, D-1
Oracle VM Manager, 1-2, Glossary-2
Oracle VM Server, Glossary-2
 API, B-2
 API options, B-2
 Components, 2-1
 Configuration file, B-2
 Configuration options, B-3
 Create guest, A-3
 Deployment, D-2
 Log file, B-2
 Logfile options, B-2
 Management, A-7
 Repository, 9-1
 VNC Server options, B-5
os-type, E-5
os-variant, E-5
/OVS directory, 9-1
ovsagent, A-2
 Command line tool, 3-1

P

P2V, 10-1, A-12
P2V kickstart file, A-12
P2V network configuration screen, A-12
Paravirtualization
 Overview, 2-2
Paravirtualized guest
 Converting Oracle Enterprise Linux 4 Update 4
 guest, 4-11
 Creating, 4-6
 Creating using virt-install, 4-3
Paravirtualized machine, Glossary-2
Preferred Server, Glossary-2
publish_pool, 9-1

Q

QEMU, E-6, Glossary-2

QoS, C-2
Quality of Service, C-2

R

Repository, 9-1
 Adding a repository, 9-1
 Mount options, 9-1
 Removing, 9-1
repos.py script, 9-2
running_pool, 9-1

S

seed_pool, 9-1
Server Pool, Glossary-2
Server Pool Master, Glossary-2
Server Pool Master Agent, D-1
Shared storage, 6-1, 7-4
sharedDisk, 9-1
Storage, 8-2
Suppress P2V screens, A-12
System Clock
 Setting, E-5

T

timer_mode, E-5

U

Utility Server, Glossary-3
Utility Server Agent, D-1

V

/var/ovs/mount/uuid directory, 9-1
VHD virtual machines
 vm.cfg example, C-5
vif, C-1, Glossary-3
virt-install, A-3
 Command line tool, 4-1
Virtual disk, Glossary-3
Virtual Iron virtual machine
 import, C-5
 vm.cfg, C-5
Virtual Machine, Glossary-3
Virtual machine management, D-2
Virtual Machine Manager, Glossary-3
Virtual Machine Server, D-1, Glossary-3
Virtual Machine Server Agent, D-1
Virtual machine template, Glossary-3
Virtual Network Interface, Glossary-3
VM, Glossary-3
VM management, D-2
VM Server, Glossary-3
VM Server Agent, D-1
vm.cfg
 Virtual Iron example, C-5
VMM, Glossary-3
VMware virtual machine conversion, 10-1

VNC access to guests, E-2

W

Wallclock Time Skew, E-5

X

Xen, Glossary-3

Xen hypervisor, 1-2, Glossary-3

xend-config.sxp configuration file, B-2

xen.independent_wallclock

Setting, E-5

xm, A-7

xm command, 5-1

xm top command, 5-2

XML-RPC interface, B-2

