Oracle® VM

Server User's Guide

Release 2.1.1

E10898-02

February 2008

ORACLE®

Oracle VM Server User's Guide, Release 2.1.1

E10898-02

Primary Author: Alison Holloway

Contributing Authors: Kurt Hackel, Herbert van den Bergh

Contributor: Tatyana Bagerman, Michael Chan

Contents

4   Creating a Guest

5   Domain Monitoring and Administration

6   Domain Live Migration

7   Managing Oracle VM Server Repositories

A   Command-Line Tools

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 wish to learn the fundamentals of virtualization and the provision of virtual guest operating systems.

Documentation Accessibility

Our goal is to make Oracle products, services, and supporting documentation accessible, with good usability, to the disabled community. To that end, our documentation includes features that make information available to users of assistive technology. This documentation is available in HTML format, and contains markup to facilitate access by the disabled community. Accessibility standards will continue to evolve over time, and Oracle is actively engaged with other market-leading technology vendors to address technical obstacles so that our documentation can be accessible to all of our customers. For more information, visit the Oracle Accessibility Program Web site at http://www.oracle.com/accessibility/.

Accessibility of Code Examples in Documentation

Screen readers may not always correctly read the code examples in this document. The conventions for writing code require that closing braces should appear on an otherwise empty line; however, some screen readers may not always read a line of text that consists solely of a bracket or brace.

Accessibility of Links to External Web Sites in Documentation

This documentation may contain links to Web sites of other companies or organizations that Oracle does not own or control. Oracle neither evaluates nor makes any representations regarding the accessibility of these Web sites.

TTY Access to Oracle Support Services

Oracle provides dedicated Text Telephone (TTY) access to Oracle Support Services within the United States of America 24 hours a day, 7 days a week. For TTY support, call 800.446.2398. Outside the United States, call +1.407.458.2479.

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/rdsk/c0t1d0s6 of=/dev/rst0 bs=10b \` `count=10000`
braces { }	Braces indicate required items: `.DEFINE {macro1}`
brackets [ ]	Brackets indicate optional items: `cvtcrt termname [outfile]`
ellipses ...	Ellipses indicate an arbitrary number of similar items: `CHKVAL fieldname value1 value2 ... valueN`
italics	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 2.1.1 documentation set:

- *Oracle VM 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*

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.
italic	Italic type indicates book titles, emphasis, or placeholder variables for which you supply particular values.
`monospace`	Monospace type indicates commands within a paragraph, URLs, code in examples, text that appears on the screen, or text that you enter.

x

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 as well as 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. They cannot access each other, so 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 from installation media or from a virtual machine template

  – Delete virtual machines

  – Power off virtual machines

  – Upload virtual machines

  – Deploy virtual machines

  – Perform live migration of virtual machines

  – Import and manage ISOs

  – Create and manage virtual machine templates

  – Create and manage sharable hard disks

- **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. It communicates with Oracle VM Manager for management of virtual machines.

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

Figure 1–1   Oracle VM Architecture
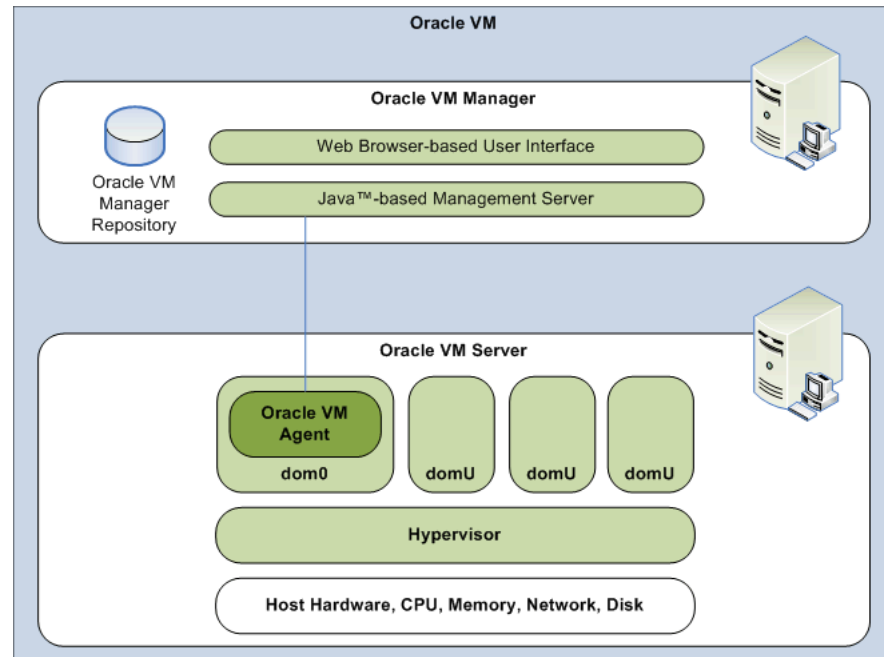
This book discusses Oracle VM Server, and the Oracle VM Agent. See the *Oracle VM Manager Installation Guide* and the *Oracle VM Manager User's Guide* for information on installing, and using Oracle VM Manager, and managing Oracle VM Servers.

2

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
- Hardware Virtualization Vs. Paravirtualization
- 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, Solaris or Microsoft Windows. Figure 2–1, "Oracle VM Server" shows the components of Oracle VM Server.

Figure 2–1   Oracle VM Server
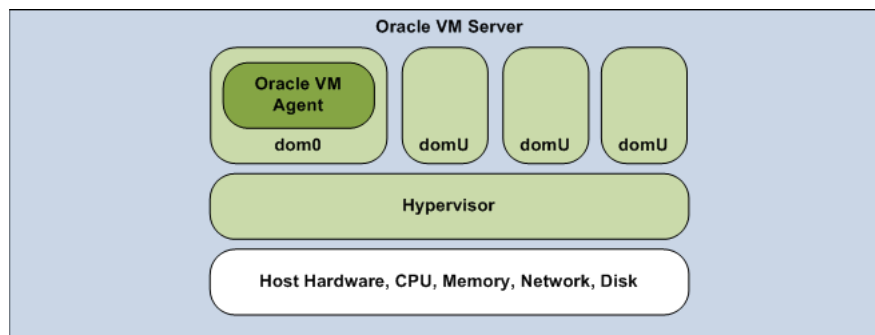
2.2 Hypervisor

Oracle VM Server is architected such that the hypervisor (or monitor, or Virtual Machine Manager) is the only fully privileged entity in the system, but is also extremely small and tightly written. 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 rebooted 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 operating system guest kernel 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
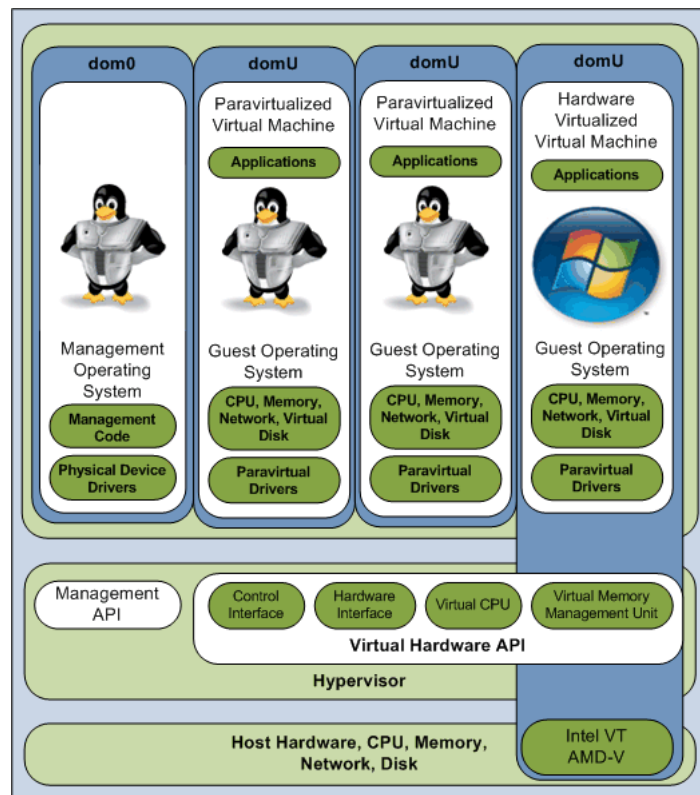
If support for hardware virtualization is available (either Intel VT or AMD SVM), the guest operating system may run completely unmodified. This hardware virtualized fully-virtualized guest is carefully monitored and trapped by Oracle VM Server when

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 Hardware Virtualization Vs. Paravirtualization

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" 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 7, "Managing Oracle VM Server Repositories" for more information.

3

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-2.2
```

Installation logs are located in:

```
/opt/ovs-agent-2.2/logs/ovs_trace.log
```

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

- Deploying Oracle VM Agent

- Oracle VM Agent API

- Oracle VM Agent Utilities

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 allows 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 root, run the Oracle VM Agent configuration script:

```
service ovs-agent configure
```

2. You are prompted whether you want to enable SSL (Secure Sockets Layer) in XMLRPC.

```
;ssl support --
;enable ssl support in xmlrpc transport?(enable/disable)
ssl=disable
ssl
```

SSL support is disabled by default. Enter `enable` to enable support, or `disable` to disable it. If you do not enter anything, the setting is not changed. Press **Enter**.

3. 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
;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 all computers in the domain example.com, enter

```
*.example.com
```

Press **Enter**.

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

```
now allow=*.example.com

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**.

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

```
now deny=

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 Oracle VM Server or the guests running on it.

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.

3.6 Deploying Oracle VM Agent

You can use Oracle VM Manager to configure the deployment of Oracle VM Agent. See the *Oracle VM Manager User's Guide* for information on configuring Oracle VM Agent with Oracle VM Manager.

3.7 Oracle VM Agent API

An application programming interface (API) is available for Oracle VM Agent. You can use the Oracle VM Agent API to perform a number of actions on guests and on the Oracle VM Server. The Oracle VM Agent API uses the XML-RPC protocol. To see a list of the XML-RPC functions available, enter the following URL in a web browser:

```
https://hostname:8899/
```

The `hostname` refers to an Oracle VM Server. A list of methods and parameters are displayed.

See the XML-RPC protocol specification for more information on how to use XML-RPC.

The following sections give examples of common Oracle VM Agent API functions. The examples are not extensive, but serve to give you an idea of how to use the API.

- Testing the Oracle VM Agent Connection
- Remotely Starting a Guest with the Oracle VM Agent API
- Remotely Stopping a Guest with the Oracle VM Agent API

> **Note:** The following sections discuss commands that must be run from a computer on which Oracle VM Server is installed. The functionality discussed is to be used to communicate from one Oracle VM Server to another Oracle VM Server.

3.7.1 Testing the Oracle VM Agent Connection

You can use the `do_rpc.py` utility to remotely test the connection to Oracle VM Agent from another computer with Oracle VM Server installed. The following command tests the connection to Oracle VM Agent:

```
# /opt/ovs-agent-2.2/utils/do_rpc.py https://username:password@hostname:8899 echo
hello
```

Where `username` and `password` refer to the Oracle VM Agent login credentials, and `hostname` refers to the host name of the computer on which Oracle VM Agent is installed. If the connection is successful, the following is displayed:

```
echo ['hello']
=> success:echo=hello
```

If this command does not display any echo information, or displays an error, check your network connection. The following error is displayed if the Secure Socket Layer (SSL) facility is disabled.

```
socket error! code=1, msg=error:140770FC:SSL routines:SSL23_GET_SERVER_
HELLO:unknown protocol
```

To enable SSL, edit the `/etc/ovs-agent/agent.ini` file and set the `ssl` parameter to `enable`.

```
ssl=enable
```

Restart the Oracle VM Agent for the changes to take effect:

```
# /sbin/service ovs-agent restart
```

3.7.2 Remotely Starting a Guest with the Oracle VM Agent API

You can use the `do_rpc.py` utility to remotely start a guest on an Oracle VM Server from another computer with Oracle VM Server installed. The following command remotely starts a guest on an Oracle VM Server:

```
# /opt/ovs-agent-2.2/utils/do_rpc.py https://username:password@hostname:8899
start_vm /OVS/running_pool/guest
```

Where `username` and `password` refer to the Oracle VM Agent login credentials, `hostname` refers to the host name of the computer on which Oracle VM Agent is installed, and `guest` refers to the name of the guest. If the guest starts without error, the following is displayed:

```
start_vm ['/OVS/running_pool/guest']
 =>success=ip=10.1.1.1
```

3.7.3 Remotely Stopping a Guest with the Oracle VM Agent API

You can use the `do_rpc.py` utility to remotely stop a guest on an Oracle VM Server from another computer with Oracle VM Server installed. The following command remotely stops a guest on an Oracle VM Server:

```
# /opt/ovs-agent-2.2/utils/do_rpc.py https://username:password@hostname:8899 stop_
vm /OVS/running_pool/guest
```

Where `username` and `password` refer to the Oracle VM Agent login credentials, `hostname` refers to the host name of the computer on which Oracle VM Agent is installed, and `guest` refers to the name of the guest. If the guest stops without error, the following is displayed:

```
stop_vm ['/OVS/running_pool/guest']
 =>success
```

3.8 Oracle VM Agent Utilities

A number of utility scripts are included with Oracle VM Agent, and you can use these to monitor and manage guests. The utility scripts discussed in this section enable you to perform actions such as:

- Monitoring a Guest with the Oracle VM Agent
- Deleting a Guest with the Oracle VM Agent

3.8.1 Monitoring a Guest with the Oracle VM Agent

You can use the `db_kfv.py` utility to monitor a guest. To check the status of a guest, enter:

```
# /opt/ovs-agent-2.2/db/db_kfv.py vm guest status
```

The status of the guest is displayed. For example, if the guest is not running, the following is displayed:

```
/OVS/running_pool/guest  status(now)=>DOWN
```

3.8.2 Deleting a Guest with the Oracle VM Agent

You can use the `db_del.py` utility to delete a guest. To delete a guest, enter:

```
# /opt/ovs-agent-2.2/db/db_del.py database guest
```

For example, to delete a guest named *myguest*, enter:

```
# /opt/ovs-agent-2.2/db/db_del.py vm myguest
```

The entries in /OVS/running_pool/myguest are removed and the guest deleted.

4

Creating a Guest

This Chapter contains information on how to create a guest operating system. You can create paravirtualized guests and hardware virtualized guests using a guest template, or using the command-line tool virt-install. virt-install 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 on Oracle VM Server using Oracle VM Manager. See the *Oracle VM Manager User's Guide* for information on creating guests with Oracle VM Manager.

This Chapter contains:

- Supported Guest Operating Systems

- Mounting an ISO

- 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

The guest you create must be one of the following configurations.

Table 4–1   64-bit Hypervisor Supported Guest Operating Systems

Guest Operating Systems	Paravirtualized 32-bit	Paravirtualized 64-bit	Hardware Virtualized 32-bit	Hardware Virtualized 64-bit
RedHat Enterprise Linux 3.x	Yes	No	Yes	Yes

Table 4–1   (Cont.) 64-bit Hypervisor Supported Guest Operating Systems

Guest Operating Systems	Paravirtualized 32-bit	Paravirtualized 64-bit	Hardware Virtualized 32-bit	Hardware Virtualized 64-bit
Oracle Enterprise Linux Release 4.x	Yes	Yes	Yes	Yes
RedHat Enterprise Linux 4.x				
Oracle Enterprise Linux Release 5.x	Yes[1]	Yes	Yes	Yes
RedHat Enterprise Linux 5.x				

[1]   The xm save and xm restore commands are not supported.

Oracle Enterprise Linux kernels are available on the Oracle VM Server CD for Release 3 Update 9, and Release 4 Update 4 onwards. The kernels are also available on the Oracle Enterprise Linux Network (ULN).

Table 4–2   32-bit Hypervisor Supported Guest Operating Systems

Guest Operating Systems	Paravirtualized 32-bit	Hardware Virtualized 32-bit
RedHat Enterprise Linux 3.x	Yes	Yes
Oracle Enterprise Linux Release 4.x	Yes	Yes
RedHat Enterprise Linux 4.x		
Oracle Enterprise Linux Release 5.x	Yes	Yes
RedHat Enterprise Linux 5.x		

The following guest operating systems have been tested and no issues have been found other than as stated in the Oracle VM documentation.

Table 4–3   Tested Guest Operating Systems

Guest Operating Systems	Hardware Virtualized 32-bit	Hardware Virtualized 64-bit
Microsoft Windows™ 2000	Yes	Yes
Microsoft Windows™ 2003	Yes	Yes
Microsoft Windows™ XP Pro	Yes	Yes
Microsoft Windows™ Vista	Yes	Yes

There are a number of ways to create virtual guests:

- Using a template
- Using the virt-install command-line tool interactively
- Using the virt-install command-line tool and passing all parameters at the same time from the command-line
- Manually

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

4.2 Mounting an ISO

You cannot create a guest from a local hard disk or CD-ROM using the virt-install command-line tool. You can, however, create an installation tree on the host operating system, and mount it as an NFS share. For example

```
# 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 using the virt-install command-line tool, 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 server 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 using the virt-install command-line tool, enter the installation location as:

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

4.3 Enabling Registration of Guests with Oracle VM Manager

You can create a guest in Oracle VM Server and manage it using Oracle VM Manager. To add guests created with the virt-install command-line tool, or created manually, you must create the guests with the following configuration:

1. Create a directory for your guest, for example

   ```
   mkdir /OVS/running_pool/myguest
   ```

   The name of the directory should be the same as the guest name.

2. Create the disk image under the new directory.

The guest is ready to import into Oracle VM Manager.

4.4 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 Enterprise Linux template from an HTTP server, and use it to create a guest.

1. Download the template into the a directory on the Oracle VM Server computer. This example uses the directory /OVS.

   ```
   cd /OVS/seed_pool
   wget  http://example.com/seeds/OVM_EL4U5_X86_HVM.tgz
   ```

   Uncompress the template:

```
tar -xzvf OVM_EL4U5_X86_HVM.tgz
```

2. Modify the vm.cfg file to change the directories to that of your environment. The entries you should modify are `image`, `kernel` and `ramdisk`. For example

```
disk = [ 'file:/OVS/seed_pool/para_seed/system.img,hda,w', ]
kernel = '/OVS/seed_pool/para_seed/vmlinuz-2.6.9-42.32.0.0.1.ELxenU'
ramdisk = '/OVS/seed_pool/para_seed/initrd-2.6.9-42.32.0.0.1.ELxenU.img'
```

Should be modified as follows:

```
disk = [ 'file:/OVS/local/para_seed/system.img,hda,w', ]
kernel = '/OVS/local/para_seed/vmlinuz-2.6.9-42.32.0.0.1.ELxenU'
ramdisk = '/OVS/local/para_seed/initrd-2.6.9-42.32.0.0.1.ELxenU.img'
```

3. Modify the `vif` (Virtual Network Interface) setting in the vm.cfg file to remove the fixed MAC address. For example

```
vif = ['type=ioemu, mac=00:16:3e:73:a3:27, bridge=xenbr0']
```

Should be modified as follows:

```
vif = ['type=ioemu, , bridge=xenbr0']
```

Alternatively, you could modify it to have no virtual interface:

```
vif = ['']
```

4. Modify the `uuid` (Universally Unique Identifier) setting in the vm.cfg file. For example

```
uuid = "8a87a97c-65ad-f243-b373-20185c83053e"
```

Should be commented as follows:

```
# uuid = "8a87a97c-65ad-f243-b373-20185c83053e"
```

5. Modify any other settings you require in the vm.cfg file. For example, `memory` or `vcpus` (virtual CPUs).

6. To create and start the guest, use the `xm create` command:

```
xm create vm.cgf
```

The guest is created and started.

4.5 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 allows 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 to allocate 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

```
256
```

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

```
/home/myhome/el5
```

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 following question is displayed:

```
What is the install location?
```

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

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

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. However, you can set up an installation tree on the host operating system, and then export it. See Section 4.2, "Mounting an ISO".

Enter the path to an Oracle VM Server installation tree, for example

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

Press **Enter**.

The guest operating system installer starts. If you enabled graphics support in Step 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 Enterprise Linux displays:
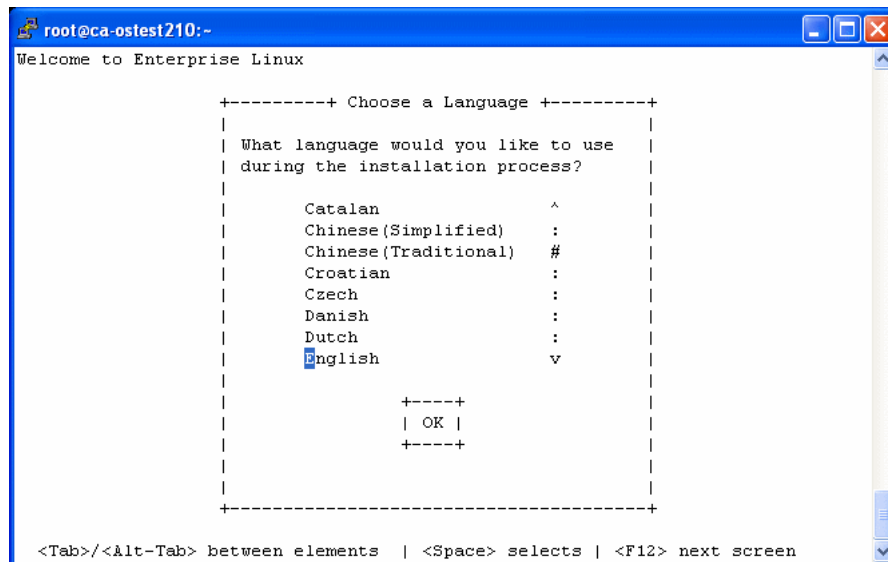
Follow the prompts to complete the guest operating system installation.

4.6 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

4.6.1 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, as well as 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 **myvmdisk1**:

       ```
       # lvcreate -L4096M -n myvmdisk1 vg
       ```

       You now have a /dev/vg/myvmdisk1. Make a file system on the partition:

       ```
       #mkfs -t ext3  /dev/vg/myvmdisk1
       ```

    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.

4.6.2 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
#unmount /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.

4.6.3 Configuring the Guest

You must modify the guest configuration files. To modify the guest configuration files:

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.

```
#unmount /mnt
```

7. Create the guest.

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

8. Get the console of the guest.

```
#xm console <Domainname>/<Domid>
```

4.7 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 Files" 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.

```
#unmount /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.

4.8 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 4 as it does not support a direct installation as a paravirtualized guest. 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 4, paravirtualized guest:

1. Copy the Oracle Enterprise-R4-U4-x86_64-dvd.iso image to the local file system of the Oracle VM Server computer

```
# ls -l /root/Enterprise-R4-U4-x86_64-dvd.iso
-rw-r--r-- 1 root root 2530611200 Aug  2 13:03 /root/Enterprise-R4-U4-x86_
64-dvd.iso
```

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

```
# lvcreate -L8G -n el4u4 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 el4u4 -f /dev/VolGroup00/el4u4 -v -c
/root/Enterprise-R4-U4-x86_64-dvd.iso -r 512 --vnc
```

If you see this error message:

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

You need to 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, reboot 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 el4u4

Using config file "/etc/xen/el4u4".
Started domain el4u4
# 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 4, which can be found in the directory extra_kernels/EL4U4PV_64 on the Oracle VM Server installation CD-ROM, to the guest:

```
# cd extra_kernels/EL4U4PV_64/
# scp kernel-xenU-2.6.9-42.32.0.0.1.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-2.6.9-42.32.0.0.1.EL.x86_64.rpm
warning: kernel-xenU-2.6.9-42.32.0.0.1.EL.x86_64.rpm: V3 DSA signature: NOKEY,
key ID b38a8516
Preparing...                ######################################### [100%]
   1:kernel-xenU            ######################################### [100%]
WARNING: No module xenblk found for kernel 2.6.9-42.32.0.0.1.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 (2.6.9-42.32.0.0.1.ELxenU)
        root (hd0,0)
        kernel /boot/vmlinuz-2.6.9-42.32.0.0.1.ELxenU ro root=LABEL=/
        initrd /boot/initrd-2.6.9-42.32.0.0.1.ELxenU.img
```

10. Shut down the guest. Modify the host configuration file /etc/xen/el4u4 to an entry similar to:

```
name = "el4u4"
memory = "512"
disk = [ 'phy:/dev/VolGroup00/el4u4,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 el4u4
```

   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/el4u4 to add a vif entry similar to:

```
name = "el4u4"
memory = "512"
disk = [ 'phy:/dev/VolGroup00/el4u4,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/el4u4, 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/el4u4
```

The guest is now ready for use.

4.9 Creating a Paravirtual Guest with Red Hat Enterprise Linux 3.8 or 3.9

A paravirtual guest cannot be created using Red Hat Enterprise Linux 3.8 or 3.9. You must first create a hardware virtualized guest, then convert it to a paravirtualized guest:

1. Create a hardware virtualized guest using Red Hat Enterprise Linux 3.8 or 3.9 as shown in Section 4.5, "Creating a Guest Using virt-install".

2. Switch the guest operating system network update server to the Oracle update server (ULN) by following the instructions at https://linux.oracle.com/switch.html.

3. Convert the hardware virtualized guest as shown in Section 4.8, "Converting a Hardware Virtualized Guest to a Paravirtualized Guest".

4.10 Installing Paravirtual Drivers

Installing Oracle Enterprise Linux Release 4 Update 4 as a guest operating system may require that you install paravirtual drivers for your hardware. This section lists the steps for installing these paravirtual drivers. To install paravirtual drivers for Oracle Enterprise Linux guest operating systems:

1. Download the paravirtual drivers from

```
http://www.oracle.com/virtualization
```

2. Untar the drivers to a temporary directory.

3. 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
```

4. Run depmod.

5. Edit the /etc/xen/xend-config.sxp file to replace the vif entry to:

```
vif = [ '', ]
```

6. Shut down the domain:

```
xm shutdown mydomain
```

7. Start the domain:

```
xm create /home/myhome/config
```

8. When prompted by kudzu, remove the old network configuration.

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

```
ifconfig eth0
```

10. Edit the /etc/xen/xend-config.sxp file to add the new MAC address:

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

11. Create 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
```

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

```
mkinitrd -f /boot/initrd-2.6.9-42.0.3.8.2.ELsmp.img 2.6.9-42.0.3.8.2.ELsmp
--omit-scsi-modules --with=xen-vbd
```

13. Reboot the computer.

5

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. See the *Oracle VM Manager User's Guide* for information on using Oracle VM Manager to manage domains.

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, 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:

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:

```
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
0000001
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
86_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--
```

6

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 create a shared virtual disk before you perform the migration. This Chapter contains:

- Creating a Shared Virtual Disk for Live Migration
- Migrating a Domain

6.1 Creating a Shared Virtual Disk for Live Migration

If you want to perform live migration of domains to other, identical, computers, you must create a shared virtual disk to be used during the live migration. You can set up a shared virtual disk in the following configurations:

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

This section discusses creating a shared virtual disk in these configurations.

6.1.1 Creating a Shared Virtual Disk Using OCFS2 on iSCSI

To create a shared virtual disk using OCFS2 on iSCSI:

1. Install the iscsi-initiator-utils RPM. The iscsi-initiator-utils RPM is available on the Oracle VM Server CDROM or ISO file.

   ```
   rpm -Uvh iscsi-initiator-utils-6.2.0.742-0.5.el5.i386.rpm
   ```

2. Start the iSCSI service:

   ```
   service iscsi start
   ```

3. 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
```

4. 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
```

5. Verify that only the necessary entries are visible:

```
iscsiadm -m node
```

6. 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
```

7. Restart the iSCSI service:

```
service iscsi restart
```

8. 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
```

9. The new device can now be used.

```
fdisk -l /dev/sdb
```

10. Create a new directory named /etc/ocfs2 directory:

```
mkdir /etc/ocfs2
```

11. Create the OCSF2 configuration file as /etc/ocfs2/cluster.conf. The following is a sample cluster.conf file:

```
node:
        ip_port = 7777
        ip_address = 10.1.1.1
        number = 0
        name = example1.com
        cluster = ocfs2
```

```
node:
        ip_port = 7777
        ip_address = 10.1.1.2
        number = 1
        name = example2.com
        cluster = ocfs2
cluster:
        node_count = 2
        name = ocfs2
```

12. Review the status of the OCFS2 cluster service:

    ```
    service o2cb status
    ```

13. Load the OCFS2 module:

    ```
    service o2cb load
    ```

14. Set the OCFS2 service to be online:

    ```
    service o2cb online
    ```

15. Configure the OCFS2 service to start automatically when the computer boots:

    ```
    service o2cb configure
    ```

16. Start up the OCFS2 service.

    ```
    service o2cb start
    ```

17. Format the shared virtual disk from any of the servers in the cluster:

    ```
    mkfs.ocfs2 /dev/sdh1
    ```

18. Mount the shared virtual disk from all the servers in the cluster on /OVS/remote:

    ```
    mount /dev/sdh1 /OVS/remote/ -t ocfs2
    ```

19. Change the /etc/fstab file to include the shared virtual disk mounted at boot:

    ```
    /dev/sdh                /OVS/remote              ocfs2   defaults        1 0
    ```

6.1.2 Creating a Shared Virtual Disk Using OCFS2 on SAN

To create a shared virtual disk 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
    ```

    Determine the share disk volume you want to use.

2. Create a new directory named /etc/ocfs2 directory:

    ```
    mkdir /etc/ocfs2
    ```

3. Create the OCSF2 configuration file as /etc/ocfs2/cluster.conf. The following is a sample cluster.conf file:

```
node:
        ip_port = 7777
        ip_address = 10.1.1.1
        number = 0
        name = example1.com
        cluster = ocfs2
node:
        ip_port = 7777
        ip_address = 10.1.1.2
        number = 1
        name = example2.com
        cluster = ocfs2
cluster:
        node_count = 2
        name = ocfs2
```

4. Review the status of the OCFS2 cluster service:

```
service o2cb status
```

5. Load the OCFS2 module:

```
service o2cb load
```

6. Set the OCFS2 service to be online:

```
service o2cb online
```

7. Configure the OCFS2 service to start automatically when the computer boots:

```
service o2cb configure
```

8. Start up the OCFS2 service.

```
service o2cb start
```

9. Format the shared virtual disk from any of the servers in the cluster:

```
mkfs.ocfs2 /dev/sdh1
```

10. Mount the shared virtual disk from all the servers in the cluster on /OVS/remote:

```
mount /dev/sdh1 /OVS/remote/ -t ocfs2
```

11. Change the /etc/fstab file to include the shared virtual disk mounted at boot:

```
/dev/sdh                /OVS/remote                ocfs2   defaults        1 0
```

6.1.3 Adding a Shared Virtual Disk Using NFS

To add a shared virtual disk using NFS:

1. Find an NFS mount point to use. This example uses the mount point:

```
mycomputer:/vol/vol1/data/ovs
```

2. Add the following entry to the /etc/fstab file:

```
myfileserver:/vol/vol1/data/ovs /OVS/remote  nfs
rw,bg,intr,hard,timeo=600,wsize=32768,rsize=32768,nfsvers=3,tcp 0 0
```

3. Mount the shared virtual disk:

```
mount /OVS/remote
```

6.2 Migrating a Domain

To migrate a domain from one computer to another identical computer:

1. Create a shared virtual disk to use during the domain migration. See Section 6.1, "Creating a Shared Virtual Disk for Live Migration". Each computer involved with the domain migration must have access to the shared virtual disk in the same way, either as an NFS or a SAN virtual disk.

2. On the Oracle VM Server that contains the existing domain, migrate the domain to to the remote computer with the following command:

```
xm migrate mydomain myremotecomputer
```

The domain is migrated to the remote computer.

7

Managing Oracle VM Server Repositories

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

- Oracle VM Server Repositories

7.1 Oracle VM Server Repositories

A repository is used for live migration of guests and for local storage.

A list of the Oracle VM Server repositories is in:

```
/etc/ovs/repositories
```

A configuration file for mounting repositories is in:

```
/etc/ovs/repositories.options
```

You can add a repository using the `ovs-makerepo` script, and remove them using the `ovs-offlinerepo -d` script.

Repositories are managed by Oracle VM Agent. If you need to manually manage the repository daemon, you can use following commands:

```
/etc/init.d/ovsrepositories [start|stop|status|restart|reload]
```

7.1.1 Repository Configuration File

A list of the Oracle VM Server repositories is in:

```
/etc/ovs/repositories
```

The format used in the repository file is:

```
uuid source
```

The `uuid` parameter is the UUID (Universally Unique Identifier) of the repository. The `source` parameter is the block device or NFS path to the file system used as a repository.

7.1.2 Repository Mount Options

You can create an optional repository option file which contains mount options for storage repositories. The repository option file must be named:

```
/etc/ovs/repositories.options
```

The format of this file is:

uuid options

The `uuid` parameter is the UUID (Universally Unique Identifier) of the repository and must match the UUID of the repository in the /etc/ovs/repositories file. The `options` parameter is list of mount options, as they would appear if the volume were mounted with `mount -o opt1,opt2,opt3`. The mount options will be used exactly as listed in the file until the end of the line.

7.1.3 Adding a Repository

You can add a file system or shared virtual disk as a repository using the `makerepo` script. The script takes the parameters:

/usr/lib/ovs/ovs-makerepo *source shared description*

The `source` parameter is the block device or NFS path to the file system to be added. The `shared` parameter sets whether the file system is to be shared between computers. Enter a `1` to have the file system shared, or a `0` to make it non-shared. The `description` parameter is the text displayed in Oracle VM Manager.

The `makerepo` script identifies the file system or shared virtual disk as a repository and updates the repository configuration to enable it.

7.1.4 Removing a Repository

You can remove a repository using the `ovs-offlinerepo` script. The script takes the parameters:

/usr/lib/ovs/ovs-offlinerepo [-d] *uuid source*

The `-d` flag removes the storage repository from the configuration file. The `uuid` parameter is the UUID of the repository. The `source` parameter is the block device or NFS path to the file system to be removed.

The `ovs-offlinerepo` script unmounts a repository and removes it from the configuration.

A

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

ovs-agent

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

```
ovs-agent {option}
```

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" discusses using the virt-install tool.

Options

[-h | --help]
Displays the virt-install command parameters and their purpose.

```
virt-install -h
```

[-n*name* | --name=*name*]
Sets the name of the guest instance.

```
virt-install -nMyGuest
```

[-r*RAM* | --ram=*RAM*]
Sets the memory to allocate for a guest instance in Megabytes.

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

[-u*UUID* | --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
```

[-f*diskfile* | --file=*diskfile*]
Sets the file to use as the disk image.

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

[-s*filesize* | --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.

[-m*value* | --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
```

[-b*value* | --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
```

[-k*value* | --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
```

[-c*CD-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, `fedora6`, `rhel5`, `solaris10`, `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 os-type=windows	Possible values for os-type=unix	Possible values for os-type=other	Possible values for os-type=linux
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

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
```

[[-x*args* | --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 for examples on using the xm command-line tool.

Options

console {*domain-id*}
Attaches to a domain's console.

```
xm console mydomain
```

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 mydomain
```

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 mydomain
```

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
```

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 mydomain 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 mydomain
```

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 mydomain
```

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 mydomain /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 mydomain
```

top
Displays real time monitoring information of the host and domains.

```
xm top
```

unpause {*domain-id*}
Unpauses a paused domain.

```
xm unpause mydomain
```

B

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 server 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 | <u>no</u>}
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 | <u>no</u>}
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 | <u>no</u>}
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 {<u>yes</u> | 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 | <u>no</u>}
Sets the 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 {<u>vif-bridge</u> | 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 | <u>no</u>}

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')
```

C

Guest Configuration Files

This Appendix gives you 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 /etc/xen/*domain*.cfg and use the following command to create the guest:

```
xm create domain.cfg
```

This Appendix contains:

- Configuration File Example 1
- Configuration File Example 2

C.1 Configuration File Example 1

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.2 Configuration File Example 2

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
```

D

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
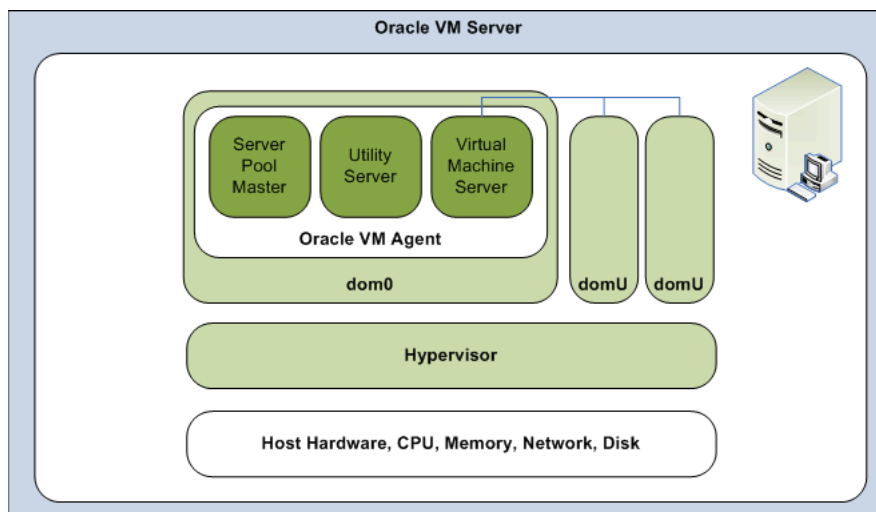
D.2 Oracle VM Agent Deployment

Oracle VM Manager communicates with Oracle VM Agent to perform management of virtual machines and 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 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
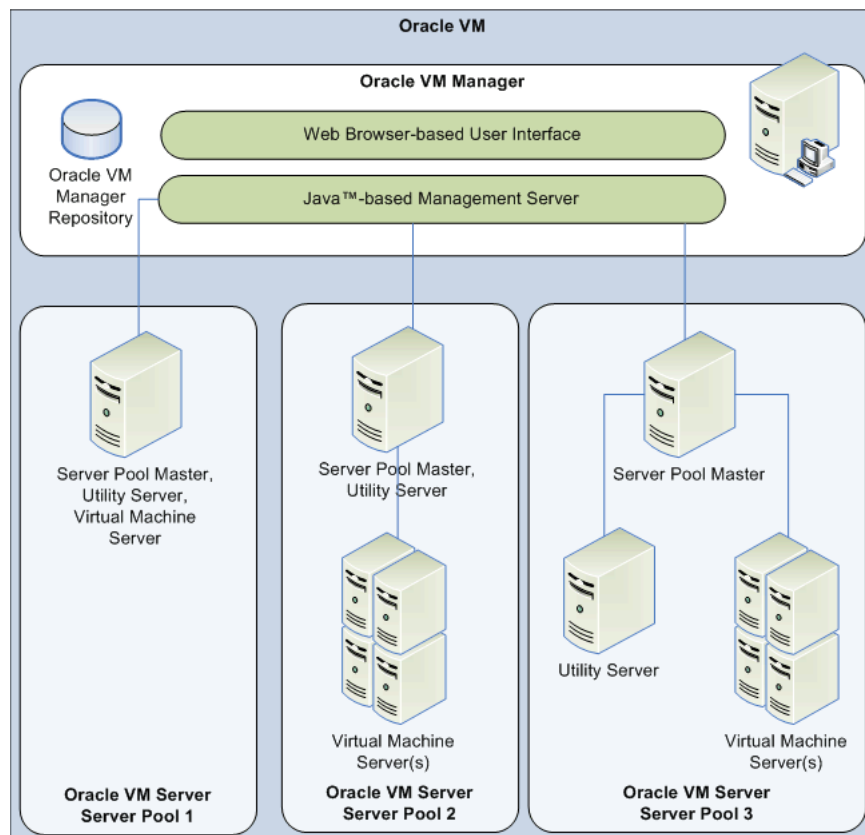
A server pool consists of one or more physical servers, and represents a logical view of the storage where the guests reside.

In Figure D–2, "Oracle VM deployment", Server Pool 3 is deployed on individual servers, while in Server Pool 1, all of the three agents are deployed on a single 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 physical 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 servers on the server pool.

The decision on how to deploy Oracle VM Agent components is made with Oracle VM Manager. You cannot configure this from 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.

E

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 Killed
- Hardware Virtualized Guest Devices Not Working as Expected
- Hardware Virtualized Guest Windows Installation
- TAP Disks with Paravirtualized Guests
- 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 xm log 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 quemu-dm process. Use the ps command to find the *pid* (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
xen top	Displays real-time information about Oracle VM Server and domains.
xm dmesg	Displays log information on the hypervisor.
xm log	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'
vncpassword '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 as a VNC configuration entry in a hardware virtualized guest configuration fie:

```
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 = 'mypassword'   # 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 Killed

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 be killed as it runs out of memory. If the guest is killed, 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 Hardware Virtualized Guest Windows Installation

During the installation of Microsoft Windows™ XP and Microsoft Windows™ 2003 as hardware virtualized guests, the domain requires a reboot and attempts to find the installation CDROM. Oracle VM Server does not know how to find the CDROM. To workaround this problem:

1. During the domain reboot, press **F6** or **F5** to select a new HAL (Hardware Abstraction Layer).

2. Select **Standard PC** and add the CDROM to your guest configuration file, for example, add the following the /etc/xen/myWindowsGuest file:

   ```
   disk = [ 'file:/ovm/guests/win2003sp1.dsk,hda,w','file:/ovm/trees/ISO/WIN/en_
   windows_server_2003_with_sp1_standard.iso,hdc:cdrom,r', ]
   cdrom="/ovm/trees/ISO/WIN/en_windows_server_2003_with_sp1_standard.iso"
   ```

3. Restart the hardware virtualized guest with the `xm create` command. The installation continues.

E.12 TAP Disks with Paravirtualized Guests

This release of Oracle VM Server does not support tap Disks with paravirtualized guests. If you use tap disks, you may encounter problems. For example, only one domain can be started and shut down before the blktapctrl process becomes unusable (using tap:sync or tap:aio works once only). If you want to use tap disks, to work around this problem, kill and restart the blktapctrl process. Do not pass the blktapctrl process any arguments.

E.13 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.14 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.15 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.16 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 workaround this problem, you can include a console in the 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 rebooted 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, as well as 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 eth*n* to vif*n.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