IBM

PowerVM Lx86 for x86 Linux Applications Administration Guide

SA38-0650-05

PowerVM Lx86 for x86 Linux Applications Administration Guide

SA38-0650-05

Note

Before using this information and the product it supports, read the information in "Notices" on page 83.

Sixth Edition (November 2010)

This edition applies to version 1.4.1.0 of IBM PowerVM Lx86 for x86 Linux Applications and to all subsequent releases and modifications until otherwise indicated in new editions.

Contents

About this publication v
How to send your comments v

Typographical conventions vii

Chapter 1. Introduction to PowerVM Lx86 1
New features in PowerVM Lx86 1.4 1
PowerVM Lx86 overview 2
PowerVM Lx86 concepts 3
  VxE for PowerVM Lx86 3
  x86 World for PowerVM Lx86 4
  Jailing and escapes for PowerVM Lx86 5

Chapter 2. Installing and uninstalling PowerVM Lx86 9
System and installation requirements for PowerVM Lx86 10
  Minimum hardware requirements for PowerVM Lx86 11
  Downloading ISOs for PowerVM Lx86 11
  Installing the Advance Toolchain 11
Installing PowerVM Lx86 and x86 World 12
PowerVM Lx86 installation reference 12
Upgrading PowerVM Lx86 14
  Upgrading previous versions of PowerVM Lx86 14

Chapter 3. Uninstalling PowerVM Lx86 and x86 World 15

Chapter 4. Creating and duplicating x86 Worlds with archives 17
Creating an archive of the x86 World 17
Installing an x86 World archive 17

Chapter 5. Running the automated installer 19

Chapter 6. Running and installing x86 applications 23
Running x86 applications 23
  Starting the PowerVM Lx86 daemon 24
  PowerVM Lx86 log files 25
  Viewing the installation log file - logviewer. . . 26
  Reporting a PowerVM Lx86 failure 26
  Monitoring x86 applications 27
  Maintaining x86 applications 27
Installing and configuring x86 applications on a PowerVM Lx86 system 28
  Setting up the x86 system configuration 28
  Installing x86 applications 29

Chapter 7. Configuring and maintaining the x86 World and PowerVM Lx86 . . . 33
Installing and updating packages in x86 World . . 33
  Installing and updating packages for Red Hat (RHEL 4) in x86 World 33
  Installing and updating packages for Novell SLES 10 in x86 World 34
PowerVM Lx86 configuration settings 36
Default PowerVM Lx86 escapes and virtual files in the x86 World 42
  Virtual files and directories 42
  Default escaped directories, files, and sockets . . 44
  Syslog support 45
Starting x86 daemons with PowerVM Lx86 46
  x86 /etc/init.d support scripts 46
SE Linux support with PowerVM Lx86 47
  Enabling SE Linux in PowerVM Lx86. 48
  Building custom policies for PowerVM Lx86 . . 49

Chapter 8. Managing remote and local users with PowerVM Lx86 53
Managing remote users with PowerVM Lx86 . . . 53
Managing local users, groups, and passwords with PowerVM Lx86 53
  The solution: A unified view. 54
  Periodic checks of the subject x86 World. . . . 54
  Known issues with the unified view approach. . 55
  WORLD_CHECK_OR_SYNC options 56
  Root user and root passwords 56

Chapter 9. PowerVM Lx86 error messages and resolutions 59
PowerVM Lx86 (powervm-lx86) errors 59
PowerVM Lx86 log file errors 62
PowerVM Lx86 daemon (powervm-lx86-daemon) errors 63
Errors and warnings from the runx86 script . . . 64
Errors from the linkx86 script 66
Errors from the x86 /etc/init.d support scripts . . 67
Errors from the execve router 71
Syslog messages 71
Alerts and errors when managing local users, groups, and passwords with PowerVM Lx86 . . . 71
E-mail messages for managing local users, groups, and passwords in PowerVM Lx86 73

Appendix. Known issues with PowerVM Lx86 77
PowerVM Lx86 translation process 78
Glossary 80
Accessibility features 81

Notices 83
Trademarks 85

Terms and conditions 85

About this publication

This guide explains how to install and configure the PowerVM™ Lx86 for x86 Linux® Applications (PowerVM Lx86) product on a Linux on POWER® system.

For information about the accessibility features of this product, see “Accessibility features” on page 81.

How to send your comments

Your feedback is important in helping to provide the most accurate and highest quality information. If you have any comments about this publication, send your comments using Resource Link™ at http://www.ibm.com/servers/resourcelink. Click **Feedback** on the navigation pane. Be sure to include the name of the book, the form number of the book, and the specific location of the text you are commenting on (for example, a page number or table number).

Typographical conventions

The following typographical highlighting conventions are used in this book:

Font	Usage
Bold	Identifies commands and graphical objects such as buttons, labels, and icons that you select.
Italics	Identifies parameters whose actual names or values you are to supply.
`Monospace`	Identifies examples of text similar to what you might see displayed, examples of portions of program code similar to what you might enter, messages from the system, or information you should literally type.
`%`	Identifies information you should type on the command line of a POWER shell.
`$`	Identifies information you should type on the command line of a translated x86 shell.

Chapter 1. Introduction to PowerVM Lx86

This guide explains how to install and configure the PowerVM Lx86 for x86 Linux Applications (PowerVM Lx86) product on a Linux on POWER system.

The guide has the following parts:

- **Introduction to PowerVM Lx86.** Gives an overview of the product, the new features in the product and includes explanations of some of the terms and concepts unique to PowerVM Lx86.
- **Installing and uninstalling PowerVM Lx86.** Explains how to install PowerVM Lx86 and the system requirements needed to run the product. A detailed installation reference is provided as well as details of how to uninstall the product. Also explains how to use the more advanced archiving and automated installation features of PowerVM Lx86.
- **Running and installing x86 applications.** Overviews and examples show you how to use PowerVM Lx86 to install and run your Linux on x86 applications on Linux on POWER.
- **Configuring x86 World and PowerVM Lx86.** Explains how to upgrade PowerVM Lx86, configure the product for your specific needs and how to install and update the packages in the x86 World. Details of how to start x86 daemons and how to use SE Linux are also provided.
- **Managing remote and local users with PowerVM Lx86.** Explains how the system deals with two definitions of local users, groups, and passwords on the system because the x86 World and the POWER system both have their own set of own set of local password files and also how to configure the x86 World to use remote users with NIS, LDAP and other remote user mechanisms.
- **PowerVM Lx86 error messages and resolutions.** Shows errors that can be reported on the terminal by the components of PowerVM Lx86 and details of how to resolve each problem.

At the end of the guide there are also appendices detailing known issues with PowerVM Lx86, a description of the translation process used by PowerVM Lx86 and a glossary of terms. This document should be used in conjunction with the PowerVM Lx86 Release Notes, which provide additional information specific to a particular release. This includes known limitations and any additional installation instructions.

New features in PowerVM Lx86 1.4

The 1.4 release of PowerVM Lx86 adds major new features and a range of improvements to the existing product.

The 1.4 release adds the following major new features to PowerVM Lx86:

- Support for RHEL5.5.
- Support for POWER 7 processors. Note that RHEL5 only supports POWER7™ in POWER6® Compatibility Mode. Customers will need to use SLES11 to take full advantage of POWER7. PowerVM Lx86 supports POWER6 Compatability mode on POWER7.
- Visibility of POWER processes in the VxE /proc filesystem. This feature is off by default and enabled via a switch in the config file.
- General performance enhancements.
- General bug fixes.

The 1.4 release removes support for the following in PowerVM Lx86:

- POWER 5 processors.
- Support for RHEL 4.4 and RHEL 4.5

Note: Support for SLES 9 SP3 and SP4 was removed in the 1.3.1 update release. Customers wanting to use SLES 9 will need to use PowerVM Lx86 version 1.3 or earlier.

PowerVM Lx86 overview

PowerVM Lx86 enables POWER machines to run Linux x86 applications alongside native Linux on POWER applications.

The Linux x86 applications run on Novell or Red Hat Linux x86 distributions within a Virtual x86 Environment (VxE). No modification or recompilation of the x86 Linux applications are needed. The x86 operating system and applications only need to be installed on or copied to a Linux on POWER machine that has PowerVM Lx86 installed on it.

Installing PowerVM Lx86 on a POWER system makes the system compatible with x86 applications. This extends the application support for Linux on POWER, allowing applications that are available on x86 but not on POWER to be run on the system.

How it works

PowerVM Lx86 creates a VxE within which x86 applications can run. The VxE is created solely within user-space; no modifications to the POWER kernel are required. PowerVM Lx86 does not run the x86 kernel on the POWER machine. Instead, it dynamically translates and maps all requests made from within the VxE to the underlying Linux operating system and POWER processor. The VxE is not a virtual machine; rather, x86 applications are encapsulated so the operating environment appears to be Linux on x86, even though the underlying system is Linux on POWER.

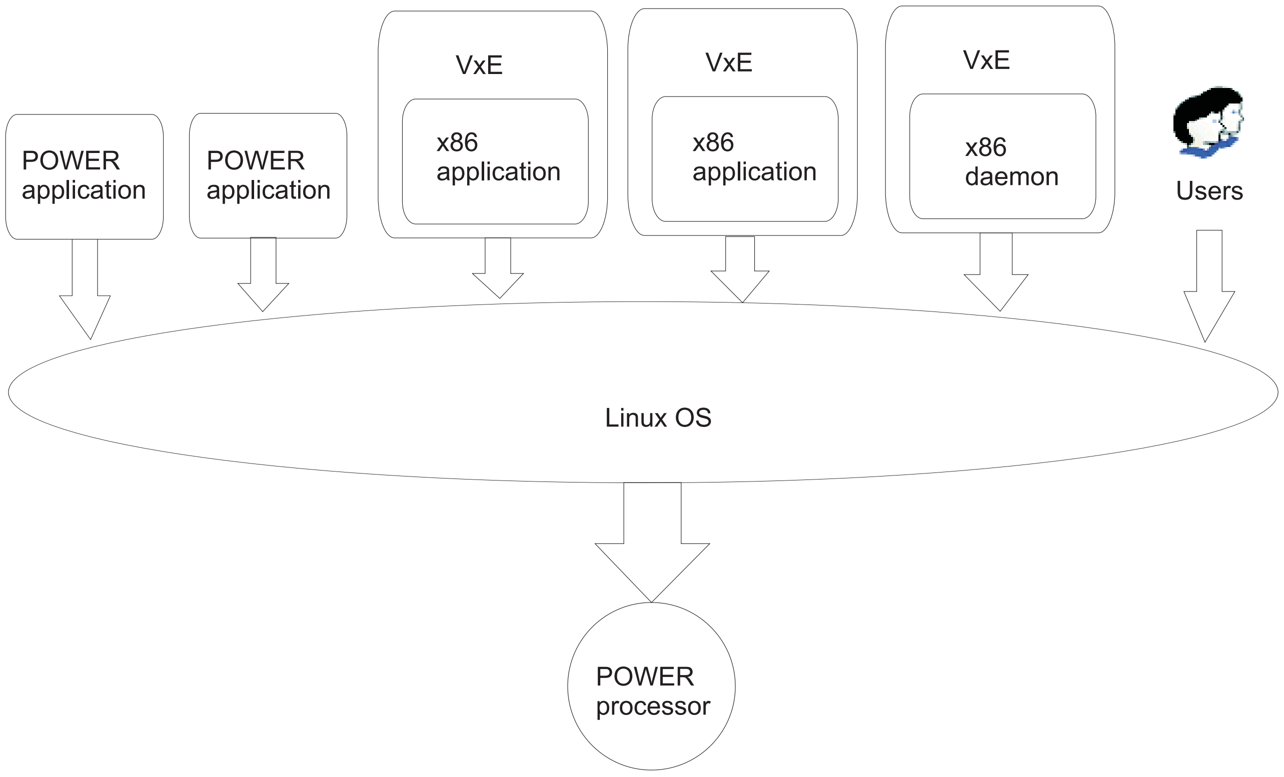


Figure 1. How it works

See “PowerVM Lx86 translation process” on page 78 in this guide for a detailed description about how requests made from within a VxE are dynamically translated and mapped to the underlying Linux operating system and POWER processor.

PowerVM Lx86 does not affect Linux on POWER applications

An x86 application running within a VxE appears to be just another user-space POWER process. It has no direct effect on any POWER applications that run natively on the host system.

As an end user

If you want to run x86 applications, you can log on to the POWER system and run the x86 applications within a VxE using PowerVM Lx86. See "Running x86 applications" on page 23 section in this guide.

PowerVM Lx86 installation

IBM® supplies a semi-automated installation script and RPM packages.

See Chapter 2, "Installing and uninstalling PowerVM Lx86," on page 9 in this guide for full installation instructions.

Installation of x86 applications

You can install x86 applications using their original installation scripts from within a VxE. Alternatively, you can copy x86 applications onto the POWER machine. See "Installing x86 applications" on page 29 for details.

PowerVM Lx86 concepts

PowerVM Lx86 concepts include the VxE, the x86 World, jailing, and escapes.

VxE for PowerVM Lx86

PowerVM Lx86 creates a Virtual x86 Environment (VxE) within which x86 applications can run.

The VxE consists of:

- A set of x86 Linux libraries, commands, applications, and other system files that are installed in a directory, known as x86 World, on the POWER system.
- A program, known as the translator, that handles the mapping of instructions and requests from the VxE onto the underlying POWER system.
- Selective integration between the VxE and the POWER environment. For example, a subset of the Linux on POWER file system can be made visible to applications from within the VxE.

Use the **runx86** command to run x86 binaries. For example, to run the x86 binary myx86Binary within a VxE, you enter the following command from a native POWER shell on the host system:

```
% runx86 ./myx86Binary
```

In this example, when myx86Binary finishes executing, the **runx86** command also exits, and the VxE no longer exists.

If you enter the **runx86** command without providing an argument, then **runx86** starts an x86 shell within a new VxE. You then interact with the x86 shell as though you are on an x86 machine. Any commands entered from the x86 shell are then automatically run within a VxE.

Following is an example using the **runx86** command:

```
% uname –srmpi
Linux 2.6.9 ppc64 ppc64 ppc64
% runx86
$ uname –srmpi
Linux 2.6.9 i686 i686 i386
$ exit
%
```

After you finish working with the x86 shell, use the **exit** command as if you were using a native x86 shell on an x86 machine. After you exit, the VxE that ran the x86 shell no longer exists, and you return to the POWER shell.

Notes:

- It is not possible to run an x86 binary directly from a native POWER shell. To ensure that the x86 binary is run within a VxE, the **runx86** command must always be used, either with the x86 binary as an argument, or without any arguments to start an x86 shell. From the shell you can then directly run the x86 binary.
- Not all commands can be run in a VxE. For example, you should do system administration work from a native POWER shell. However, software management of the x86 applications and environment in the x86 World should be done in the VxE.

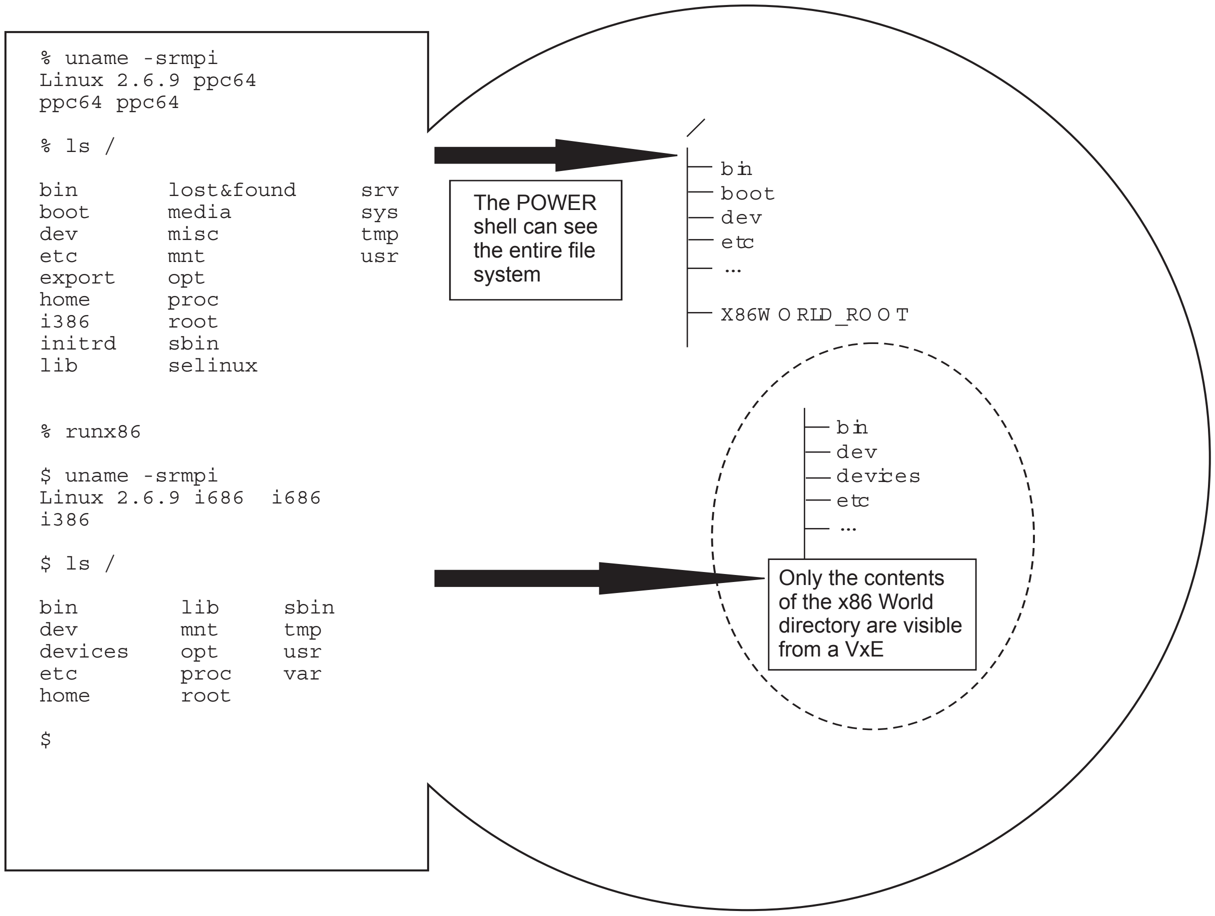
For more information about using **runx86**, see "Running x86 applications" on page 23.

x86 World for PowerVM Lx86

The x86 World directory contains the x86 binaries, common x86 libraries, and infrastructure files. x86 applications should also be installed in this directory. This directory is configurable and is specified during the installation process. The default location for x86 World is the `/i386` directory. The convention *X86WORLD_ROOT* is used in this document to represent the directory where the x86 files are installed.

An x86 World can be created using x86 libraries and binaries obtained from an existing x86 Linux operating system distribution.

As described previously, you run x86 binaries within a VxE with the **runx86** command. For an x86 application or shell running within a VxE, the accessible part of the file system is restricted to x86 World. This is shown in the following example. The area ringed with a dashed line is x86 World.



```
% uname -srmpi
Linux 2.6.9 ppc64
ppc64 ppc64

% ls /

bin       lost&found     srv
boot      media          sys
dev       misc           tmp
etc       mnt            usr
export    opt
home      proc
i386      root
initrd    sbin
lib       selinux


% runx86

$ uname -srmpi
Linux 2.6.9 i686  i686
i386

$ ls /

bin        lib     sbin
dev        mnt     tmp
devices    opt     usr
etc        proc    var
home       root

$
```

The POWER shell can see the entire file system
/
bin
boot
dev
etc
...
X86WORLD_ROOT
bin
dev
devices
etc
...
Only the contents of the x86 World directory are visible from a VxE

Figure 2. x86 World

In this example, when the **ls** command is issued from within the x86 shell, the x86 `/bin/ls` binary is run within a VxE, and it displays the contents of the root of the x86 World filesystem. That is `/i386` by default.

This restriction of the view of the Linux file system is called *jailing*, and is similar in concept to UNIX® **chroot**.

Jailing and escapes for PowerVM Lx86

Jailing an application ensures that it accesses x86 libraries and infrastructure files, just as if the application is running natively on an x86 machine.

Any application running within a VxE sees the directory structure under *X86WORLD_ROOT* as its root file system. The VxE replaces references to / at the start of a file path with *X86WORLD_ROOT*.

Following is an example of a jail:

Assume the x86 application tries to open the `/lib/libc.so.6` library.

If this path is left unmodified and passed through to the POWER kernel, then the application incorrectly opens the POWER version of the C runtime library, which is incompatible with x86.

So, the path is jailed, and the jailed path is passed through to the POWER kernel as *`X86WORLD_ROOT`*`/lib/libc.so.6`.

This causes the x86 C runtime library to be opened, and subsequently loaded, allowing the x86 application to run.

Escapes (access to files outside of x86 World)

Sometimes it is necessary for an x86 application to access files that are located outside of x86 World. For example, the application might need to access data stored on a remote shared filesystem, or it might need to access data that is stored locally but not within x86 World. To allow access to files stored locally that are external to x86 World, a mechanism known as an *escape* is used.

An escape links a path inside x86 World with a path outside x86 World. You create escapes with the **linkx86** command supplied with PowerVM Lx86. The **linkx86** command enables PowerVM Lx86 to recognise escapes. The escape is then transparent to an x86 application running within a VxE, similarly to the way a mounted file or directory in Linux is transparent to a Linux application. Escapes persist across reboots of the host POWER system, and they are visible to all x86 applications.

Note: An escape is implemented as a symbolic link that follows a specific naming convention to ensure that access from the VxE functions correctly. It is not recommended to use the **ln** command directly to create escapes. This can cause x86 applications running within a VxE to fail. Due to the nature of its implementation, an escape can be safely removed from the system by deleting the link using a POWER shell.

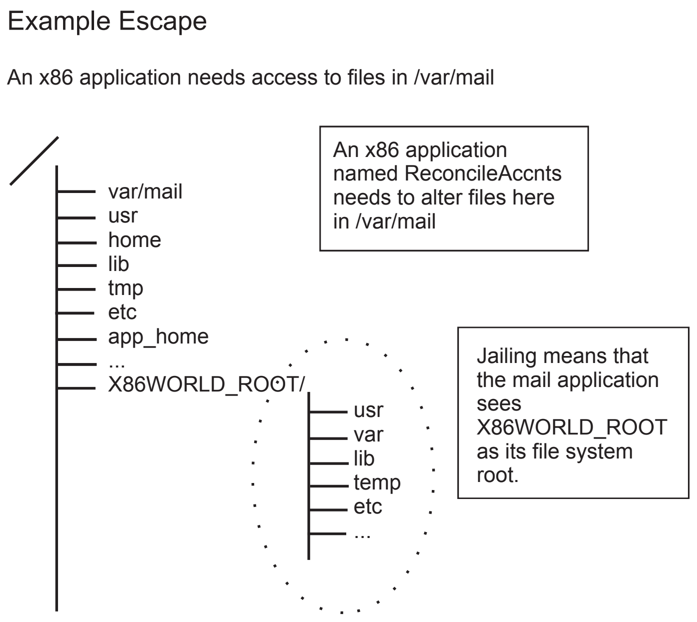

Example Escape
An x86 application needs access to files in /var/mail
var/mail
usr
home
lib
tmp
etc
app_home
...
X86WORLD_ROOT/
usr
var
lib
temp
etc
...
An x86 application named ReconcileAccnts needs to alter files here in /var/mail
Jailing means that the mail application sees X86WORLD_ROOT as its file system root.

Figure 3. Example escape

You can use the **linkx86** command to create an escape to the `/var/mail` directory, as the following example demonstrates, starting in a POWER shell:

```
% runx86

$ ls /var
cache lib log
$ exit

% linkx86 /var/mail
% runx86

$ ls /var
cache lib log mail
$
```

The x86 application now sees the `/var/mail` directory.

If access is needed to a remote filesystem from within a VxE, then the remote filesystem can be mounted to a mount point under the x86 World root or mounted in a native POWER shell with an escape created from the x86 World. The **mount** command and the **linkx86** command should both be issued from a native POWER shell, and not from an x86 shell.

See section "Creating mountpoints or access to devices" on page 30 for details of how to create a mount point accessible to x86 World.

Chapter 2. Installing and uninstalling PowerVM Lx86

This section describes how to install and uninstall PowerVM Lx86. This reference includes how to use the automated installation features of the PowerVM Lx86 installer and how to archive an installed x86 World for backup, migration to another system, or distribution to a set of systems. The PowerVM Lx86 installation script installs the software. It should always be used when installing, upgrading or uninstalling PowerVM Lx86 and x86 Worlds. The installer.pl script installs the required PowerVM Lx86 RPM package. With the installer.pl script you can install an x86 World using an x86 distribution from CD, DVD or ISO images.

This reference includes:

- System and installation requirements.
- Downloading Linux x86 distribution media.
- Running the PowerVM Lx86 installation script.
- Information about uninstalling PowerVM Lx86.
- Details of where the PowerVM Lx86 and x86 World files are installed on the host system.
- Archiving previously installed x86 Worlds for backup or migration to other systems.
- Automated installation for non-interactive installation of both PowerVM Lx86 and the x86 World. The automated installation facility also supports installing the x86 World from an archive.
- IBM Installation Toolkit for Linux on POWER.

Release contents

A PowerVM Lx86 release can be obtained on a CD or downloaded from IBM's Web site.

A typical release contains the following files:

- powervm-lx86-1.4.0.0-1.tgz: the product package containing the PowerVM Lx86 binaries, documentation and installer script.
- powervm-lx86-release-notes-1.4.0.0.txt: the latest Release Notes for the product containing amendments to the Administration Guide in text format.
- powervm-lx86-release-notes-1.4.0.0.pdf: the latest Release Notes for the product containing amendments to the Administration Guide in PDF format.

The PowerVM Lx86 product package has the following directory structure within the tgz file:

- `powervm-lx86-installer-1.4.0.0-1/installer.pl`
- `powervm-lx86-installer-1.4.0.0-1/lib/`
- `powervm-lx86-installer-1.4.0.0-1/resources/`
- `powervm-lx86-installer-1.4.0.0-1/doc/`

To install PowerVM Lx86, copy the product package, powervm-lx86-1.4.0.0-1.tgz, to the local machine. Extract the file and then, as root, run the powervm-lx86-installer-1.4.0.0-1/installer.pl script.

IBM Installation Toolkit for Linux on POWER support

The IBM Installation Toolkit for Linux on POWER is supported by PowerVM Lx86 and will make use of the advanced PowerVM Lx86 installation features.

See the IBM Installation Toolkit for Linux on POWER website for further details, http://www.ibm.com/developerworks/power/library/l-power-installation-toolkit/index.html.

System and installation requirements for PowerVM Lx86

Requirements include the necessary access levels, hardware, and software requirements for the installation.

Prerequisites

Root access when installing the PowerVM Lx86 RPM package and x86 World is necessary.

On all operating systems except RHEL4 you must install the Advanced Toolchain. See "Installing the Advance Toolchain" on page 11

Supported POWER operating systems summary

- Red Hat 4 AS (RHEL 4 AS) Linux Update 6, Update 7 and Update 8
- Red Hat 5 AS (RHEL 5 AS) Linux Update 3, Update 4 and Update 5
- Novell SUSE Linux Enterprise Server 10 (SLES 10)
- Novell SUSE Linux Enterprise Server 10 (SLES 10) Service Pack 1, service Pack 2 and Service Pack 3
- Novell SUSE Linux Enterprise Server 11 (SLES 11)

Supported x86 and POWER operating system combinations

The officially supported OS combinations for the 1.4.0 release are:

Table 2. Officially supported OS combinations

POWER OS distribution	Supported x86 World OS distribution
RHEL 4.6	RHEL 4.6
RHEL 4.7	RHEL 4.6 RHEL 4.7
RHEL 4.8	RHEL 4.6 RHEL 4.7 RHEL 4.8
RHEL 5.3	RHEL 5.3
RHEL 5.4	RHEL 5.3 RHEL 5.4
RHEL 5.5	RHEL 5.3 RHEL 5.4 RHEL 5.5
SLES 10	SLES 10
SLES 10 SP1	SLES 10 SLES 10 SP1
SLES 10 SP2	SLES 10 SLES 10 SP1 SLES 10 SP2

SLES 10 SP3	SLES 10 SLES 10 SP1 SLES 10 SP2 SLES 10 SP3
SLES 11	SLES 11

Note:
- Only x86 RHEL Application Server (AS), not ES or WS is supported by the installer.

Minimum hardware requirements for PowerVM Lx86

PowerVM Lx86 runs on the following hardware platforms:
- System p® server with POWER6 or POWER7 processors.

Downloading ISOs for PowerVM Lx86

Before you proceed with the installation, you must make sure that the x86 CDs, DVDs, or ISO images required by PowerVM Lx86 are available to the installer script.

The Virtual x86 Environment uses a set of x86 libraries, commands, applications and other system files. These are contained in the X86 Linux Distribution closely corresponding to the POWER Linux installed on the host machine. See “System and installation requirements for PowerVM Lx86” on page 10 for the Linux versions that can be installed on each given host operating system. If you do not already have the CDs, DVDs, or ISOs, you can download the ISO images from the Red Hat or Novell Web sites.

The best method for installation is to download all ISOs to one directory and provide the name of that directory to the installer.

Note For RedHat Customers

The software subscription for RedHat is system and architecture specific. Regular customers need to purchase one subscription to cover ISO images for the x86 architecture and a second subscription to cover the ISO images for the POWER architecture. However, you are not asked to pay for two subscriptions with PowerVM Lx86. The PowerVM Lx86 installation collects the data needed for Red Hat to upgrade the Linux POWER entitlement to include an entitlement for Linux on x86 as well. This additional x86 entitlement is only to be used with PowerVM Lx86 on a POWER system. The Lx86 installer will provide prompts to enable this additional activation when run on RedHat systems.

Installing the Advance Toolchain

Before you proceed with the installation you must download the RPMs for the Advance Toolchain and install them on to your POWER system. This does not apply on RHEL4 systems.

The Advance Toolchain provides a self contained set of system libraries in a non-default location (`/opt`.) It provides optimized system libraries for all POWER architectures.

Information on the Advance Toolchain including download locations can be found on IBM DeveloperWorks at http://www-128.ibm.com/developerworks/forums/forum.jspa?forumID=1518. Several RPMs are provided for each operating system, only the following are required by PowerVM Lx86:
- advance-toolchain-devel
- advance-toolchain-runtime

Installing PowerVM Lx86 and x86 World

Always use the supplied installer to install PowerVM Lx86 and x86 World.

The following procedure assumes that you are installing onto a system for the first time or that any previous installations of PowerVM Lx86 have been uninstalled. If the default locations are accepted PowerVM Lx86 is installed in the `/opt/powervm-lx86` directory. You can install it in an alternative location if necessary.

The default location for the x86 World libraries and binaries, referred to as *X86WORLD_ROOT* in this document, is the `/i386` directory.

1. Download the PowerVM Lx86 package on to your system from the IBM website or copy the PowerVM Lx86 files from the CD.
2. Become root on the POWER system.
3. Extract the PowerVM Lx86 installer. For example:

   ```
   % tar -xvzf powervm-lx86-1.4.0.0-1.tgz
   ```

4. Run the PowerVM Lx86 installation script:

   ```
   % powervm-lx86-installer-1.4.0.0-1/installer.pl
   ```

   You are asked to read through and agree to the license agreement before continuing with the installation. The installer provides a set of menus with options to install, upgrade or remove PowerVM Lx86.

For further details about how to configure PowerVM Lx86 and x86 World, see Chapter 7, "Configuring and maintaining the x86 World and PowerVM Lx86," on page 33.

To start running x86 applications, see "Running x86 applications" on page 23.

PowerVM Lx86 installation reference

This section includes the details of a PowerVM Lx86 installation including PowerVM Lx86 directories and files.

PowerVM Lx86 directory structure

The PowerVM Lx86 installer will install the software, by default, into the directory structure as shown in this chart:

Directory	File or sub-directory	Description
/opt/powervm-lx86/bin/	`powervm-lx86`	The PowerVM Lx86 translator application.
	`powervm-lx86-daemon`	PowerVM Lx86 translator daemon.
	`i386-router`	PowerVM Lx86 router.
	`dependency_checker.pl`	/etc/init.d dependency checker (SLES systems only).
	`powervm-lx86-rcmonitor`	/etc/init.d directory monitor application (SLES systems only).
/opt/powervm-lx86/lib/perl5	powervm_lx86_scripts/	PowerVM Lx86 globalization support files.
/opt/powervm-lx86/locale	*locale directories*	PowerVM Lx86 globalization files.
/opt/powervm-lx86/selinux	*selinux files*	SE Linux support files (RHEL systems only).

Directory	File or sub-directory	Description
/opt/powervm-lx86/extras	powervm-lx86-tools-1.0-1.i386.rpm	An RPM installed in the VxE containing utilities useful from within the VxE.
/etc/init.d	`powervm-lx86`	Start-up script to start PowerVM Lx86 daemon.
	`powervm-lx86-rc1`	x86 init.d runlevel script.
	`powervm-lx86-rc2`	x86 init.d runlevel script.
	`powervm-lx86-rc3`	x86 init.d runlevel script.
	`powervm-lx86-rc4`	x86 init.d runlevel script.
	`powervm-lx86-rc5`	x86 init.d runlevel script.
	`powervm-lx86-rc6`	x86 init.d runlevel script.
	`powervm-lx86-rccommon`	x86 init.d runlevel script.
	`powervm-lx86-rcmonitor`	/etc/init.d directory monitor script (SLES systems only).
/etc/opt/powervm-lx86	config	Configuration file, created if PowerVM Lx86 or x86 World are installed into non-default locations.
/etc/opt/powervm-lx86	activation-detail	Contents of IBM activation e-mail created during installation.
/etc/opt/powervm-lx86	exec_types	SE Linux support file.
/etc/opt/powervm-lx86/license	license	Copy of license agreed to during installation.
/usr/sbin/	powervm-lx86-world-sync	Script to manage local user files.
/var/opt/powervm-lx86/log	*powervm-lx86.log.xx.yyyy.zzzzzz*	PowerVM Lx86 debug log files.
/var/opt/powervm-lx86/daemon	`powervm-lx86-daemon.log`	Contains event logs for the PowerVM Lx86 daemon.
	`powervm-lx86-daemon.lock`	The PowerVM Lx86 daemon execution lock.
	`cache`	The PowerVM Lx86 daemon cache.
/var/opt/powervm-lx86/selinux	*selinux files*	SE Linux support scripts, created when SE Linux enabled for PowerVM Lx86 (RHEL systems only).
/i386	*x86 World files*	The default directory for the x86 World files. When x86 applications are run, they see this directory as root (/).
/usr/local/bin	`linkx86`	Script to creates x86 World escapes.
	`runx86`	Script to invoke PowerVM Lx86.

Directory	File or sub-directory	Description

/opt/powervm-lx86/installer	`installer.pl`
	
	`lib/`
	
	`resources/`
	
	`doc/`
		Local copy of PowerVM Lx86 installer.
		
		Auxiliary installer files and installer globalization text.
		
		Installer RPMs and license files.
		
		Release Notes and PowerVM Lx86 Administration Guide.

/opt/powervm-lx86/doc	*Release notes* *Administration Guide*	Symlink to the Release Notes and the PowerVM Lx86 Administration Guide (all locales)

Upgrading PowerVM Lx86

You can upgrade earlier versions of PowerVM Lx86.

For systems with a previous installation of PowerVM Lx86 either:
- Upgrade to the new version of PowerVM Lx86, or
- Uninstall PowerVM Lx86 and the x86 World and install the new version of the software and a new x86 World.

Check the PowerVM Lx86 release notes to see which method is recommended for your version of PowerVM Lx86 and x86 World.

Upgrading previous versions of PowerVM Lx86

To upgrade PowerVM Lx86 to the new version, use the following instructions:
1. Check that all running x86 applications and processes have been closed or shut down.
2. Run the installer script: `% ./installer.pl`
3. Select option **2. Upgrade Software.** from the menu.
4. Select option **1. Upgrade a powervm-lx86 installation.**.
5. Select option **1. powervm-lx86-<*version name*>.** to upgrade from the current version of the software.
6. If you are upgrading from PowerVM Lx86 version 1.1.x or 1.2.x , the installer will prompt you to update the install path to use the new product directory name. `Would you like to update the install path from /opt/p-ave to /opt/powervm-lx86? [Y/n]` Press `Y` and then Enter or just press Enter to update the path or `N` and Enter to use the original directory name.
7. If you are upgrading from PowerVM Lx86 version 1.1.x or 1.2.x , the installer will prompt you to update the log file path. `Would you like to update the log file path from /var/opt/p-ave/log to /var/opt/powervm-lx86/log? [Y/n]` Press `Y` and then Enter or just press Enter to update the log file path or `N` and Enter to use the log file path.
8. When the upgrade is complete, select option **6. Quit.** from the menu.

Chapter 3. Uninstalling PowerVM Lx86 and x86 World

The installer.pl script is used to uninstall PowerVM Lx86 and x86 World from your system.

You can use the script to remove the following:

- PowerVM Lx86 software and configuration (optional)
- x86 World and any applications or user files installed or modified since the original installation (optional)

Notes:

- The **rpm -e** command cannot remove PowerVM Lx86. **rpm** cannot remove any files added to the installation directories, including any applications that have been installed within x86 World. The installer.pl script can, if you so choose, completely remove the installation directories and contents.
- You must run the installer.pl script from a POWER shell and not an x86 shell.
- Before uninstalling PowerVM Lx86 and the x86 World, please quit all running x86 applications. If any x86 applications are still running during the uninstall, you will be prompted to let the uninstall script kill these processes. You must kill the running processes for the uninstall to proceed.
- Back up any critical data (either applications or user files) from the x86 World before removing the x86 World.
- The installer.pl creates a full copy of itself in `/opt/powervm-lx86/installer` during the installation of PowerVM Lx86.

Chapter 4. Creating and duplicating x86 Worlds with archives

For faster deployment, duplication and storage purposes the PowerVM Lx86 installer can create and install images of the x86 World.

Creating an archive of the x86 World

For an existing PowerVM Lx86 and x86 World installation, the x86 World can be archived. The archive in an image file and can be copied and stored like any other file.

Once you have installed PowerVM Lx86 and the x86 World onto a system, you can create an archive of the x86 World at any point. The PowerVM Lx86 installer is used to create the archive and also used to unpack the archive onto the same or another system. A common deployment option is to create an x86 World installation on one system. You can install any additional custom packages, x86 applications and data files within that x86 World. Once the x86 World meets all your requirements, an archive can be made which can then be easily installed on multiple systems.

Note:

- The **star** package must be installed on the system before an archive can be created or installed. The PowerVM Lx86 installer script will inform you if the **star** package is not installed.
- The installer will suggest a default name for the archive file. Any name and extension is allowed.
- Any escapes in the x86 World will be preserved in the archive and will be recreated by the installer when the archive is installed on another machine, even if the location of the x86 World is different from the original x86 World.
- An x86 World archive can only be created in the interactive mode. Although you can install an archive onto a system using the non-interactive mode, you cannot create an archive using non-interactive mode. See Chapter 5, “Running the automated installer,” on page 19 for me details about the installer non-interactive mode.

1. Become root on the POWER system.
2. Run the PowerVM Lx86 installation script:
   ```
   % ./installer.pl
   ```
3. The option to create an archive of the x86 World is in the Configure Software section of the installer menu.

Installing an x86 World archive

This section describes how to install a previously created x86 World archive onto a system.

Chapter 5, “Running the automated installer,” on page 19

The PowerVM Lx86 installer is used to install the archive onto a system. You must first install PowerVM Lx86. The automated installer can also be used to install an x86 World archive, see Chapter 5, “Running the automated installer,” on page 19

Note:

- The **star** package must be installed on the system before an archive can installed. The PowerVM Lx86 installer script will inform you if the **star** package is not installed.
- The archive can only be installed onto a POWER system with a compatible Linux operating system. See section “System and installation requirements for PowerVM Lx86” on page 10 for the list of officially supported OS combinations.

- Any escapes in the original x86 World will be preserved in the archive and will be recreated by the installer when the archive is installed, even if the location of the x86 World is different from the original x86 World.

1. Become root on the POWER system.
2. Run the PowerVM Lx86 installation script:
   ```
   % ./installer.pl
   ```
3. The option to install an x86 World archive is in the Install Software section of the installer menu.

Chapter 5. Running the automated installer

The PowerVM Lx86 installer provides a non-interactive installation option, allowing PowerVM Lx86, the x86 World or an x86 World archive, to be installed automatically on a system using the command line.

Non-interactive mode supports installation and uninstallation of PowerVM Lx86, the x86 World or an x86 World archive, using the default locations (as per the interactive install), or using a predefined set of locations in a configuration file that uses the same format as the PowerVM Lx86 configuration file. For details, see "PowerVM Lx86 configuration settings" on page 36. The installation of the x86 World from media (CD and DVD ISO images, a directory of RPMs, or a mounted DVD), and from an archive are both supported. The installer's upgrade and configuration options are not supported in non-interactive mode.

Note:
- The **star** package must be installed on the system before an archive can unpacked and installed by the non-interactive mode. The PowerVM Lx86 installer script will inform you if the **star** package is not installed.
- An x86 World archive can only be created in the interactive mode. Although you can install an archive onto a system using the non-interactive mode, you cannot create an archive using non-interactive mode.
- Any escapes in the x86 World will be preserved in the archive and will be recreated by the installer when the archive is installed on another machine, even if the location of the x86 World is different from the original x86 World.

The default installation locations are:
- PowerVM Lx86 directory: `/opt/powervm-lx86`
- PowerVM Lx86 log file directory: `/var/opt/powervm-lx86/log`
- PowerVM Lx86 configuration file: `/etc/opt/powervm-lx86/config`
- x86 World directory: `/i386`

License agreement

You must accept the terms of the PowerVM Lx86 license agreement to install the product. If you have not read the license agreement, run the installer script in interactive mode first, by typing `% ./installer.pl` and read the license agreement and then quit the installer. In non-interactive mode, you accept the license agreement by passing `--license-accepted` to the installer as a command-line argument.

Examples of usage

Install PowerVM Lx86 to the default location (`/opt/powervm-lx86`):

```
% ./installer.pl --install Lx86 --license-accepted
```

Install PowerVM Lx86 and the x86 World of type `SLES10_MIN` (the SLES 10 minimal OS install) to the default locations (`/opt/powervm-lx86` and `/i386` respectively):

```
% ./installer.pl --install Lx86 --license-accepted --install x86world --distro SLES10_MIN
--media /path/to/sles10.cd1.iso --media /path/to/sles10.cd2.iso
```

Install PowerVM Lx86 and an x86 World archive:

```
% ./installer.pl --install Lx86 --license-accepted --install x86world --archive
/powervm-lx86-archive-SLES10_MIN.bin
```

Use a configuration file to specify a non-default location for PowerVM Lx86:

```
% ./installer.pl --install Lx86 --license-accepted --config my-config.conf
```

Non-interactive configuration file options

The following options are supported in the installation configuration file to change the behaviour of the installer in non-interactive mode.

The configuration file may also be used as the PowerVM Lx86 configuration if it is stored in the standard file location: `/etc/opt/powervm-lx86/config`. If the file is placed in this location, the `--config` option is unnecessary as the configuration file will be read automatically by the non-interactive installer and any installer settings in the file will override the default settings. To further override any of the settings in the configuration file, specify the values on the command line.

Non-interactive mode configuration switch name	Parameters and use
INSTALLER_SW_TARBALL_LOCATION	**Type** String **Parameters** An absolute path **Use** Optional. Same as `--archive` on the command line. This switch specifies where the x86 World archive is located.
INSTALLER_SW_DISTRO	**Type** String **Parameters** An absolute path **Use** Optional. Same as `--distro` on the command line. This switch specifies which Linux distribution will be installed for the x86 World, also designating a minimal or full installation.
INSTALLER_SW_INSTALL_MEDIA	**Type** String **Parameters** An absolute path **Use** Optional. Same as `--media` on the command line. This switch specifies the location of the Linux distribution media. May be specified multiple times in the configuration file.
POWERVM_LX86_LOCATION	This switch specifies the PowerVM Lx86 installation directory, see "PowerVM Lx86 configuration settings" on page 36 for further details.
LOGFILE_PATH	This switch specifies the PowerVM Lx86 log file directory, see "PowerVM Lx86 configuration settings" on page 36 for further details.
SUBJECT_WORLD_ROOT	This switch specifies the x86 World directory, see "PowerVM Lx86 configuration settings" on page 36 for further details.

Non-interactive mode configuration switch name	Parameters and use
INSTALLER_SW_HOMEDIR	This switch specifies the x86 World /home directory. You should set this to the directory containing user home directories on your system if they are in a different location to the default of /home. The home directories are automatically escaped as part of the install process. See "Default PowerVM Lx86 escapes and virtual files in the x86 World" on page 42 for more information on escapes in the x86 World.

Chapter 6. Running and installing x86 applications

This section describes how to run x86 applications using PowerVM Lx86 and also how to install additional x86 applications into the x86 World.

Running x86 applications

You must use the **runx86** command from a native POWER shell.

All x86 applications must be run within a VxE. The **runx86** command must always be used to ensure that an application, command, or utility is run within a VxE. Applications can be run in a VxE in either of the following ways:

- Use the **runx86** command to start an x86 shell. From within the x86 shell, run the x86 application as you normally would on an x86 system.
- Run x86 applications from a native POWER shell by using the **runx86** command with the x86 application as an argument to the command.

Notes:

- When invoking an x86 application, the current working directory must be accessible from the x86 World (for example, the *X86WORLD_ROOT* itself, `/i386`). Either the application must be installed within the x86 World or be visible from the x86 World. You can make the application visible from the x86 World by either mounting the application directly to a mountpoint within x86 World, or to a mountpoint external to x86 World and then creating an escape to the mountpoint. To create the escape, use the **linkx86** command. If the current working directory is not visible from the x86 World, the current working directory will be changed in the x86 shell to the *X86WORLD_ROOT* and a warning message will alert you to this change of directory.
- The PowerVM Lx86 translation daemon must be running in order to run x86 applications on the POWER machine. The PowerVM Lx86 installer will start the PowerVM Lx86 daemon on the system after the installation is completed. For further details about the PowerVM Lx86 translation daemon, see “Starting the PowerVM Lx86 daemon” on page 24

Running x86 applications from an x86 shell

Using an x86 shell is the most flexible way of running x86 applications, but it has disadvantages in that the starting process is manual. For these reasons, running applications from an x86 shell is generally only suited to applications that are run by experienced users or system administrators. An example of this is a middleware application that is run on an application server.

Example: Running applications from an x86 shell

This example shows how to run an application named TradeOffice. It normally runs on a Linux on x86 machine on a network. TradeOffice monitors a designated remote file system, processes files from the file system, and sends files to another remote file system.

In a POWER shell, enter the following command:

```
% runx86
```

The **runx86** command creates a VxE and starts an x86 shell from the native shell.

To confirm you are using a translated x86 shell, you can check the current architecture of the shell by entering the following command:

```
$ arch
```

A translated x86 shell will output `i686` rather than ppc64.

In the translated x86 shell, start the application with the following command:

```
$ TradeOffice
```

This starts the TradeOffice application from the x86 shell.

Running x86 applications from a native POWER shell

Applications can be started directly from a native POWER shell by passing them to the **runx86** command as a parameter. The path to the application must be a subpath relative to the x86 World root (for example, `/bin/ls`). Arguments are passed directly to the x86 application, so any paths that are passed as arguments should also be subpaths below the x86 World root (for example, `/tmp` rather than *X86WORLD_ROOT*/ tmp).

This method has the advantage that it can be set up as a script that an end user runs. The end users do not need to know that they are running the application in a VxE on a POWER machine.

Example: Running applications directly from a native POWER shell

This example translates the x86 World `/bin/ls` binary, and lists the contents of the directory.

You must enter the command from a directory that is visible from x86 World. See "Default escaped directories, files, and sockets" on page 44 in this guide for a list of directories that are automatically visible from x86 World. When invoking an x86 application directly from a native POWER shell, the **runx86** command will not change the current working directory for you automatically to one that is visible from the x86 World.

For example, you can run the following command from a POWER shell:

```
% runx86 /bin/ls /tmp
```

This command creates a VxE, translates the **ls** command, shows the results of the **ls** command, and then closes the VxE.

Note: x86 applications cannot be run directly from a POWER shell without invoking the **runx86** command.

Starting the PowerVM Lx86 daemon

The PowerVM Lx86 translation daemon must be running in order to run x86 applications on the POWER machine.

The PowerVM Lx86 daemon enables x86 processes running within VxEs to communicate with each other. The PowerVM Lx86 installer starts the PowerVM Lx86 daemon on the system after the installation is completed. It also installs a startup script for the PowerVM Lx86 daemon named **/etc/init.d/powervm-lx86** to start the PowerVM Lx86 daemon and any x86 daemons when the system reboots or starts up.

Note: For details of how to run x86 daemons see "Starting x86 daemons with PowerVM Lx86" on page 46.

The PowerVM Lx86 daemon will normally be running on your system after installation. However, to start the PowerVM Lx86 daemon manually, run the **/etc/init.d/powervm-lx86** script as root. The output is as follows:

```
% /etc/init.d/powervm-lx86 start
Starting powervm-lx86-daemon [  OK  ]
Starting x86 services
Entering non-interactive startup
Starting system logger: [  OK  ]
Starting kernel logger: [  OK  ]
Starting crond: [  OK  ]
```

When the PowerVM Lx86 daemon starts, it creates a `/var/opt/powervm-lx86` directory. This directory must exist, and have full read and write permissions for all users, for PowerVM Lx86 to work.

You can now start a simple x86 binary using the **runx86** command. For example, the **ls /** x86 command should give output similar to that shown in the following example, run in a POWER shell:

```
% runx86 /bin/ls /
bin dev home lib mnt proc sbin srv tmp var
boot etc initrd media opt root selinux sys usr
```

PowerVM Lx86 start up script parameters

The PowerVM Lx86 daemon start up script **/etc/init.d/powervm-lx86** takes the following arguments:

- `start`: Checks if the PowerVM Lx86 daemon is started already, if not, it starts the daemon.
- `stop`: Halts the PowerVM Lx86 daemon.
- `restart`: Halts the PowerVM Lx86 daemon, then starts the PowerVM Lx86 daemon again.
- `status`: Reports the current status of the PowerVM Lx86 daemon.

Example of using the PowerVM Lx86 startup script

To stop the PowerVM Lx86 daemon, run the following command in a POWER shell:

```
% /etc/init.d/powervm-lx86 stop
```

To restart the PowerVM Lx86 daemon, run the following command in a POWER shell:

```
% /etc/init.d/powervm-lx86 restart
```

Note:

- You must have root access rights to run the PowerVM Lx86 startup script.
- Stopping the PowerVM Lx86 daemon while any x86 applications or x86 daemons are running will cause them to shut down.

PowerVM Lx86 log files

PowerVM Lx86 creates log files during installation, for the PowerVM Lx86 daemon and any translated x86 processes that have errors. These log files are not automatically deleted, so you may want to clean them out periodically.

The PowerVM Lx86 daemon log file is created in the `/var/opt/powervm-lx86/daemon` directory, and is always given the name powervm-lx86-daemon.log. The powervm-lx86-daemon.log file lists communication failures between the translated x86 applications and the PowerVM Lx86 daemon, and internal errors, such as being out of memory.

Log files are created for x86 processes if they produce an error message, warning message, or fail as they are running within a VxE. The log files are created in the `/var/opt/powervm-lx86/log` directory. The filename takes the form:

powervm-lx86.log.<*process_name*>.<*process_id*>.<*unique_id*>.

The PowerVM Lx86 installer script generates a log file when it is run that contains a verbose record of each session including all screen output, user input and time stamps for each event. When the installer exits, it prints out the path to the log file for future reference. The log file is found in `/tmp/powervm-lx86_install_xxxxxx.log`, where xxxxxx is a unique identifier.

Report any application failures to IBM support. See "Reporting a PowerVM Lx86 failure" for more details.

Viewing the installation log file - logviewer

A script called **logviewer** is provided to allow the PowerVM Lx86 installation log files to be viewed in terminals with non-English locales.

The installation log file is stored in the UTF-8 encoding to support different locales. Although the log file contains the complete character information, it cannot be viewed in a terminal until it has been transcoded into the encoding specified by the current locale.

Description

logviewer reads a PowerVM Lx86 installation log file (stored in the UTF-8 encoding) and transcodes it to the encoding specified in the current locale. **logviewer** behaves in a similar way to **cat**, taking files on stdin or as arguments and returning them to stdout. If you invoke **logviewer** without any arguments it will wait for input from stdin. You may specify the option **--more** or **--less** to have the output piped to the **more** or **less** command line programs respectively.

Location

```
<PowerVM Lx86 install location>/installer/resources/bin/RO/logviewer
```

Usage

logviewer [--help]

logviewer /tmp/powervm-lx86_install_XXXXXX.log

logviewer [--more | --less] /tmp/powervm-lx86_install_XXXXXX.log

Reporting a PowerVM Lx86 failure

If a Linux x86 application fails while being translated, an error is displayed.

In addition, an error log is created in the `/var/opt/powervm-lx86/log` directory. You can change the location of the default log directory during installation. Log files are created for each running process that encounters an error.

Please report any errors to IBM support and include a description of the failure and what events preceded the failure.

Note: The reported error may be caused by an issue with the Linux x86 application being executed and may not be a problem with PowerVM Lx86.

See "PowerVM Lx86 log files" on page 25 for more details about log files generated by PowerVM Lx86. See Chapter 9, "PowerVM Lx86 error messages and resolutions," on page 59 for more information on error messages generated by PowerVM Lx86. See "Known issues with PowerVM Lx86," on page 77 for general limitations which are not associated with an error message.

Monitoring x86 applications

You can monitor running x86 applications using x86 commands.

x86 application monitoring commands displays information about any processes running in a VxE. POWER processes are not shown.

Processes running within a VxE can also be seen from the host POWER system using commands such as the **ps** command and the **top** command. The output is more verbose and shows the translator processes running the x86 applications. These details might not be required if you are only determining which x86 processes are running. However, you might prefer to use a POWER tool that you know is running natively, and use a script to filter out unwanted information.

Following is an example that shows the use of commands to monitor x86 applications. (On this system, the only x86 processes running are **bash** and **ps**).

From a translated x86 shell, enter the following:

```
$ ps -A
```

The output is similar to the following:

```
PID   TTY     TIME     CMD
 3998 ?        00:00:00 syslogd
 4006 ?        00:00:00 klogd
 4033 ?        00:00:00 crond
 4144 pts/2    00:00:00 ps
 4135 pts/2    00:00:00 bash
    1 ?        00:00:01 init
```

From a POWER shell, enter the following:

```
% ps w w ax
```

The output is similar to the following:

```
16097 pts/13   Ss    0:00 -bash
 3932 ?        Ssl    0:00 /opt/powervm-lx86/bin/powervm-lx86-daemon
 3998 ?        Ssl    0:00 /opt/powervm-lx86/bin/powervm-lx86 -D3 -F4 -f3ff -argv0
                           syslogd /i386/sbin/syslogd -m 0
 4006 ?        Ssl    0:00 /opt/powervm-lx86/bin/powervm-lx86 -D3 -F4 -f3ff -argv0
                           klogd /i386/sbin/klogd -x
 4033 ?        Ssl    0:00 /opt/powervm-lx86/bin/powervm-lx86 -D3 -F4 -f3ff -argv0
                           crond /i386/usr/sbin/crond
16252 ?         R+    0:00 ps w w ax
```

Maintaining x86 applications

Maintaining x86 applications includes debugging x86 applications and generating x86 core dump files.

Debugging

When an in-house x86 application is migrated to POWER, developers might need to build or support the application on the POWER system. Developers cannot use native Linux on POWER debugging tools when the application is running in a VxE, because it results in the translator program itself being debugged. Instead, developers should use x86 debugging tools running within a VxE on the POWER machine.

The x86 command line debugging tools gdb, strace, and ltrace are supported within a VxE. It is possible to debug an x86 application from within a translated x86 gdb session. It is also possible to attach a translated x86 gdb to a running x86 process.

Note: Hardware watchpoints are not supported by PowerVM Lx86. When running gdb under translation you may see the following message:

```
"Couldn't write debug register: Input/output error."
```

This message is not fatal and is expected behaviour for PowerVM Lx86.

x86 core dump files

Core dump files are supported for x86 processes running in a VxE. If an x86 process crashes unexpectedly while running within a VxE, it can produce a core dump file. If the crash was caused by a problem with the translator, then an error log will also be generated. The translator can also produce a core dump.

Note: Core dumps of simple applications (single threaded applications and those that do not register signal handlers) may produce inaccurate core files. To generate an accurate core file, set the EXTRA_DEBUG_SUPPORT_FROM_START configuration switch to y in the configuration file located in `/etc/opt/powervm-lx86/config` or as an environment variable and retry the application. This step is not required for the majority of applications. See the examples below.

Example: Enabling accurate core dumps for simple applications

Using the PowerVM Lx86 configuration file, `/etc/opt/powervm-lx86/config`, the configuration variable can be set by adding the following line:

```
EXTRA_DEBUG_SUPPORT_FROM_START=y
```

Using the PowerVM Lx86 environment variable, set:

```
% export LX86_CFG_EXTRA_DEBUG_SUPPORT_FROM_START=y
```

For more details about configurations switches for PowerVM Lx86, see “PowerVM Lx86 configuration settings” on page 36.

Installing and configuring x86 applications on a PowerVM Lx86 system

This section explains how to install and configure x86 applications onto a POWER platform, and how to migrate existing applications from an x86 platform to a POWER platform.

The general approach to migration is to make the x86 application and data accessible from the POWER machine. This typically means installing the application or copying or mounting the application files. No alteration of application binaries is necessary and no conversion of the data is necessary.

The combination of an application and the data that it needs is defined here as a *workload*.

The migration has two parts:
1. Setting up the x86 system configuration.
2. Installing x86 applications.

Setting up the x86 system configuration

Setting up local or remote user authentication, connecting to remote file systems, configuration x86 daemons, and setting up environment variables are part of the set up process that needs configuring on the system or migrating from an from an existing x86 system.

The installation script installs x86 libraries, commands, utilities, and infrastructure files in the x86 World. For more information, see “Installing and updating packages in x86 World” on page 33. You can add additional packages later to x86 World (see Installing and updating packages in x86 World in “Installing and updating packages in x86 World” on page 33). This section summarizes the following areas of the x86 system that might need configuring or migrating:

- Local or remote user authentication
- Remote file systems
- x86 daemon configuration
- Environment variables

Local and remote user authentication

The local users within x86 World are separate from the local users on the POWER system.

The root password for x86 World can be the same as the root password for the POWER system, but the root password for x86 World is stored and maintained in the password files in x86 World.

If an x86 application creates a new user (for example, during installation), then this user is created in x86 World and is not available on the POWER system.

For more details about setting up local users in the x86 World, please see “Managing local users, groups, and passwords with PowerVM Lx86” on page 53.

Remote user authentication is also supported in the x86 World. For details, please see “Managing remote users with PowerVM Lx86” on page 53.

Remote file systems

Non-local file systems can be made accessible from the x86 World. This is done by mounting the file system on the POWER system. You can mount the remote file system directly to a mountpoint within x86 World, or to a mountpoint external to x86 World, and then you use the **linkx86** command from the POWER side to create an escape to the mountpoint. Note that you can only use the **linkx86** command to create an escape on the POWER system and not from within a VxE. For details about creating mountpoints, please see “Installing x86 applications.”

x86 configuration and daemons

x86 daemons are supported in the x86 World. Applications that use daemons will normally install and configure the daemons automatically when the application is installed.

It might be necessary to manually migrate x86 daemons onto the POWER system. Daemons can be automatically started in a VxE on the host system by inserting a suitable script in the *X86WORLD_ROOT*/etc/init.d directory. See “x86 /etc/init.d support scripts” on page 46 for more information on how x86 daemons are handled by PowerVM Lx86.

Installing x86 applications

This procedure describes how to install x86 applications onto a POWER system.

Installing an x86 application onto a POWER system is the next step after you install PowerVM Lx86 and configure the system. Installing x86 applications consists of the following tasks:

- Install, copy, or set access to the x86 application binaries.
- Transfer data, or set access to application data.
- Create access to required devices, such as remote file systems.

Install, copy, or set access to the x86 application binaries

The x86 application binaries must be either installed in x86 World, or made accessible from x86 World.

To install binaries into x86 World, copy them directly into x86 World (or to a location accessible from x86 World through an escape or a mountpoint). If x86 application installation scripts or packages exist, then you can copy them into x86 World, or an accessible location, and run them using the **runx86** command.

You can make application binaries that are already installed on an x86 machine on the network accessible from x86 World either by creating a mountpoint on the POWER system and creating an escape from x86 World to the mountpoint, or by mounting a remote filesystem to a mountpoint within x86 World.

Java™ installation scripts will need the Java runtime libraries installed first (see Installing Java applications in x86 World below), although many ISV applications provide a Java runtime library as part of the installation.

Transfer data, or set access to application data

If an x86 application requires access to specific data, the data must be made available. You can do this by copying the data to a location within x86 World, or making it available through a mountpoint or an escape.

Data stored in files on disk can be transferred between x86 machines and POWER machines without the need for any conversion. Provided the necessary mounts or escapes are created, an x86 application running with a VxE is able to access data stored in files that reside on both x86 and POWER file systems.

The example below shows how to make a CD drive accessible to the x86 World. The same technique can be used to make directories or files available to the x86 World.

Creating mountpoints or access to devices

This procedure describes how to create a mountpoint for the x86 World or access to devices needed by x86 applications. A device, directory or file can be made accessible from x86 World either by creating a mountpoint on the POWER system and creating an escape from x86 World to the mountpoint, or by mounting a remote filesystem to a mountpoint within x86 World, or copying the contents of the directory or files to the x86 World itself.

Create access to required devices

Some applications might require access to specific devices, such as tape drives. It is recommended that access to these devices are set in the Linux on POWER operating system.

Most applications will not require access to specific devices to be configured. Many common devices such as file storage and network interfaces appear to applications as files or directories in the VxE filesystem.

Standard devices that are not accessible by default, such as CD devices, should be mounted from the POWER system directly to a mountpoint within x86 World. Alternatively, you can mount to a mountpoint external to x86 World and then use the **linkx86** command to create an escape to the mountpoint. It is possible to mount the device in the VxE, but this is not the best method.

Following is an example of creating access to a device.

Example: Creating access to a CD drive

In Linux, CD drives appear as devices in the /dev directory, such as the /dev/cdrom drive. The CD drives are accessed natively by mounting to a directory in the file system. To make a CD drive accessible from the VxE, you can:

1. Mount it at a mountpoint inside the x86 World using the POWER **mount** command.
2. Mount it to a directory accessible from x86 World by an escape using the POWER **mount** command.
3. Mount it using the x86 **mount** command from within a VxE.

These three examples are shown below:

Mounting from within a POWER shell into the x86 World:

The following is an example of mounting a directory into the x86 World from a POWER shell, and then listing the mounted directory in the translated x86 shell:

```
% mkdir X86WORLD_ROOT/cdrom
% mount /dev/cdrom X86WORLD_ROOT/cdrom
mount: block device /dev/cdrom is write-protected, mounting read-only
% runx86
$ ls /cdrom
Copyright README installer ...
```

Mounting from within a POWER shell to an escape accessible from x86 World:

The following is an example of mounting a directory for x86 World from a POWER shell, making that directory visible to the x86 World by creating an escape using the **linkx86** command, and then listing the mounted directory in the translated x86 shell:

```
% mkdir /cdrom
% mount /dev/cdrom /cdrom
mount: block device /dev/cdrom is write-protected, mounting read-only
% linkx86 /cdrom
% runx86
$ ls /cdrom
Copyright README installer ...
```

Mounting from within an x86 shell:

The following is an example of starting an x86 shell from a POWER shell, and then mounting a directory in the translated x86 shell:

```
% runx86
$ mkdir /cdrom
$ mount /dev/cdrom /cdrom
mount: block device /dev/cdrom is write-protected, mounting read-only
$ ls /cdrom
Copyright README installer ...
```

For further details about creating mountpoints and some potential issues with creating mountpoints within an x86 shell, please see the following section “Issues using mount with PowerVM Lx86.”

Issues using mount with PowerVM Lx86

Special care must be taken when using the mount command in the x86 World and also when using the **mount** command on the POWER system to make devices or directories accessible to x86 World. In particular, special care must be taken when unmounting these devices or directories with the **umount** command.

Using mount from an x86 shell or application

If a directory or device is mounted from within a translated x86 shell, then the directory must only be unmounted from within a translated shell.

Unmounting the directory in a POWER shell will cause the unmount to occur, but the device may still appear to be mounted in the x86 World. This must then be explicitly cleared.

Using mount from a POWER shell

If a directory or device is mounted from a POWER shell, then the directory must be unmounted from a POWER shell only.

Using NFS mounts within a translated x86 shell or application

Mounting NFS folders from an x86 shell is not supported. Mount the NFS folders from a POWER shell into the x86 World or to a mountpoint that is visible to the x86 World.

Installing Java applications in x86 World

This procedure describes how to install x86 Java applications onto a POWER system.

PowerVM Lx86 can run Java applications on a POWER system. Java applications are run using an x86 Java Virtual Machine (JVM) that is running in the VxE.

Note: The correct x86 Java Runtime Environment (JRE) must be installed in x86 World before Java applications can be run.

It is possible to have an entirely different POWER JRE installed on the host system that is running PowerVM Lx86. These libraries do not affect the x86 Java applications.

Configuring x86 World for Java

The only configuration required is for the JRE to be installed in x86 World, just as on a native x86 machine. Typically this means that it is installed in the *X86WORLD_ROOT*/usr/bin/ directory. Note that the x86 Java must be installed from an x86 shell.

Example: installing IBM's J2SE 1.4 runtime binaries:

1. Download the IBMJava2-142-ia32-JRE-1.4.2-8.0.i386.rpm file from http://www.ibm.com/developerworks/java/jdk/linux/download.html to *X86WORLD_ROOT*.
2. Start an x86 shell by entering the following command in a POWER shell:
```
% runx86
```
3. Install the RPM by running the following command in the translated x86 shell:
```
$ rpm -ivh IBMJava2-142-ia32-JRE-1.4.2-8.0.i386.rpm
```
4. Follow the installation prompts. Remember that the x86 shell is jailed, so the *X86WORLD_ROOT*/usr/bin/ directory appears as /usr/bin/.

Running Java applications

Java applications are run using the **runx86** command, just like any other x86 applications. No special switches or system daemons are required.

Chapter 7. Configuring and maintaining the x86 World and PowerVM Lx86

This reference section provides details about configuring and maintaining the x86 World and configuring PowerVM Lx86.

The reference includes:
- Information about upgrading PowerVM Lx86.
- Installing and updating packages in the x86 World.
- PowerVM Lx86 configuration settings.
- Details of default escapes and virtual files in the x86 World.
- Installing and running x86 daemons.
- SE Linux support for x86 applications.

Installing and updating packages in x86 World

This section describes how to manage the software packages within the x86 World. As with any normal system, refer to your system administrator for advice and best practices before adding and upgrading software in the x86 World.

The x86 World using PowerVM Lx86 on a POWER system should be managed just as if it were an independent x86 system. The x86 World contains a set of x86 libraries, command line tools, applications and other system files, just like a native Linux on x86 file system. You can install new packages and update existing packages using the standard x86 package management tools, such as RPM. More advanced package management tools, such as system-config-packages (RHEL), up2date (RHEL), and YaST2 (SLES) are also supported.

The following sections provide detailed instructions for installing packages on RHEL4 and SLES10 systems. For other distributions please follow the equivalent procedure as described by your Linux distributor.

Note: When updating packages in the x86 World, ensure that you do not update them to a newer version of the Linux distribution than the underlying POWER system. For example, if you are running Red Hat Enterprise Linux 4.6 on the POWER system, ensure that you upgrade the POWER system to Red Hat 4.5 before upgrading the x86 World to that version. See "System and installation requirements for PowerVM Lx86" on page 10 for more details on which x86 World Linux distribution versions are supported under PowerVM Lx86 with which POWER operating system versions.

Installing and updating packages for Red Hat (RHEL 4) in x86 World

For Red Hat, the up2date tool is recommended for managing packages. You can use up2date to install new packages and download updates from the internet via the Red Hat Network.

In addition to up2date, the rpm and system-config-packages tools can also be used to add packages to the x86 World. The system-config-packages tool is not installed by default in a minimum x86 World install.

Configuring up2date for the first time

1. Enter the *X86WORLD_ROOT*, for example by entering the following command in a POWER shell:
   ```
   % cd /i386
   ```
2. Run PowerVM Lx86 by entering the following command in a POWER shell:

```
% runx86
```

3. Become root in a translated x86 shell with the command:
   ```
   $ su
   ```
4. Run the up2date tool in the translated x86 shell with the command:
   ```
   $ up2date --config
   ```
   If you need access to the Internet via a proxy, enter it under httpProxy (option 11) and then enable the proxy (option 3). Save the settings by pressing Enter.
5. If you see a prompt to install a GPG key, do that by entering the following command in the translated x86 shell:
   ```
   $ rpm --import /usr/share/rhn/RPN-GPG-KEY
   ```

Registering the system with the Red Hat Network (RHN)

This process only needs to be done once per PowerVM Lx86 installation.

1. Become root in a translated x86 shell by typing:
   ```
   $ su
   ```
2. Run the up2date in the translated x86 shell tool by typing:
   ```
   $ up2date
   ```
   Follow the prompts on the screen. Enter your Red Hat Network registration details. Once complete, you will see the message: "You have successfully registered this System Profile on Red Hat Network."

Adding packages to the x86 World

The up2date tool is used to add packages and their dependencies into the x86 World.

1. Become root in a translated x86 shell by typing:
   ```
   $ su
   ```
2. To install a package, use the –i command line option to up2date. For example, to install gcc (and its dependencies) by entering the following command in the translated x86 shell:
   ```
   $ up2date –i gcc
   ```

Updating packages within x86 World

The up2date tool can also update packages within the x86 World.

1. Become root in a translated x86 shell by typing:
   ```
   $ su
   ```
2. Perform an online update of the x86 World by entering the following command in the translated x86 shell:
   ```
   $ up2date --update
   ```

Installing and updating packages for Novell SLES 10 in x86 World

For Novell SLES 10, the recommended method for managing (adding and updating) packages is to use the YaST tool.

A media source must be set up to allow YaST to manage packages in the x86 World. The media source contains the SLES 10 Linux distribution ISO images from which YaST can access all the x86 packages. The media source must be either on the local file system or on a shared server.

Accessing the SLES 10 media source

The instructions assume that the media source has been created on a shared server (fileserver) which can be accessed by each system that needs additional packages installing.

Copy the SLES 10 Linux distribution ISO images into a suitable directory on the shared server. The instructions assume the ISO images are located in `/fileserver/isos/sles10x86`.

The shared server is assumed to already have been mounted on the POWER system to the `/fileserver` directory.

First, ensure the `/fileserver` directory is accessible from within the x86 World:

1. Become root in a POWER shell by typing:
   ```
   % su
   ```
2. Run the **linkx86** command in the POWER shell to create the escape to `/fileserver`:
   ```
   % /usr/local/bin/linkx86 /fileserver
   ```

Next, verify that the shared server is accessible from with the x86 World:

1. Enter the *X86WORLD_ROOT*, by entering the following command in a POWER shell:
   ```
   % cd /i386
   ```
2. Run PowerVM Lx86 by entering the following command in a POWER shell:
   ```
   % runx86
   ```
3. List the contents of the media source by typing entering the following command in the translated x86 shell:
   ```
   $ ls /fileserver/isos/sles10x86
   ```

   The output from this command should list the ISO images on the shared server. If not, check that the shared server is accessible from a POWER shell and check the steps above.

Configuring YaST to access the media source

Now that the media source is accessible in the x86 World, the next step is to configure YaST to access the media source so that it can find the x86 packages.

1. Become root in a translated x86 shell by typing:
   ```
   $ su
   ```
2. Run YaST by entering the following command in a POWER shell:
   ```
   $ yast
   ```

   The YaST Control Centre will start and you will see a graphical text screen.
3. Select **Software** from the main menu on the left, and press Enter to confirm.
4. Select **Change Source of Installation** from the Software list on the right, and press Enter to confirm.
5. Press Tab to select the **Add** menu and press Enter to confirm.
6. Press Tab to select **Local Directory** from the list, and press Enter to confirm.
7. Press Tab to select **ISO image**, and press Enter to confirm.
8. Press Tab to select **Browse**, and press Enter to confirm.
9. Navigate to the SUSE SLES10 ISO images (`/fileserver/isos/sles10x86`) from the browse list using the Tab and Arrow keys and use the Enter key to highlight your selection.
10. Press Tab to select **Ok**, and press Enter to confirm. 12.
11. Press Tab to select **Next**, and press Enter to confirm.
12. A License agreement will be displayed. Press Tab to select **Yes** if you agree to the license, and press Enter to confirm.

13. Press Tab to select **Next**, and press Enter to confirm.
14. Add additional media sources if required by repeating steps 5-13 or press Tab to select **Finish**, and press Enter to confirm.

Managing packages

Now that YaST is aware of the media sources, it is possible to add or update packages in the x86 World.

1. From the YaST Software menu, select **Software Management**, and press Enter to confirm.
2. Press Tab to select **Filter**, then **Search**.
3. In the **Search Phrase** field, enter the name of the package you want to install, for example, `gcc`.
4. In the list of available packages, use the arrow keys to navigate and use the enter key to select packages you want to install. Package dependencies will be automatically resolved.
5. Repeat the previous two steps for all packages you want to install.
6. After you have selected all the packages you want to install, use the Tab key to select the **Accept** button, and press Enter to confirm. A prompt to display the resolved dependencies may appear; press the enter key to confirm. YaST will now install the selected packages.
7. At the **Install or remove more packages** prompt, press Tab to select **No**, and press Enter to confirm.
8. After the installation is complete, you can exit YaST by using Tab to select **Quit** and pressing Enter to confirm.

PowerVM Lx86 configuration settings

PowerVM Lx86 can be configured using various switches. The configuration switches supply parameters to the translator and change aspects of the translator's run time behavior. The configuration switches can be set using a configuration file or by setting environment variables.

PowerVM Lx86 configuration file

The configuration file is located on the POWER system at: `/etc/opt/powervm-lx86/config`. Configuration switches can be added to the configuration file. The configuration switches are checked whenever a new process or application is started from an x86 shell. Changes made to the configuration switches will not affect processes that are already running.

The configuration switches take the following form:

```
<CONFIGURATION_SWITCH>=<VALUE>
```

Notes:

- Each configuration switch must be entered on a separate line in the configuration file.

By default, no configuration file is present when you install PowerVM Lx86. A configuration file will only be created by the PowerVM Lx86 installation process if non-default options are chosen for any one of the following installation options:

- PowerVM Lx86 directory
- PowerVM Lx86 log file directory
- x86 World directory

Creating a configuration file

If a configuration file is not present on the system, one can be created using a standard text editor. Save the file as: `/etc/opt/powervm-lx86/config`

Example configuration file

This is an example of the contents of a configuration file for a PowerVM Lx86 installation with the non-default PowerVM Lx86 directory: `/mylx86/install-location` and the non-default log file location: `/var/mylx86logs/log`

```
POWERVM_LX86_LOCATION=/mylx86/install-location
LOGFILE_PATH=/var/mylx86logs/log
LOCALISATION_FILES_DIR=/mylx86/install-location/locale
```

Note: The LOCALISATION_FILES_DIR configuration switch is set when a non-default PowerVM Lx86 directory is chosen.

Configuration file switches

Switches affecting installation directories and log files:

Configuration switch name	Parameters and use
POWERVM_LX86_LOCATION	**Type** String **Parameters** An absolute path **Use** This switch specifies the PowerVM Lx86 installation directory. This should only be updated by the PowerVM Lx86 installer.pl script or set in the configuration file for non-interactive installation, see Chapter 5, “Running the automated installer,” on page 19.
LOGFILE_PATH	**Type** String **Parameters** An absolute path **Use** This switch specifies the PowerVM Lx86 log file directory. If this is changed manually, then a log file directory must exist with the correct permissions (owner and group 'root' and set to 01777) before PowerVM Lx86 will run correctly. This defaults to `/var/opt/powervm-lx86/log` when no configuration switch is set. For non-interactive installations, this value can be set before installation.
SUBJECT_WORLD_ROOT	**Type** String **Parameters** An absolute path **Use** This switch specifies the x86 World directory. This should only be updated by the PowerVM Lx86 installer.pl script or set in the configuration file for non-interactive installation, see Chapter 5, “Running the automated installer,” on page 19. The defaults to `/i386` when no configuration switch is set.
LOCALISATION_FILES_DIR	**Type** String **Parameters** An absolute path **Use** This switch specifies the PowerVM Lx86 localization files directory. This should only be updated by the PowerVM Lx86 installer.pl script and will be a subdirectory of the directory set by POWERVM_LX86 called `locale`.

Configuration switch name	Parameters and use
CATCH_CRASHES	**Type** Boolean **Parameters** `y` or `n` **Use** The default is `y`. Setting this to `n` will suppress the creation of PowerVM Lx86 log files on the filesystem. Error messages will still be displayed to the screen.
CATCH_CRASHES_SILENT	**Type** Boolean **Parameters** `y` or `n` **Use** The default is `n`. Setting this to `y` will suppress any output to the screen in the even of a crash. Log files will still be generated to the filesystem unless CATCH_CRASHES is also set to `n`.

Switches affecting networking support:

Note: See "Default PowerVM Lx86 escapes and virtual files in the x86 World" on page 42 for more details about the use of these configuration switches.

Configuration switch name	Parameters and use
HAVE_SEPARATE_RESOLV_CONF_FILES	**Type** Boolean **Parameters** `y` or `n` **Use** The default is `n`. Setting this to `y` will cause PowerVM Lx86 to manage the x86 World resolv.conf file separately from the POWER system version of the file. This is an option for advanced users only.

Switches affecting the local user management support:

Note: See "Managing local users, groups, and passwords with PowerVM Lx86" on page 53 for more details about using these configuration switches, in particular the parameters for WORLD_CHECK_OR_SYNC.

Configuration switch name	Parameters and use
HAVE_SEPARATE_PASSWORDS	**Type** Boolean **Parameters** `y` or `n` **Use** The default is `n`. Setting this to `y` will cause PowerVM Lx86 to favor the x86 World entries for passwords, rather than the entries in the POWER system files.

Configuration switch name	Parameters and use
MERGE_PASSWD_FILES	**Type** Boolean **Parameters** `y` or `n` **Use** The default is `y`. Setting this to `n` will cause PowerVM Lx86 to manage the `/etc/passwd`, `/etc/group` and `/etc/shadow` (and on Red Hat systems there is also `/etc/gshadow`) files completely separately from the native POWER system. This is not recommended unless it is explicitly known that there are no security risks involved in doing so.
WORLD_CHECK_OR_SYNC	**Type** String **Parameters** `sync_all, check_all, check_passwd, check_group, force_sync_mtab, none` **Use** The default is check_all. This switch defines whether the *user-ID cron* job, which checks for differences between the user files in the x86 World and the POWER system, runs on the system. Setting this to none will disable the *user-ID cron* job and the system administrator will not be notified of any conflicts between the user files in each environment.

Switches affecting SE Linux support on RHEL systems:

Note: See section "SE Linux support with PowerVM Lx86" on page 47 for more details on how to enable and use SE Linux with PowerVM Lx86.

Configuration switch name	Parameters and use
ENABLE_SELINUX_TRANSITIONS	**Type** Boolean **Parameters** `y` or `n` **Use** The default is `n`. If SE Linux is enabled on your POWER system, you can enable SE Linux for x86 applications by setting this switch to `y`. If SE Linux is not enabled or installed on your POWER system, enabling this switch will have no effect.

Switches affecting floating-point accuracy of PowerVM Lx86:

Configuration switch name	Parameters and use
X87_PRECISION_TYPE	**Type** String **Parameters** `64BIT, 80BIT, 64BIT_PLUS` **Use** The default is `64BIT_PLUS`. PowerVM Lx86 provides different modes of accuracy for x87 floating-point math. These modes are 64-bit, 80-bit, or 64-bit plus. The 64-bit mode matches the precision of the underlying POWER processor and has high performance. The 80-bit mode matches the higher precision of the x87 processor and has lower performance. The default 64-bit plus mode aims to provide a hybrid mode, which has the performance of the 64-bit mode, but improves the accuracy to be closer to the 80-bit mode. In the 64-bit plus mode, most calculations are carried out using 64-bit math, but explicit 80-bit math will switch to 80-bit mode, before switching back to the 64-bit mode.

Switches controlling the virtual memory usage of PowerVM Lx86:

The amount of virtual memory used by PowerVM Lx86 can be restricted, specified as a ratio to the amount of memory used by the x86 application being translated. When PowerVM Lx86's memory use exceeds this ratio, it will discard POWER code translations until it is within the allowed threshold.

The limit is set to 250% of the x86 application's memory usage by default; that is PowerVM Lx86 will not use more than two and a half times the amount of memory that the x86 application would have used. Setting the value to 100% will cap the memory usage when it reaches 1:1 with the x86 application usage. Setting the ratio to zero will disable the limit. Setting the limit to a low ratio, less than the default, may impact the performance of PowerVM Lx86 and hence the x86 application being translated.

Configuration switch name	Parameters and use
MEMORY_MONITOR_TRIGGER_RATIO	**Type** Integer **Parameters** `Integer` **Use** The default is `250`, limiting the virtual memory usage of PowerVM Lx86 to 250% (or 2.5 times) the memory usage of the x86 application binaries. To disable the Lx86 virtual memory limit, set the value to `0`.

Switches affecting gdb when it is being translated by PowerVM Lx86:

Note: See section "Maintaining x86 applications" on page 27 for more details about debugging x86 applications.

Configuration switch name	Parameters and use
EXTRA_DEBUG_SUPPORT_FROM_START	**Type** Boolean **Parameters** y or n **Use** The default is n. Setting this to y will cause PowerVM Lx86 to generate accurate core files for simple x86 applications (single threaded applications and those that do not register signal handlers). This step is not required for the majority of applications.

Switches affecting the visibility of the POWER system from within the VxE:

Configuration switch name	Parameters and use
SUPPORT_TARGET_PROC	**Type** Boolean **Parameters** y or n **Use** The default is n. Setting this to y will allow PowerVM Lx86 processes to see all processes in the system via /proc. Utilities such as ps will see all POWER processes. This step is not required for the majority of applications. If you are using x86 system monitoring tools it may be useful.

Switches affecting the environment of x86 processes:

Configuration switch name	Parameters and use
LD_PRELOAD_OVERRIDE	**Type** string **Parameters** A whitespace separated list of preload libraries to be used for all x86 processes. **Use** The default is /usr/local/lib/memcpy.so on POWER7 systems and empty (not set) on POWER6 systems. The preload default on POWER7 replaces the x86 memory copy routines with versions suitable for a POWER system. To disable this behavior add this switch to the configuration file with an empty value. This switch sets the LD_PRELOAD variable in the environment of x86 processes.

Setting configuration switches using environment variables

In addition to setting configuration switches in the configuration file, configuration switches can also be set using environment variables. The configuration switches must be set in a POWER shell before invoking PowerVM Lx86 with the **runx86** command. Some configuration switches affect the global characteristics of PowerVM Lx86 and cannot be set for each shell using environment variables. The configuration switches that can be set using environment variables are listed in the following table:

Table 1. Configuration switches that can be set using environment variables

Configuration switches that can be set using environment variables
EXTRA_DEBUG_SUPPORT_FROM_START
X87_PRECISION_TYPE

To set a configuration switch using environment variables, the name of the switch must be preceded with `LX86_CFG_` to ensure that PowerVM Lx86 detects that the configuration switch has been set in the environment. Set the configuration switch in the environment of a POWER shell and invoke **runx86** with the following commands in the POWER shell:

```
% export LX86_CFG_<CONFIGURATION_SWITCH>=VALUE
% runx86
```

Any applications that are run from this shell will have this configuration switch set.

To run another x86 application with the default configuration settings, either start a new POWER shell and invoke a new instance of PowerVM Lx86 with the **runx86** command or unset the switch in the current shell. To do this, exit from the translated x86 shell and return to the POWER shell where the environment variable was originally set, unset the value and then reinvoke **runx86** with the following commands in the POWER shell:

```
% unset LX86_CFG_<CONFIGURATION_SWITCH>
% runx86
```

Different x86 applications can be run concurrently with different configuration switches set by invoking them from different POWER shells with the environment variables set to the appropriate values.

Default PowerVM Lx86 escapes and virtual files in the x86 World

This section describes how virtual files are handled and how they are useful for system administrators managing a system with PowerVM Lx86 installed. In addition, the PowerVM Lx86 installer sets up some escapes by default that allow PowerVM Lx86 access to certain directories, files, and sockets on the POWER system.

Virtual files and directories

PowerVM Lx86 maintains some system files in x86 World as virtual files.

Passwd, group, and shadow files

See "Managing local users, groups, and passwords with PowerVM Lx86" on page 53 for information on how PowerVM Lx86 manages user administration files within the x86 World. PowerVM Lx86 manages the files in the following table:

Name	Type of virtual file	Directory (D), File (F), or Socket (S)
/etc/passwd	Merged	F
/etc/group (optional during installation)	Merged	F
/etc/gshadow (RHEL only)	Merged	F
/etc/shadow	Merged	F

These files are virtual, and the contents are managed by PowerVM Lx86. In addition, physical files exist for these in the x86 World. The virtual file is a merged file, meaning that the contents are generated by merging the contents of the x86 World physical file and the POWER system version of the file.

If the files are written by a translated x86 application or shell, the underlying physical file in the x86 World will be updated. The change will be visible in the combined virtual view generated by PowerVM Lx86. In addition, any changes in this file will be detected by the user ID cron job, and the system administrator may be notified of any inconsistencies between the x86 World file and the POWER system file. See “Managing local users, groups, and passwords with PowerVM Lx86” on page 53 for more information.

utmp and wtmp

Name	Type of virtual file	Directory (D), File (F), or Socket (S)
/var/run/utmp	Contents as POWER system	F
/var/log/wtmp	Contents as POWER system	F

Both of these files are created during installation of the x86 World. In addition, physical files exist for these in the x86 World. In this case, the physical files are just stubs and will not be updated by any translated x86 application or shell. The virtual files have exactly the same contents as the equivalent files on the POWER system.

Writing to these from a translated x86 application, or while in a translated x86 shell causes an update to the file on the POWER system. There is effectively only one version of the file on the entire system – the one on the POWER system.

Updates to the POWER system version of the file are reflected in the virtual view of the file from within the x86 World.

If the physical x86 World files are opened from a non-translated shell (for example, by running vi `/i386/var/run/utmp`), then any changes are written back to the physical stub, but PowerVM Lx86 ignores the changes, and they are not visible to translated applications. They will only see the contents of the virtual file.

Do not delete the stub files. If you delete the utmp or wtmp x86 stub files this will normally have no effect, but it could cause a failure in applications that attempt to find the files, since the files will not appear in a directory listing.

Note:

- Although deleting the stub files is not recommended, if you want to delete them, it is only possible to delete either of the stub files using a native POWER shell (for example, rm /i386/var/run/utmp). If you attempt to delete the files from a translated x86 shell, it will fail.
- Editing the stub files from an x86 shell is not recommended. They are binary files and contain specific data structures. Editing the files manually may corrupt the data structures of the underlying POWER versions of the files.

/etc/resolv.conf

Name	Type of virtual file	Directory (D), File (F), or Socket (S)
/etc/resolv.conf	Contents as POWER system	F

This file is virtual, PowerVM Lx86 manages the contents, and it is the same as the POWER system version of the file. In addition, a physical file exists in the x86 World. The contents of the physical file are an exact copy of the POWER system version of the file at the point when PowerVM Lx86 is installed.

The virtual file can be read from a translated x86 application, or while in a translated x86 shell, but it cannot be written to. If you need to update the contents of the file, you must update the POWER system version of the file. Once the file has been updated, the change is seen by applications in the x86 World.

It is possible for PowerVM Lx86 to manage the x86 World `/etc/resolv.conf` file independently of the POWER system version in advanced mode. In this case, the physical file is visible to translated x86 applications and from a translated shell.

HAVE_SEPARATE_RESOLV_CONF_FILES=y
: The configuration switch to enable advanced mode. This causes the contents of the x86 file to be used. Any changes to the x86 file will be seen by translated x86 applications. In this mode, if you change the POWER file, this will not be seen by translated applications.

HAVE_SEPARATE_RESOLV_CONF_FILES=n
: To disable advanced mode, either set the switch to this or remove the line from the configuration file. The contents of the POWER file will be visible to translated x86 applications. The x86 file will not be affected.

If you try to delete the `/etc/resolv.conf` file from an x86 shell you will get an error. The x86 version of the file must exist so that it can be opened, but its contents are virtual and appear the same as the POWER file. The x86 version of the file can be deleted from a POWER shell (for example, rm /i386/etc/resolv.conf). Once deleted, the file will not be visible to x86 applications. After deleting the x86 file, the user can recreate the file from a POWER shell. The contents of the x86 file are ignored and will always have the contents of the POWER file.

/proc

Name	**Type of virtual file**	**Directory (D), File (F), or Socket (S)**
/proc	Entirely virtual	D

The `/proc` directory is created by the PowerVM Lx86 installer. There are no physical files in the `/proc` directory. If inspected from a non-translated shell, the directory appears to be empty (for example, ls /i386/proc). The entire contents of the `/proc` directory are virtual and managed by PowerVM Lx86 to represent the detailed processor and system information of an x86 platform. The specific contents of the `/proc` tree vary depending on the specific OS distribution that is installed for the x86 World and on the underlying POWER system.

Some of the directories and files in `/proc` are very specific to the x86 hardware and are not supported by PowerVM Lx86. PowerVM Lx86 will deny access if any attempt is made to access these directories and files.

Default escaped directories, files, and sockets

Directories and files are created as escapes when PowerVM Lx86 is installed. These are required for PowerVM Lx86 to function.

x86 World path or filename	**Directory (D), File (F), or Socket (S)**
/dev	D
/home (optional during installation)	D
/media	D
/mnt	D

x86 World path or filename	Directory (D), File (F), or Socket (S)
/selinux (RHEL only)	D
/sys	D
/var/yp/binding	F

As an example, `/home` in the x86 World (the directory itself is visible as `/i386/home` from a POWER shell) is escaped to `/home` on the POWER system. This means that x86 and POWER applications share home directories on the system. See "Jailing and escapes for PowerVM Lx86" on page 5 for more details.

Syslog support

Support for logging system messages is handled in a special manner for PowerVM Lx86 so that system messages in the x86 World and the POWER system are kept separate.

Kernel messages are only logged by the POWER system and are found in `/var/log/messages` by default. System messages generated by POWER applications are also logged there by default. System messages generated by translated x86 applications are logged in `/var/log/messages` in the x86 World (for example, `/i386/var/log/messages` from a POWER shell).

System logging background

It is possible to run the system logging daemons within x86 World, but the sockets and files they use are handled specially by PowerVM Lx86. The `/etc/init.d/syslog` script starts two daemons: **klogd** and **syslogd**. The **klogd** daemon is responsible for collecting any messages that have come directly from the kernel. It can do this by either reading `/proc/kmsg` (default) or by making the **syslog** system call. If there is no data to read in `/proc/kmsg` **klogd** blocks and waits for data to appear. When **klogd** gets messages from the kernel, it passes them to the **syslogd** daemon via the `/dev/log` socket. The **syslogd** daemon waits on the `/dev/log` socket for data which can come from either **klogd** or directly from a user program such as **initlog** or **logger**. The messages are then written to the `/var/log/messages` file.

System logging with PowerVM Lx86

In the x86 World, the **syslog** system call, `/proc/kmsg` file, and `/dev/log` files are handled in a special manner. If a translated x86 **klogd** process tries to read data from `/proc/kmsg`, PowerVM Lx86 will never read `/proc/kmsg`, but instead blocks the process by never returning any data. Therefore, kernel messages are not logged into the x86 World `/var/log/messages` file.

Kernel messages are only logged by the POWER system and will be found in `/var/log/messages` by default.

Any translated x86 processes such as **syslogd** that perform operations on the `/dev/log` socket will work as normal, however, PowerVM Lx86 will not open `/dev/log`, but instead will open the file `/var/opt/powervm-lx86/devLog` instead. All operations on `/dev/log` will map directly to the socket file `/var/opt/powervm-lx86/devLog`. If an x86 process tries to delete `/dev/log` it will actually correspond to deleting `/var/opt/powervm-lx86/devLog`. The system logs will be written to `/var/log/messages` in the x86 World (for example, `/i386/var/log/messages` from a POWER shell). Only messages from translated x86 applications are logged in the `/var/log/messages` file in the x86 World. All other messages are logged in `/var/log/messages` on the POWER system. This ensures that translated x86 processes cannot collect messages either from the kernel or other POWER processes.

Note: The `/dev` directory is escaped by default in the x86 World to the `/dev` directory on the POWER system. The `/dev/log` file is a special case and is not escaped.

x86 World log socket	Maps to	Directory (D), File (F) or Socket (S)
/dev/log	/var/opt/powervm-lx86/devLog	S

Starting x86 daemons with PowerVM Lx86

You can run x86 daemons in the VxE.

The PowerVM Lx86 **/etc/init.d/powervm-lx86** script starts any x86 daemons in x86 World after starting the PowerVM Lx86 daemon.

The script runs whenever the POWER system is booted or the runlevel changes to 2, 3, or 5. The script triggers the rc script for the x86 distribution (rc is responsible for starting and stopping services on runlevel changes) and starts the appropriate x86 services according to the current runlevel of the POWER distribution. For example, if the current runlevel on the POWER distro is 3, the x86 distro will start the x86 services configured for runlevel 3.

x86 /etc/init.d support scripts

On a Linux system, the `/etc/init.d` directory contains initialization and termination scripts for setting up subsystems or start-stop services.

Introduction

Every kernel runlevel has a corresponding directory in `/etc/rc{0-6}.d` (for example, rc0.d, rc1.d etc.) where symbolic links are created to the scripts located under `/etc/init.d/`. When the system boots, reboots, or there is any other runlevel change, the symbolic links are called to start and stop services.

The scripts are called by the rc script according to a priority number and script name. In a system with PowerVM Lx86 installed, there are two sets of init.d scripts. One set corresponds to the host POWER system, and the other set corresponds to the x86 World.

PowerVM Lx86 contains a number of utilities that enable the execution of init.d scripts in the x86 World with every runlevel change on the host system. This infrastructure allows PowerVM Lx86 to start x86 services with init.d scripts in the same way as if they were running on the original x86 platform in a completely transparent way for the system administrator.

If a newly installed x86 application adds entries to the x86 World init.d scripts, they are handled correctly, with the new services being started or stopped when the system boots, reboots, or when the runlevel changes, or when the PowerVM Lx86 daemon is started or stopped manually.

Implementation

A new installation of PowerVM Lx86, and corresponding x86 World, only have a limited number of services enabled. These are only dbus and syslog. During installation of the x86 World, once all the x86 RPMs are installed, the installer clears the `/etc/rc{0-6}.d` directories, leaving only the necessary services for initial operation of PowerVM Lx86.

After installation, the administrator has the opportunity to start any installed service using the normal utilities provided with the distribution. These typically include chkconfig and other distribution specific tools, such as SUSE's YaST2. During normal operation, every time the POWER system boots, reboots, or changes runlevel, a PowerVM Lx86 script is run to trigger the execution of the corresponding init.d scripts in the x86 World. This PowerVM Lx86 script acts as a wrapper for the rc script in the x86 World. These PowerVM Lx86 scripts are named `powervm-lx86-rc{2-5}`, and are installed in the `/etc/init.d` directory in the host system.

Some of the services that normally run on an x86 environment are not necessary in a PowerVM Lx86 x86 World. This implementation checks that these unnecessary services, or any services that conflict with already running POWER services, are disabled. In addition, SUSE allows dependencies to be defined between the scripts. Therefore, as part of the installation process and later as part of the maintenance tasks, a dependency checker will be run to edit the init.d scripts and delete any known unnecessary dependencies (for example, boot.*, acpid, haldemon, and so on).

The perl script that edits the init.d scripts is named `dependency_checker.pl`, and the installer calls it as the final step in the installation process, and during normal operation when a change in the x86 World `/etc/init.d` directory is detected. Notification of changes in the x86 World `/etc/init.d` directory are sent to a directory monitor called rc_monitor. This program receives events when new files are added or when the permissions are changed on existing files. These events trigger the execution of the `dependency_checker.pl` script.

The directory monitor, rc_monitor, is installed in `/etc/init.d/powervm-lx86-rcmonitor`. Although this is invoked automatically by the `/etc/init.d/powervm-lx86` script (which also invokes the powervm-lx86-daemon), the rc_monitor can be controlled independently of the `/etc/init.d/powervm-lx86` script by running `/etc/init.d/powervm-lx86-rcmonitor` manually.

The powervm-lx86-rcmonitor has the following usage options:

```
/etc/init.d/powervm-lx86-rcmonitor [start|stop|force-reload|restart|status]
```

SE Linux support with PowerVM Lx86

SE Linux is supported by PowerVM Lx86 when running on RHEL. This section describes how to enable SE Linux and how to configure security polices to work with PowerVM Lx86.

Overview of SE Linux support in PowerVM Lx86

As of version 1.3, PowerVM Lx86 provides support for running translated x86 applications in restricted SE Linux domains on Red Hat Enterprise Linux (RHEL) systems. This chapter provides an overview of this support, including instructions for enabling it and information on creating custom policies for x86 applications. Some familiarity with SE Linux concepts is assumed in this chapter; for a more complete description of SE Linux itself, please refer to your Red Hat documentation.

Prior versions of PowerVM Lx86 ran all processes in the unconfined_t domain, including those processes for which more restricted domains were defined in the SE Linux policy. For version 1.3 and above, this remains the default behaviour; after installing PowerVM Lx86 for the first time all x86 processes will continue to run without domain-specific SE Linux restrictions. By installing PowerVM Lx86 extensions to the standard SE Linux policies, x86 processes can be restricted to domains, and the SE Linux features of the underlying Linux kernel will enforce these restrictions. For guidance on installing and enabling these policy extensions and configuring PowerVM Lx86 to use them, please refer to the next section.

Due to certain additional requirements of the PowerVM Lx86 translator, including the need to communicate with the PowerVM Lx86 daemon, translated x86 processes are not able to run in the restricted domains provided by the standard SE Linux policies. In order to support SE Linux, for each domain in the base policy, an equivalent PowerVM Lx86 domain has been created, which permits each domain access to a small number of additional operations as required by the PowerVM Lx86 translator. These extra domains, and the supporting policy rules, are distributed in the PowerVM Lx86 rpm in both binary and source form. When these policy extensions are installed, the PowerVM Lx86 translator will adhere to the SE Linux policy when the `ENABLE_SELINUX_TRANSITIONS` switch is turned on in the PowerVM Lx86 config file.

For example, the syslog daemon, syslogd, will run in the syslogd_t domain on an SE Linux enabled Red Hat system. When SE Linux is enabled for PowerVM Lx86 and the x86 syslog daemon is started it will run in the lx86_syslogd_t domain, which is defined in the policy extensions provided with PowerVM Lx86. This domain is very similar to the normal syslogd_t domain, but it also allows for the additional requirements of the translator.

As well as extra PowerVM Lx86 domains to support translated x86 processes, the PowerVM Lx86 daemon is also run in its own restricted domain, lx86_t. Note that only the domains in Red Hat's targeted policies are supported by PowerVM Lx86. Although it is possible to run PowerVM Lx86 in more comprehensive policies, such as the strict policy, no additional policy extensions are provided or supported with PowerVM Lx86. Note also that during installation of an x86 World, such as when Lx86 is first installed, the PowerVM Lx86 SE Linux support must be disabled. The installer will not allow x86 World installation to proceed with SE Linux support enabled in PowerVM Lx86. To disable PowerVM Lx86's SE Linux support, simply set `ENABLE_SELINUX_TRANSITIONS=n` in the config file.

Enabling SE Linux in PowerVM Lx86

To enable support for SE Linux in PowerVM Lx86, some additional steps are required. PowerVM Lx86 supplies a prebuilt binary SE Linux policy for each supported version of RHEL 4, and a set of binary policy modules for RHEL 5. Additionally, policy source files are provided for users wishing to use custom policies. The steps needed to enable SE Linux in PowerVM Lx86 for each RHEL version are provided below. Details of how custom policies are supported are provided in the next section.

Note that any changes to the SE Linux configuration, for example enabling or disabling SE Linux on the POWER machine or making any changes to the SE Linux policy or the SE Linux setting in the PowerVM Lx86 config, should be made while no x86 processes are running. To ensure that no x86 processes are running, close all running x86 applications, then execute:

```
% /etc/init.d/powervm-lx86 stop
```

After stopping all translated processes and the powervm-lx86-daemon, the following steps are required to enable PowerVM Lx86's SE Linux support. All of these steps should be run as the root user, from a native POWER shell.

Enabling SE Linux in PowerVM Lx86 for RHEL 4

1. Enable SE Linux on the POWER machine. If SE Linux is not enabled in the POWER kernel, PowerVM Lx86 cannot make use of its features. If SE Linux is not already enabled, please refer to your Red Hat Linux documentation for details on how to do this.
2. Install the appropriate binary policy. The supported binary policies are provided in the SE Linux subdirectory of the PowerVM Lx86 install (by default, this will be `/opt/powervm-lx86/selinux`). For each supported RHEL 4 revision there is a directory containing the files specific to this version; for example the RHEL 4 update 6 files are provided in `/opt/powervm-lx86/selinux/rhel4u6`. Select the appropriate directory for the RHEL 4 version of your POWER machine, and copy the policy.18 file into the `/etc/selinux/targeted/policy` directory, for example:
   ```
   % cp /opt/powervm-lx86/selinux/rhel4u4/policy.18
            /etc/selinux/targeted/policy/policy.18
   ```
3. Install the appropriate file contexts. Accompanying each binary policy is a set of file contexts listing how the filesystem should be labelled. For each RHEL 4 revision, there is a file_contexts file in the same directory as the policy. This must be placed in the `/etc/selinux/targeted/contexts/files/` directory, for example:
   ```
   % cp /opt/powervm-lx86/selinux/rhel4u4/file_contexts
            /etc/selinux/targeted/contexts/files/file_contexts
   ```
4. Load the new policy. Having installed the policy files, the new policy can be enabled by executing:
   ```
   % /usr/sbin/load_policy /etc/selinux/targeted/policy/policy.18
   ```
   Alternatively, rebooting the POWER system will cause this policy to be loaded.
5. Ensure that PowerVM Lx86 files are correctly labelled. To reset the labels for PowerVM Lx86's files and directories based on the new policy, either invoke a full filesystem relabel (as described in your Red Hat Linux documentation), or execute restorecon on the PowerVM Lx86 package's files:

```
% rpm -ql powervm-lx86 | restorecon -vR -f -
```

Note the dash (-) at the end of the command line options for restorecon.

6. Enable PowerVM Lx86 domain transitions in the PowerVM Lx86 config file. To do this, add an entry to /etc/opt/powervm-lx86/config as follows:

   ```
   ENABLE_SELINUX_TRANSITIONS=y
   ```

7. Finally, start the PowerVM Lx86 daemon using the init script:

   ```
   % /etc/init.d/powervm-lx86 start
   ```

All PowerVM Lx86 processes, as well as the PowerVM Lx86 daemon, will now run in confined domains. Note that these domains are not the same as those available to POWER processes. For example, a process that would normally run in the unconfined_t domain will now run in lx86_unconfined_t. This difference is not visible to translated x86 processes, but may be observed from the host POWER system. No further action should be required to make use of the SE Linux features. Executing **runx86** to start a translated shell will now automatically move into an equivalent lx86 domain.

Enabling SE Linux in PowerVM Lx86 for RHEL 5

1. Enable SE Linux on the POWER machine. If SE Linux is not enabled in the POWER kernel, PowerVM Lx86 cannot make use of its features. If SE Linux is not already enabled, please refer to your Red Hat Linux documentation for details on how to do this.
2. Load the policy module for the core PowerVM Lx86 types. This module is provided in the SE Linux subdirectory of the PowerVM Lx86 install (by default, this will be `/opt/powervm-lx86/selinux`). This module is compatible with all supported releases of RHEL 5. To load the module, enter:

   ```
   % semodule -i /opt/powervm-lx86/selinux/lx86.pp
   ```

3. Ensure that PowerVM Lx86 files are correctly labelled. To reset the labels for PowerVM Lx86's files and directories based on the new policy, either invoke a full filesystem relabel (as described in your Red Hat Linux documentation), or execute restorecon on the PowerVM Lx86 package's files:

   ```
   % rpm -ql powervm-lx86 | restorecon -vR -f -
   ```

   Note the dash (-) at the end of the command line options for restorecon.
4. Load the policy module for the translated SE Linux domains. A policy is provided for each supported release of RHEL 5, and is located in the corresponding subdirectory of the PowerVM Lx86 installation. For example, for RHEL 5 update 3, the policy module will be located in `/opt/powervm-lx86/selinux/rhel5u3` by default. To load this module, enter the following:

   ```
   % semodule -i /opt/powervm-lx86/selinux/rhel5u3/lx86_x86.pp
   ```

5. Enable PowerVM Lx86 domain transitions in the PowerVM Lx86 config file. To do this, add an entry to /etc/opt/powervm-lx86/config as follows:

   ```
   ENABLE_SELINUX_TRANSITIONS=y
   ```

6. Finally, start the PowerVM Lx86 daemon using the init script:

   ```
   % /etc/init.d/powervm-lx86 start
   ```

All PowerVM Lx86 processes, as well as the PowerVM Lx86 daemon, will now run in confined domains. Note that these domains are not the same as those available to POWER processes. For example, a process that would normally run in the unconfined_t domain will now run in lx86_unconfined_t. This difference is not visible to translated x86 processes, but may be observed from the host POWER system. No further action should be required to make use of the SE Linux features. Executing **runx86** to start a translated shell will now automatically move into an equivalent lx86 domain.

Building custom policies for PowerVM Lx86

If PowerVM Lx86 SE Linux support is required on a Red Hat system using a custom SE Linux policy, or if modifications to the PowerVM Lx86 policies are needed, some additional steps may be required. Note that this section assumes familiarity with the process of compiling and loading SE Linux policies for your particular Red Hat version.

Customizing or adding PowerVM Lx86 domains

This section describes how to install PowerVM Lx86 policy extensions into an environment where the policy in use is different to the default targeted policy provided by Red Hat. For details on configuring or adding PowerVM Lx86 domains, please see the next section.

For RHEL 5, this configuration is already catered for by the modular policy. The PowerVM Lx86 policies are provided in two modules, lx86.pp and lx86_x86.pp, which can be loaded into custom policies, or loaded along side other modules as required.

For RHEL 4, only a single monolithic policy is supported. To allow PowerVM Lx86 policy extensions to operate correctly in a custom policy, it will be necessary to compile a new policy including both the local customizations and the PowerVM Lx86 extensions. In order to do this, four files from the PowerVM Lx86 rpm must be included in the policy sources: lx86.te, lx86.fc, lx86_x86.te and lx86_x86.fc. Note that the different lx86_x86 files are provided for each minor release of RHEL 4. Copy the appropriate .te and .fc files into the appropriate subdirectories of the SE Linux policy source tree, along with any local customizations required, and recompile the policy.

For example, on a RHEL 4 update 6 system, it will be necessary to copy `/opt/powervm-lx86/selinux/lx86.te` and `/opt/powervm-lx86/selinux/rhel4u6/lx86_x86.te` into the `domains/program` subdirectory of the SE Linux policy source. `/opt/powervm-lx86/selinux/lx86.fc` and `/opt/powervm-lx86/selinux/rhel4u6/lx86_x86.fc` should be copied to the `file_contexts/program` subdirectory. For further details on this process, please refer to the Red Hat documentation.

If customization of an PowerVM Lx86 domain is needed, or if a new PowerVM Lx86 domain is to be added, the PowerVM Lx86 policy extensions will need to be recompiled. For RHEL 4, this can be done simply by modifying the lx86_x86.te file and recompiling the monolithic policy as described in the preceding section.

For RHEL 5, the lx86_x86 policy module must be recompiled, then reinstalled into the policy. To do this, in the directory containing the policy module for the appropriate RHEL 5 version, for example, `/opt/powervm-lx86/selinux/rhel5u1`, execute the following commands:

```
% make -f /usr/share/selinux/devel/Makefile lx86_x86.pp
% semodule -i lx86_x86.pp
```

Note that the selinux-policy-devel package must be installed to recompile the policy.

The remainder of this section briefly discusses the details of PowerVM Lx86 policy modification.

As described above, each domain which is used by a translated PowerVM Lx86 process must provide a number of additional operations to permit PowerVM Lx86 to function correctly. These extra rules are defined in the PowerVM Lx86 policy extensions with respect to the lx86_domain attribute. Each PowerVM Lx86 domain has this attribute, and thus is permitted the additional access required to run under translation. Any new domains which are added to the policy for use in the x86 environment must also declare this attribute, or they may fail to operate correctly.

Having defined the new domain, created the required policy rules for it and added the lx86_domain attribute, the entry points to the domain must be considered. If the domain is to be entered by a transition caused by executing files with particular labels, as in the case of the standard targeted daemons (for example, syslogd_t entered by executing syslogd_exec_t files), then two such exec_types must be created. Firstly, the type of the executables must be created (syslogd_exec_t in the above example). Secondly, an additional exec type is needed by PowerVM Lx86 to make the transition; this must be named the same as the first exec_type, but with an lx86_ prefix (for example, lx86_syslogd_exec_t). This type is required by the PowerVM Lx86 translator to make the transition between domains when the binary is executed.

Finally, to complete the addition of the new domain, an entry must be added to `/etc/opt/powervm-lx86/exec_types`. This is a simple list of the supported domain entry points in the current policy. The entry added here should be the type without the lx86_ prefix.

The new policies may now be loaded onto the system. As described above, it is important to ensure that all PowerVM Lx86 processes and the PowerVM Lx86 daemon are shut down before making modifications to the underlying policy.

Chapter 8. Managing remote and local users with PowerVM Lx86

This section provides a reference to managing and configuring the x86 World to support local and remote users with PowerVM Lx86.

Managing remote users with PowerVM Lx86

To enable remote users to authenticate in the x86 environment, for example using NIS, LDAP, or Kerberos, you must set up the x86 World to match the remote user settings of the host POWER system.

With the creation of the x86 World there are now two sets of system configuration files, one for the new x86 World and one for the host POWER system. In particular, there are now two sets of remote user configuration files on the system. To enable remote users to authenticate in the x86 environment, for example using NIS, LDAP or Kerberos, you must enable remote user support in the x86 World. Set up the remote user configuration files in *X86WORLD_ROOT* to match the settings on the host POWER system. The exact changes to the x86 World files will depend on the specific configurations used in your network environment.

As with any normal system, refer to your system administrator for advice and best practices before changing the Linux configuration files in the x86 World.

Managing local users, groups, and passwords with PowerVM Lx86

With the creation of the x86 World there are now two definitions of users, groups, and passwords on the system.

Conflicting definitions of users, groups, and passwords

The x86 World that is installed with PowerVM Lx86 comes with its own set of passwords, groups, and shadow files that are normally found on a native POWER system in the locations `/etc/passwd`, `/etc/group` and `/etc/shadow`, respectively. Therefore, with the introduction of the x86 World there are now two definitions of users, groups and passwords on the system.

This can be confusing to the system administrator and to end users, and there is also a potential security risk. Supposing the x86 World is installed in the location `/i386`, consider the two possible scenarios outlined below:

Scenario 1: Consider two users named fred and bob, both of whom share the same user ID. User fred exists in the native POWER `/etc/passwd` file, and user bob exists in the x86 World `/i386/etc/passwd` file. Suppose you change to user fred in a native POWER shell and then run PowerVM Lx86. In the x86 World you are now user bob, since both fred and bob share the same user ID. Not only is this confusing (try running the **id** command, and you can see that your username has changed from fred to bob), but it can pose security issues, because users fred and bob might have different primary groups.

Scenario 2: Now consider that a user named fred exists in both `/etc/passwd` and `/i386/etc/passwd`, but they have different user IDs. In a native POWER shell you log in as user fred and create a file in `/home/fred` that is only readable by fred. Now suppose you run PowerVM Lx86 as a normal user, and then change to user fred and try to read that file. You won't be able to read it because you have different user IDs.

These two scenarios also apply in a similar way to groups. In its default installation, PowerVM Lx86 attempts to manage these problems transparently by presenting the user with a consistent *one system*, or unified, view of users and groups.

The solution: A unified view

PowerVM Lx86 attempts to unify the definitions of users, groups, and passwords by collecting information from both the native POWER system and the x86 World and producing a merged view of the `/etc/passwd`, `/etc/group` and `/etc/shadow` files (and on Red Hat there is also `/etc/gshadow`).

Again, assuming the x86 World is installed in the `/i386` directory, the solution for `/etc/passwd` follows:

- Whenever a translated x86 program tries to open `/i386/etc/passwd` (the passwd file in the x86 World), PowerVM Lx86 instead opens both the native POWER file `/etc/passwd` and the x86 World file `/i386/etc/passwd` at the same time and attempts to merge all the entries together to provide a single view in such a way that all conflicts and inconsistencies are resolved.

  For example, if PowerVM Lx86 finds that user bob exists in both `/i386/etc/passwd` and `/etc/passwd` files, then PowerVM Lx86 favors the bob entry in `/i386/etc/passwd`, and ignores the bob entry in `/etc/passwd`. This solves the problem described in Scenario 2 above.
- Or if PowerVM Lx86 finds users in both `/i386/etc/passwd` and `/etc/passwd` sharing the same user ID PowerVM Lx86 will always favor the native POWER entry and ignore the x86 version. This solves the problem described in Scenario 1 above.

Operations on `/i386/etc/group` are treated in a similar manner, which solves both of the issues mentioned previously.

However, when conflicting users are found in the shadow files on both the native POWER system and the x86 World, PowerVM Lx86 favors the POWER system entry by default. It is possible to configure PowerVM Lx86 to always favor the x86 World shadow entries by setting the HAVE_SEPARATE_PASSWORDS=y configuration variable in the standard `/etc/opt/powervm-lx86/config` configuration file.

By default, PowerVM Lx86 operates in this one system mode; however, it is possible to revert PowerVM Lx86 back to a two-system mode by setting the MERGE_PASSWD_FILES=n configuration variable in the standard PowerVM Lx86 configuration file. This means that the `/etc/passwd`, `/etc/group`, and `/etc/shadow` (and on Red Hat there is also `/etc/gshadow`) files are handled completely separately by PowerVM Lx86 and the native POWER system. This is not recommended unless it is explicitly known that there are no security risks involved in doing so.

For further details about setting configuration variables for PowerVM Lx86, see “PowerVM Lx86 configuration settings” on page 36.

For further details about the escapes and virtual files for local user management, see “Default escaped directories, files, and sockets” on page 44.

Periodic checks of the subject x86 World

As part of the PowerVM Lx86 installation, a cron job is installed in `/etc/cron.d/powervm-lx86` that invokes the powervm-lx86-world-sync script (by default located in `/usr/sbin`). This periodically checks the x86 World to see if the password, group, or shadow files have become inconsistent (that is, there is a difference between the corresponding x86 World and native POWER file).

From a security perspective, PowerVM Lx86 checks for aliased user IDs (different user names with the same user ID) and new users present in the x86 World but not in the native POWER system. It is the responsibility of the system administrator to adjust the periodicity and timing of the cron job.

By default, the cron job is installed to check the environments every twenty minutes. If a problem is found, then a message is logged to /var/log/messages, and an e-mail is sent to the root user. This e-mail contains clear guidance about how best to resolve the inconsistency by using standard Linux utilities available on the native POWER system.

If the system administrator does not care that certain users or groups are present in the x86 World but not in the native POWER system, then it is possible to configure the cron job to not report about such users or groups. This can be achieved by creating a *white list* of users and groups in the `/etc/opt/powervm-lx86/user_ignore` and `/etc/opt/powervm-lx86/group_ignore` files, respectively. For example, if the system administrator knew about users fred, jane, and bob, but did not want to be warned about them, the administrator could generate the `/etc/opt/powervm-lx86/user_ignore` file with the following contents:

```
fred
bob
jane
```

Each user needs to be on a new line. The same applies for groups.

It is possible to disable this cron job by editing the PowerVM Lx86 configuration file to set the WORLD_CHECK_OR_SYNC=none variable. However, this causes the system administrator to have no visibility of issues as they arise. For further details of the options for the WORLD_CHECK_OR_SYNC configuration switch, see "WORLD_CHECK_OR_SYNC options" on page 56.

Although the cron job does not modify any of the native POWER or x86 World password, group, or shadow files, there are cases when PowerVM Lx86 will update the actual x86 World files on disk. In these circumstances, the x86 World files will be synchronized physically on disk by PowerVM Lx86 with the equivalent merged view. Three cases where this could occur are:

- A user manually adds, deletes or modifies a user or group.
- A user manually changes a user or group password.
- Users or groups are added automatically as part of an application installation such as the WebSphere® software or DB2® software.

PowerVM Lx86 will always present a consistent merged view of the password, shadow, or group files even though this virtual file is not synchronized physically on disk all the time. Any updates to the physical files will be reflected in the merged view. In any of these cases, the cron job will detect the presence of new users or groups and warn the system administrator accordingly.

Known issues with the unified view approach

Find some known issues with this unified view approach that the system administrator should understand.

- It is possible to enable NIS for the x86 World. If NIS is enabled in the x86 World, NIS users will be visible when running PowerVM Lx86. If NIS support is not enabled in the x86 World, PowerVM Lx86 will ignore any NIS entries from the native POWER system.
- PowerVM Lx86 always favors the native password entries. In the case of a user ID clash there is a danger that the user's home directory will not be visible within the subject world. That is, consider these entries: /etc/passwd: 'fred:x:30003:12113::/fred:/bin/bash' /i386/etc/passwd: 'bob:x:30003:12113::/bob:/bin/bash' Since PowerVM Lx86 favors user fred from the native POWER system, it is possible that the /i386/fred directory does not actually exist within the x86 World. However, the cron job should detect any such issues, and provide the system administrator with a means to resolve them.
- It is possible for entries to appear and disappear in a translated x86 shell. For example, consider the following sequence of events:

1. A system administrator adds user fred in the x86 World and then logs in as user fred. The administrator then runs the id utility, which shows output such as: 'uid=30001(fred) gid=500(some company) groups=17(audio),500(some company)'
2. The administrator then adds user bob in a native POWER shell, which happens to be allocated the same user ID (30001). The system administrator then runs a translated x86 shell and again runs the id utility, now producing the following output: 'uid=30001(bob) gid=500(some company) groups=18(uucp),500(some company)'
3. If the system administrator now returns to a native POWER shell, deletes user bob, then runs the 'id' utility again in a translated x86 shell, the output will be as before: 'uid=30001(fred) gid=500(some company) groups=17(audio),500(some company)' As above, the cron job will periodically check the x86 World for any such inconsistencies and warn the system administrator

WORLD_CHECK_OR_SYNC options

The WORLD_CHECK_OR_SYNC configuration switch has multiple options which effects which of the system files are checked by PowerVM Lx86.

The options are shown in the following chart:

Configuration switch value	Effect
check_all	The default value. The cron will check the passwd and group files for any issues.
check_passwd	The cron will only check the passwd files for any issues. The group files will be not be checked for issues with group ids.
check_group	The cron will only check the group files for any issues. The passwd files will not be checked for issues with user ids.
none	This will disable the cron job without removing the cron files. No checking of the passwd or group files or updates to the /etc/mtab files will occur.
sync_all	Performs the same checks as the check_all option, but in addition the cron will keep the /etc/mtab file in the x86 World up to date with entries found in POWER /proc/mounts whenever the cron runs.
force_sync_mtab	The cron will only keep the /etc/mtab file in the x86 World up to date with entries found in POWER /proc/mounts whenever the cron runs. No checking of the passwd or group files will occur.

It is possible to invoke the powervm-lx86-world-sync script directly with the force_sync_mtab option. This forces the x86 World mtab to be updated in sync with the POWER version of the file. This is only expected to be used in a situation where the x86 World mtab file has become corrupt or out of date.

Invoke the script as root with the command:

```
% /usr/sbin/powervm-lx86-world-sync force_sync_mtab
```

Note: The PowerVM Lx86 daemon must be running for this operation to succeed.

Root user and root passwords

The root user on the system is also a local user and is handled by the user ID mechanism in a special way.

The privileges of a translated root user are the same as a root user on the host POWER system. In a translated x86 shell, if the user is running as root, then the user has the same privileges as root in the POWER environment. This is an expected behavior of the x86 environment.

If an escape is created from the x86 World to the POWER filesystem (which can only be carried out by the POWER root user), then a root user in the x86 environment can access the POWER file system as the root user.

The user ID support within PowerVM Lx86 defaults to using the POWER password for the root user in the x86 World. When prompted for the root password while running in the x86 environment, use the POWER password. However, if PowerVM Lx86 is explicitly setup to use separate passwords (by setting the HAVE_SEPARATE_PASSWORDS=y configuration variable in the standard `/etc/opt/powervm-lx86/config` configuration file) for the x86 World and POWER system, when you are prompted for the root password, use the x86 World password.

For further details about setting configuration variables for PowerVM Lx86, see "PowerVM Lx86 configuration settings" on page 36.

Chapter 9. PowerVM Lx86 error messages and resolutions

This section describes the error messages that can be reported on the terminal by the components of PowerVM Lx86 and details of how to resolve each problem.

If the resolution section for each error does not help you to resolve the problem, report the failure by contacting IBM Support.

There are several components of PowerVM Lx86 that can produce error messages:
- The PowerVM Lx86 translator itself (powervm-lx86)
- The PowerVM Lx86 translator daemon (powervm-lx86-daemon)
- The x86 /etc/init.d scripts
- The local user ID support (powervm-lx86-world-sync)
- The PowerVM Lx86 installer

Error message template

The PowerVM Lx86 errors messages use the following template:

```
[Module][Error: xxxx]<error text>
```

Where [Module] is powervm-lx86, powervm-lx86-daemon, or similar, xxxx in [Error: xxxx] is a unique error number for that module starting from 0001, and <error text> is plain text describing the error and its possible resolution.

PowerVM Lx86 (powervm-lx86) errors

This section lists error messages for PowerVM Lx86 (powervm-lx86), including cause and resolution.

Error message	**[powervm-lx86][Error: 0001] Process received signal <signalname> (<signalnumber>).**
Cause	One of the running x86 applications received a signal unexpectedly.
Resolution	Normally an x86 application will produce an error, report or log file when this problem occurs. Contact IBM support for further help.

Error message	**[powervm-lx86][Error: 0002] Access denied for x86 binary '<binaryname>'. Please check the permissions on the file.**
Cause	You do not have permissions to access the binary file.
Resolution	Check the permissions on the binary you attempted to execute and try again.

Error message	**[powervm-lx86][Error: 0003] File '<filename>' is not a valid x86 binary. The file may be a POWER binary. Please check the type of the file.**
Cause	The binary may not be a valid x86 binary.
Resolution	Check that the binary is a valid x86 binary, for example by running the command line tool 'file'. PowerVM Lx86 can only execute Linux/x86 ELF binary files.

Error message	**[powervm-lx86][Error: 0004] Cannot read x86 binary '<filename>'. Please check the permissions on the file.**
Cause	You do not have permissions to read the binary file. This case should be handled correctly when running within a translated x86 shell.
Resolution	Ensure that you are running within a translated x86 shell and attempt to execute the binary again.

Error message	**[powervm-lx86][Error: 0005] Current working directory is not visible from the x86 World root. Please "cd "<path>" and try again.**
Cause	The current working directory must be one of the following: • The *X86WORLD_ROOT* directory or any of its subdirectories, for example, `/i386` or `/i386/etc`. • An escape directory or any of its subdirectories, for example, `/home/mike` or `/home/mike/myDirectory`.
Resolution	Ensure that the current working directory is visible to the Linux/x86 environment, by ensuring it meets the criteria in the Cause section above. Ensure that you are invoking the **runx86** script correctly and check the default list of escapes for the x86 World and any escapes you have added to the x86 World since installation.

Error message	**[powervm-lx86][Error: 0006] x86 binary '<binaryname>' is not a valid binary. It may be a data file. Please check that the file is an executable binary.**
Cause	The binary may not be a valid Linux/x86 ELF binary.
Resolution	Check that the binary is a valid Linux/x86 ELF binary, for example by running the command line tool 'file'. PowerVM Lx86 can only execute valid Linux/x86 ELF binary files.

Error message	**[powervm-lx86][Error: 0007] x86 binary '<binaryname>' is not a valid file. It may be a directory. Please check that the file is an executable binary.**
Cause	You may have attempted to execute a directory instead of a binary file within that directory, for example, `/home/user` instead of `/home/user/myDirectory/myBinary`.
Resolution	Ensure that you have correctly typed the name of the binary file you want to execute.

Error message	**[powervm-lx86][Error: 0008] Cannot access x86 binary '<binaryname>'. Please check that the file exists, the path to the file is valid and that the permissions on the path are correct.**
Cause	You may not have permissions to access the binary file or the file may not exist or the path to the file may not be valid.
Resolution	Check that the file exists, the path to the file is valid and that the permissions on the path are correct and try again.

Error message	**[powervm-lx86][Error: 0009] Too many symbolic link loops encountered for x86 binary '<binaryname>'. Please check for loops in any symbolic links in the path to the binary and try again.**
Cause	When attempting to resolve the path to the file, too many (more than 20) symbolic links were encountered. This may be caused by a symbolic link loop, for example, a symbolic link pointing back to itself.

Error message	**[powervm-lx86][Error: 0009] Too many symbolic link loops encountered for x86 binary '<binaryname>'. Please check for loops in any symbolic links in the path to the binary and try again.**
Resolution	Ensure that a symbolic link loop has not been created for the file you are trying to access.

Error message	**[powervm-lx86][Error: 0010] Unknown problem with file '<filename>'. Please save the log file '<logfile>' and contact IBM support.**
Cause	Unknown.
Resolution	Contact IBM support with details of how the error occurred and send any log files that were generated.

Error message	**[powervm-lx86][Error: 0011] Could not contact the powervm-lx86-daemon (error ('<errorname>', '<errornumber>')). Please check that the powervm-lx86-daemon is running and try again.**
Cause	PowerVM Lx86 has failed to connect to the PowerVM Lx86 daemon (powervm-lx86-daemon).
Resolution	Check that the powervm-lx86-daemon is running by using the command: `/etc/init.d/powervm-lx86 status`. If the powervm-lx86-daemon is not running, then start the powervm-lx86-daemon by using the command: `/etc/init.d/powervm-lx86 start`. If the powervm-lx86-daemon is running, then try launching the x86 application again. If PowerVM Lx86 still cannot contact the powervm-lx86-daemon, then contact IBM support.

Error message	**[powervm-lx86][Error: 0012] powervm-lx86 cannot write to the specified log file '<logfile>' (error (<errorname>, <errornumber>)**
Cause	An error has occurred and PowerVM Lx86 has attempted but failed to write to the log file.
Resolution	• Check that the directory where the log file is being written to exists, if not, create a directory with the same name and try again. • Check that permissions have been granted to allow access to the directory.

Error message	**[powervm-lx86][Error: 0013] powervm-lx86 has terminated unexpectedly. Please save the log file '<filename>' and contact IBM support.**
Cause	An error has occurred in PowerVM Lx86 or the x86 application, causing the translated process to be terminated.
Resolution	Save the log files and contact IBM support.

Error message	**[powervm-lx86][Error: 0014] Please invoke PowerVM Lx86 using the runx86 script.**
Cause	The powervm-lx86 binary has been invoked directly (for example, `/opt/powervm-lx86/bin/powervm-lx86`).
Resolution	To invoke PowerVM Lx86, use the **runx86** script, for example, `/usr/local/bin/runx86`.

Error message	**ERROR: ld.so: object '/usr/local/bin/libmemcpy.so' from LD_PRELOAD cannot be preloaded. Ignored.**
Cause	The powervm-lx86-tools RPM has not been installed in the VxE when running on a POWER7 system.

Error message	**ERROR: ld.so: object '/usr/local/bin/libmemcpy.so' from LD_PRELOAD cannot be preloaded. Ignored.**
Resolution	Installation of the powervm-lx86-tools is handled automatically by the installer.pl. Copy the RPM, found at `/opt/powervm-lx86/extras/`, into the x86 World and install it from an x86 shell. If you do not wish to use the supplied memcpy.so preload you can use the LD_PRELOAD_OVERRIDE config switch.

PowerVM Lx86 log file errors

This section lists the log file error messages, including cause and resolution.

Error message	**[powervm-lx86][Error: 0015] Cannot write to specified log directory '<logdirectory>'. Please check the permissions on the directory.**
Cause	PowerVM Lx86 is attempting to write to the log file directory but failed.
Resolution	Check that the directory has write permissions.

Error message	**[powervm-lx86][Error: 0016] Cannot open logfile. Unexpected error occurred while initializing '<logfile>'. Please contact IBM support.**
Cause	Unknown.
Resolution	Contact IBM support.

Error message	**[powervm-lx86][Error: 0017] Failed to create unique logfile name.**
Cause	PowerVM Lx86 attempts to create a uniquely numbered log file name each time it generates a new log file, `powervm-lx86.log.<binaryname>.<processID>.<uniquenumber>` (for example, powervm-lx86.log.perl.23724.4) PowerVM Lx86 has been unable to create a file with a new unique number.
Resolution	Check the log directory to see if a particular process has generated an unexpected number of log files. Do not delete the log files unless you are certain they are not required to help resolve the issue further. If the problem persists, contact IBM support.

Error message	**[powervm-lx86][Error: 0018] Cannot open logfile - filesystem is full.**
Cause	The file system where PowerVM Lx86 stores the log files appears to be full.
Resolution	Check for free space on the file system and make more space available if none is free.

Error message	**[powervm-lx86][Error: 0019] Cannot open logfile - filesystem is not writable. Please check the permission on the file.**
Cause	The log file is not writable because the whole file system is read-only.
Resolution	Either change the log file directory location in the config file or remount the file system where the log file is located with read-write permissions.

PowerVM Lx86 daemon (powervm-lx86-daemon) errors

This section describes PowerVM Lx86 daemon (powervm-lx86-daemon) error messages, including cause and resolution.

Error message	**[powervm-lx86-daemon] powervm-lx86-daemon not starting due to errors. Please correct the problem and try again.**
Cause	An error has occurred while invoking the PowerVM Lx86 daemon.
Resolution	A more detailed error message will be printed with this message. Follow the instructions in that message.

Error message	**[powervm-lx86-daemon][Error: 0001] Usage: /etc/init.d/powervm-lx86 [start|stop|force-reload|restart|status]**
Cause	• The `/etc/init.d/powervm-lx86` script has been invoked with an invalid argument not listed above. • The powervm-lx86-daemon binary has been invoked directly on the command line with an argument (it can only be invoked directly if no arguments are provided and this will invoke the daemon with the start option).
Resolution	Invoke the powervm-lx86-daemon using the `/etc/init.d/powervm-lx86` script with one of the arguments listed above. If you want to invoke powervm-lx86-daemon directly, then do not provide any arguments to the binary on the command line.

Error message	**[powervm-lx86-daemon][Error: 0002] This machine model is not supported. Please check the system requirements in the PowerVM Lx86 Administration Guide.**
Cause	The PowerVM Lx86 daemon failed to start as it appears that the system is not an IBM System p server.
Resolution	Ensure that you are running PowerVM Lx86 on a supported platform by checking the system requirements in the PowerVM Lx86 Administration Guide.

Error message	**[powervm-lx86-daemon][Error: 0003] Failed to open lock file '<lockfile>' (error ('<errorname>', <errornumber>)). Please check the permissions on the directory and lock file, then try again.**
Cause	The PowerVM Lx86 daemon failed to start because it cannot open the lock file.
Resolution	Check the permissions on the directory and lock file. The lock file is located in `/var/opt/powervm-lx86/daemon/powervm-lx86-daemon.lock`.

Error message	**[powervm-lx86-daemon][Error: 0004] powervm-lx86-daemon is already running (lock file detected). You do not need to invoke powervm-lx86-daemon again.**
Cause	The PowerVM Lx86 daemon is already running and you have attempted to start another instance.
Resolution	Check that the powervm-lx86-daemon is running with the following command: `/etc/init.d/powervm-lx86 status`. If the powervm-lx86-daemon is running, then continue to launch x86 applications as normal. If not, then start the powervm-lx86-daemon with the following command: `/etc/init.d/powervm-lx86 start`.

Error message	**[powervm-lx86-daemon][Error: 0005] Failed to open log file '<logfile>' (error ('errorname', <errornumber>)). Please check the permissions on the directory and log file, then try again.**
Cause	The PowerVM Lx86 daemon could not open the log file.
Resolution	Check that the directory where the log file is being written to exists, if not, create a directory with that name and try again. Check the permissions on the directory (which should be writable by daemon) and the log file (which should be owned by daemon and readable and writable by daemon) and try again.

Error message	**[powervm-lx86-daemon][Error: 0006] Failed to make directory '<directoryname>' (error ('<errorname>', <errornumber>)). Please check the permissions on the directory and try again.**
Cause	The PowerVM Lx86 daemon could not create a directory on the system.
Resolution	Check the permissions on the parent directory (which should be writable daemon) where the directory is being created and try again.

Error message	**[powervm-lx86-daemon][Error: 0007] Socket directory '<socketdirectory>' must be owned by user '<user1>' (id <userid1>) but is owned by user '<user2>' (id <userid2>). Please correct the ownership on the directory and try again.**
Cause	The ownership of the PowerVM Lx86 daemon socket is not correct.
Resolution	Change the ownership of the socket directory from **<user2>** to **<user1>** and try again.

Error message	**[powervm-lx86-daemon][Error: 0008] powervm-lx86-daemon cannot change user or group. Please invoke the powervm-lx86-daemon as root.**
Cause	The PowerVM Lx86 daemon was not invoked by root.
Resolution	Invoke PowerVM Lx86 daemon as root. First become root, then invoke the daemon with the following command: `/etc/init.d/powervm-lx86 start`.

Errors and warnings from the runx86 script

The runx86 script may produce these errors when being used to run x86 applications.

Warnings from the runx86 script

Warning message	**[runx86][Warning: 0001] The current working directory is not visible from the x86 World. The x86 shell will be invoked with the x86 World root '<X86WORLD_ROOT>' as the current working directory.**
Cause	As the current working directory is not visible from the x86 World, the runx86 command warns that it changed the current working directory to the x86 World root.

Warning message	**[runx86][Warning: 0002] Cannot re-open pseudo terminal <file> -> <pseudo terminal path>: open: <error>.**
Cause	runx86 was unable to re-open a pseudo terminal with the new SELinux security context, so if the security context is restrictive then terminal interaction may not work.

Warning message	**[runx86][Warning: 0003] Cannot re-open pseudo terminal <file> -> <pseudo terminal path>: dup2: <error>.**
Cause	runx86 was unable to re-open a pseudo terminal with the new SELinux security context, so if the security context is restrictive then terminal interaction may not work.

Errors from the runx86 script

Error message	**[runx86][Error: 0001] Cannot execute '<x86 binary>'. The current working directory is not accessible from your x86 World '<x86 World>'. To execute '<x86 binary>' either change directories so that you are in a directory visible from the x86 World (e.g. 'cd /i386') or use the link86(8) command to make the current directory visible to the x86 World (consult the Administration Guide).**
Cause	runx86 has been invoked with an x86 program as an argument, but the current working directory is not accessible from the x86 World. runx86 will not change the current working directory for you.
Resolution	You must change the directory manually to the x86 World or make the directory visible from the x86 World as directed.

Error message	**[runx86][Error: 0002] Cannot chdir into '<X86WORLD_ROOT>'.**
Cause	runx86 cannot change the directory to the x86 World root.
Resolution	Check the directory exists and the permissions on the directory are correct then try again.

Error message	**[runx86][Error: 0003] The required PowerVM Lx86 security context '<security context>' was not valid. Please ensure the PowerVM Lx86 security policy is loaded (consult the Administration Guide).**
Cause	The PowerVM Lx86 security context is not valid.
Resolution	If you did not intend to use SELinux, remove 'ENABLE_SELINUX_TRANSITIONS=y' from the PowerVM Lx86 config file and try again. If you did intend to user SELinux, ensure that the security policy is loaded. Consult the security section of the Administration Guide ("SE Linux support with PowerVM Lx86" on page 47) for further details.

Error message	**[runx86][Error: 0004] Cannot execute powervm-lx86: '<error>'.**
Cause	PowerVM Lx86 failed to execute.
Resolution	Ensure that the binary exists, the permissions on the file and directory are correct, and runx86 is executing in the appropriate SELinux security context if appropriate.

Error message	**[runx86][Error: 0005] SELinux transitions are enabled but SELinux does not appear to be installed. Please disable SELinux transitions in the PowerVM Lx86 config file (consult the Administration Guide).**
Cause	PowerVM Lx86 failed to execute because SELinux transitions are enabled but SELinux does not appear to be installed on the POWER system.
Resolutio	If you intend to use SELinux, ensure that it is installed, enabled, and that `/usr/sbin/selinuxenabled` exists and is executable. Otherwise you will need to remove 'ENABLE_SELINUX_TRANSITIONS=y' from the PowerVM Lx86 config file and try again.

Error message	**[runx86][Error: 0006] SELinux transitions are enabled but SELinux is disabled. Please enable SELinux or disable SELinux transitions in the PowerVM Lx86 config file, and then restart the powervm-lx86 daemon (consult the Administration Guide).**
Cause	PowerVM Lx86 failed to execute because SELinux transitions are enabled but SELinux is disabled on the POWER system.
Resolution	If you intended to use SELinux, ensure that it is enabled. Otherwise you will need to remove 'ENABLE_SELINUX_TRANSITIONS=y' from the PowerVM Lx86 configuration file and try again.

Error message	**[runx86][Error: 0007] Cannot open '<config file>' for reading: <error>.**
Cause	The specified PowerVM Lx86 configuration file could not be read.
Resolution	Check that the permissions on the file are correct.

Error message	**[runx86][Error: 0008] Cannot open '/proc/self/attr/current' for reading: <error>.**
Cause	There was a problem finding the current SELinux security context.
Resolution	Ensure that SELinux is installed correctly or remove 'ENABLE_SELINUX_TRANSITIONS=y' from the PowerVM Lx86 config file and try again.

Error message	**[runx86][Error: 0009] Cannot open '/proc/self/attr/exec' for writing: <error>.**
Cause	There was a problem switching to the required SELinux security context.
Resolution	Ensure that SELinux is installed correctly or remove 'ENABLE_SELINUX_TRANSITIONS=y' from the PowerVM Lx86 config file and try again.

Error message	**[runx86][Error: 0010] Cannot switch to the required PowerVM Lx86 security context '<context>': <error>.**
Cause	There was a problem switching to the required SELinux security context.
Resolution	Ensure that SELinux is installed correctly or remove 'ENABLE_SELINUX_TRANSITIONS=y' from the PowerVM Lx86 config file and try again.

Error message	**[runx86][Error: 0011] Cannot close '/proc/self/attr/exec': <error>.**
Cause	There was a problem switching to the required SELinux security context.
Resolution	Ensure that SELinux is installed correctly or remove 'ENABLE_SELINUX_TRANSITIONS=y' from the PowerVM Lx86 config file and try again.

Errors from the linkx86 script

The linkx86 script may produce these errors when being used to create an escape from the x86 World to a directory on the POWER system.

Error message	**[linkx86][Error: 0001] linkx86 must not be run under translation.**
Cause	The linkx86 script has been invoked from within the x86 environment, probably from a translated x86 shell.

Error message	**[linkx86][Error: 0001] linkx86 must not be run under translation.**
Resolution	linkx86 can only be invoked from a POWER shell. Check that you are using a POWER shell and invoke the linkx86 script again.

Error message	**[linkx86][Error: 0002] Superuser privileges are required for this script.**
Cause	The linkx86 script has been invoked by a non-root user.
Resolution	Ensure that you are root and try again.

Error message	**[linkx86][Error: 0003] Path must be absolute.**
Cause	The path provided as an argument to linkx86 is not an absolute path. It is likely that you provided a relative path as an argument.
Resolution	Invoke linkx86 with an absolute path as an argument.

Error message	**[linkx86][Error: 0004] Path must not be the root '/' directory.**
Cause	The path provided as an argument to linkx86 was the root ('/') directory.
Resolution	Invoke linkx86 with an absolute path as an argument that is not the root ('/') directory.

Error message	**[linkx86][Error: 0005] <path> does not exist.**
Cause	The path that you are trying to create an escape directory to does not exist on the POWER system.
Resolution	Check that the path exists on the POWER system. If not, create the directory on the POWER system. Check that you have correctly typed the name of the path and invoke linkx86 again.

Error message	**[linkx86][Error: 0006] Unable to create <path/filename> because it already exists.**
Cause	The escape link cannot be created because something with that name already exists in the x86 World.
Resolution	Ensure that the link you are trying to create does not already exist. Check that you have correctly typed the name of the link and invoke linkx86 again.

Error message	**[linkx86][Error: 0007] Unable to create <path/filename>. Please check you have the necessary permission.**
Cause	The escape link cannot be created because you do not have the correct permissions.
Resolution	Check the permissions on the directory where the file is being created and ensure that users have write permissions.

Errors from the x86 /etc/init.d support scripts

This section describes error messages from the x86 /etc/init.d support scripts, including cause and resolution.

Errors from the /etc/init.d/powervm-lx86 script

Error message	[/etc/init.d/powervm-lx86][Error: 0001] You must be root to run this script.
Cause	The powervm-lx86 script has been launched by a non-root user.
Resolution	The powervm-lx86 script must be run by root. Ensure that you are running as root and run the script again.

Error message	[/etc/init.d/powervm-lx86][Error: 0002] /etc/opt/powervm-lx86/config must be owned by root.
Cause	The PowerVM Lx86 configuration file needs to be owned by root to ensure that it cannot be tampered with by any non-root user. The configuration file is not currently owned by root.
Resolution	Check that the configuration file is owned by root. Contact IBM support if you have any further problems.

Error message	[/etc/init.d/powervm-lx86][Error: 0003] /etc/opt/powervm-lx86/config must be writable only by root.
Cause	The PowerVM Lx86 configuration file must be writable only by root to ensure that it cannot be tampered with by any non-root user. The configuration file is currently writable by non-root users.
Resolution	Check that the configuration file is writable only by root. Contact IBM support if you have any further problems.

Error message	[/etc/init.d/powervm-lx86][Error: 0004] Kernel doesn't contain binfmt_misc, and loading it as a module failed. Please check your kernel configuration and ensure binfmt_misc is available.
Cause	The powervm-lx86 script requires that the binfmt_misc kernel feature is enabled on the POWER system.
Resolution	Contact IBM support for further help.

Error message	[/etc/init.d/powervm-lx86][Error: 0005] Could not mount /proc/sys/fs/binfmt_misc (from binfmt_misc).
Cause	The powervm-lx86 script requires that the binfmt_misc kernel feature is enabled on the POWER system. The script failed to mount the binfmt_misc file in `/proc`.
Resolution	Contact IBM support for further help.

Error message	[/etc/init.d/powervm-lx86][Error: 0006] Failed to register i386 handler with binfmt_misc.
Cause	The powervm-lx86 script failed to register the i386 handler with binfmt_misc on the POWER system.
Resolution	Contact IBM support for further help.

Error message	[/etc/init.d/powervm-lx86][Error: 0007] Failed to register i386so handler with binfmt_misc.
Cause	The powervm-lx86 script failed to register the i386so handler with binfmt_misc on the POWER system.
Resolution	Contact IBM support for further help.

Error message	**[/etc/init.d/powervm-lx86][Error: 0008] Unable to source init-script functions.**
Cause	The powervm-lx86 script failed to launch.
Resolution	Contact IBM support.

Error message	**[/etc/init.d/powervm-lx86][Error: 0009] /var/opt/powervm-lx86/daemon/<file> must be owned by daemon.**
Cause	The PowerVM Lx86 daemon file is currently not owned by daemon.
Resolution	Check that the daemon file is owned by daemon and update if necessary. Contact IBM support if you have any further problems.

Error message	**[/etc/init.d/powervm-lx86][Error: 0010] /var/opt/powervm-lx86/daemon/<file> must be writable by daemon.**
Cause	The PowerVM Lx86 daemon file is currently not writable by daemon.
Resolution	Check that the daemon file is writable by daemon and update if necessary. Contact IBM support if you have any further problems.

Errors from the powervm-lx86-rc runlevel scripts

The powervm-lx86-rc runlevel scripts (`/etc/init.d/powervm-lx86-rc{1 - 6}`) are invoked when the host POWER system runlevel changes.

Error message	**[/etc/init.d/powervm-lx86-rc<number>][Error: 0001] You must be root to run this script.**
Cause	The powervm-lx86-rc script was launched by a non-root user.
Resolution	The powervm-lx86-rc scripts must be run by root. The scripts are not expected to be run manually. Contact IBM support.

Error message	**[/etc/init.d/powervm-lx86-rc<number>][Error: 0002] /etc/opt/powervm-lx86/config must be owned by root.**
Cause	The PowerVM Lx86 configuration file needs to be owned by root to ensure that it cannot be tampered with by any non-root user. The configuration file is not currently owned by root.
Resolution	Check that the configuration file is owned by root. Contact IBM support if you have any further problems.

Error message	**[/etc/init.d/powervm-lx86-rc<number>][Error: 0003] /etc/opt/powervm-lx86/config must be writable only by root.**
Cause	The PowerVM Lx86 configuration file must be writable only by root to ensure that it cannot be tampered with by any non-root user. The configuration file is currently writable by non-root users.
Resolution	Check that the configuration file is writable only by root. Contact IBM support if you have any further problems.

Error message	**[/etc/init.d/powervm-lx86-rc<number>][Error: 0004] Error changing x86 runlevel.**
Cause	The powervm-lx86-rc script failed to change the x86 runlevel.
Resolution	Contact IBM support.

Error message	**[/etc/init.d/powervm-lx86-rc<number>][Error: 0005] Unable to source init-script functions.**
Cause	The powervm-lx86-rc script failed to launch when invoked.
Resolution	Contact IBM support.

Errors from the /etc/init.d/powervm-lx86-rcmonitor script

Error message	**[/etc/init,d/powervm-lx86-rcmonitor][Error: 0001] You must be root to run this script.**
Cause	The powervm-lx86-rcmonitor script has been launched by a non-root user.
Resolution	The powervm-lx86-rcmonitor script must be run by root. Ensure that you are running as root and run the script again.

Error message	**[/etc/init,d/powervm-lx86-rcmonitor][Error: 0002] Unable to source init-script functions.**
Cause	The powervm-lx86-rcmonitor script failed to launch.
Resolution	Contact IBM support.

Errors from the dependency_checker.pl script

Error message	**[dependency_checker.pl][Error: 0001] You must be root to run this script**
Cause	The dependency_checker.pl script has been invoked by a non-root user.
Resolution	The powervm-lx86-rcmonitor script normally invokes this script. The script should not normally be invoked manually. Contact IBM support for further help.

Error message	**[dependency_checker.pl][Error: 0002] Directory <directoryName> does not exist.**
Cause	The dependency_checker.pl script failed to find the main init.d directory.
Resolution	Check if the directory exists. Contact IBM support for further help.

Error message	**[dependency_checker.pl][Error: 0003] Cannot open <file> for reading: <errornumber>.**
Cause	The dependency_checker.pl script failed to open a file in the main init.d directory.
Resolution	Check that the file exists and the permissions on the file. Contact IBM support for further help.

Error message	**[dependency_checker.pl][Error: 0004] Cannot open <file> for writing: <errornumber>.**
Cause	The dependency_checker.pl script has failed to open a file in the main init.d directory.
Resolution	Check that the file exists and the permissions on the file. Contact IBM support for further help.

Errors from the execve router

The execve router is responsible for launching certain processes for PowerVM Lx86. In the unlikely event that it failed to invoke PowerVM Lx86, one of the following errors may be seen:

Error message	**[powervm-lx86 exec router][Error: 0001] powervm-lx86 exec router could not invoke powervm-lx86, (error <errornumber>).**
Cause	The powervm-lx86-world-sync script failed to invoke the PowerVM Lx86 binary (powervm-lx86). The powervm-lx86 binary may not exist or the configuration file (if one exists) may have POWERVM_LX86_LOCATION set to an incorrect location.
Resolution	Check that the powervm-lx86 binary exists in the default installation location or where you chose to install the binary if to a non-default location. If installed to a non-default location, check that POWERVM_LX86_LOCATION configuration switch in /etc/opt/powervm-lx86/config correctly points to the powervm-lx86 binary. If you cannot resolve the problem, contact IBM Support.

Error message	**[powervm-lx86 exec router][Error: 0002] Path to powervm-lx86 binary is too long (<number>).**
Cause	The path to the powervm-lx86 binary is too long, for example `/opt/<many_character_directory_name>/powervm-lx86`.
Resolution	Ensure that the PowerVM Lx86 binary is installed into a directory with a reasonably short length directory path.

Syslog messages

Support for logging system messages is handled in a special manner by PowerVM Lx86 so that system messages in the x86 World and the POWER system are kept separate.

See "Default PowerVM Lx86 escapes and virtual files in the x86 World" on page 42 for detail of how system logs and error messages are handled by PowerVM Lx86.

Alerts and errors when managing local users, groups, and passwords with PowerVM Lx86

This section describes alert and error messages, including cause and resolution.

E-mail alerts

The following alerts are sent via e-mail by the powervm-lx86-world-sync script to root and logged in `/var/log/messages` on the POWER system when conflicts are found with the users, groups and passwords with PowerVM Lx86 and the underlying POWER system.

Alert message	**New user account (<useraccount>) found in <file>**
Cause	A new user account has been added to the file (for example, `/etc/passwd`) in the x86 World.
Resolution	The root user will be e-mailed specific details about how to resolve the issues with this event. See "E-mail messages for managing local users, groups, and passwords in PowerVM Lx86" on page 73.

Alert message	**Aliased user id (<userid>) found in <file>**
Cause	A username has been found in the file (for example, `/etc/passwd`) which has the same user id as an entry in the POWER version of the file.
Resolution	The root user will be e-mailed specific details about how to resolve the issues with this event. See “E-mail messages for managing local users, groups, and passwords in PowerVM Lx86” on page 73.

Alert message	**New group (<group>) found in <file>**
Cause	A new group has been added to the file (for example, `/etc/group`) in the x86 World.
Resolution	The root user will be e-mailed specific details about how to resolve the issues with this event. See “E-mail messages for managing local users, groups, and passwords in PowerVM Lx86” on page 73.

Alert message	**Aliased group id (<groupid>) found in <file>**
Cause	A group name has been found in the file (for example, `/etc/group`) which has the same group id as an entry in the POWER version of the file.
Resolution	The root user will be e-mailed specific details about how to resolve the issues with this event. See “E-mail messages for managing local users, groups, and passwords in PowerVM Lx86” on page 73.

Errors from the powervm-lx86-world-sync script

Error message	**[powervm-lx86-world-sync][Error: 0001] You must be root to run this script.**
Cause	The powervm-lx86-world-sync script has been invoked by a non-root user.
Resolution	The cron job `/etc/cron.d/powervm-lx86` invokes powervm-lx86-world-sync as root. If this error message occurs, contact IBM support.

Error message	**[powervm-lx86-world-sync][Error: 0002] Failed to get current set of mount entries**
Cause	The powervm-lx86-world-sync script has been invoked with the `sync_all` argument, but was unable to access the POWER system `/proc/mounts` file.
Resolution	Contact IBM support.

Error message	**[powervm-lx86-world-sync][Error: 0003] Cannot open <filename>: <errorcode>**
Cause	The powervm-lx86-world-sync script failed to open a user management file, for example `/etc/passwd`.
Resolution	Check the file exists in the x86 World and check that it is owned by root, group root, and the permissions are set to 644 (RW owner, R group, and R others).

Error message	**[powervm-lx86-world-sync][Error: 0004] The powervm-lx86-daemon is not running. Please start the powervm-lx86-daemon.**
Cause	The powervm-lx86-daemon is not running and is required for managing users, groups and passwords with PowerVM Lx86

Error message	[powervm-lx86-world-sync][Error: 0004] The powervm-lx86-daemon is not running. Please start the powervm-lx86-daemon.
Resolution	Invoke PowerVM Lx86 daemon as root. First become root, then invoke the daemon with the following command: /etc/init.d/powervm-lx86 start

Error message	[powervm-lx86-world-sync][Error: 0005] Unrecognised option 'WORLD_CHECK_OR_SYNC=<option>'
Cause	The `WORLD_CHECK_OR_SYNC` configuration file option has been set to an unrecognized value.
Resolution	The valid options are: `sync_all`, `check_all`, `check_passwd`, `check_group`, `force_sync_mtab`, `nonecheck_all` is the default. The `WORLD_CHECK_OR_SYNC` option can also be enabled by default to `check_all` by removing the WORLD_CHECK_OR_SYNC line from the configuration file.

E-mail messages for managing local users, groups, and passwords in PowerVM Lx86

These tables show the templates for the e-mails sent to the root user when a local user ID alert occurs. The text with square brackets ([_*number*]) represent specific files and details to the alert.

Alert message	New user account ('<useraccount>') found in <file>.
Example e-mail	A new user account has been found in the x86 World password file ([_1]) that is not present in the POWER password file. The relevant password entry found in [_2] is shown here: [_3] where the fields represent the user account, password, user id, primary group id, comment, home directory and default shell, respectively. Also, the output of the translated command '/usr/bin/id [_4]' is shown below [_5] where the second field shows the primary group and the third field shows the full list of groups to which user [_6] belongs, including the primary and supplementary groups. In order to stop receiving this email in future you have two options. The first option is to add the user [_7] to the user whitelist file found in `/etc/opt/powervm-lx86/user_ignore`. In this case you don't need to add the user to the POWER side and this user will be ignored in future. The second option is to add the user on the POWER side yourself. One way of doing this (but not necessarily the most complete) is to execute the following command in a native POWER shell: `/usr/sbin/useradd -m -d <home directory> [_8]` To replicate the x86 World setup completely you will have to ensure that all the groups to which user [_9] is a member exist on the POWER side and then set both the primary and supplementary groups for the user [_10] accordingly. (This mail was generated by the cron script [_1] and can be disabled by setting the configuration variable `WORLD_CHECK_OR_SYNC=none` in `/etc/opt/powervm-lx86/config`).

Alert message	**Aliased user id ('<userid>') found in <file>.**

Example e-mail	A user account has been found in the x86 World password file ([_1]) that shares the same user id ([_2]) with an account found in the POWER password file (`/etc/passwd`). The relevant password entry found in the x86 World file ([_3]) is shown here:
	
	[_4] and the relevant password entry found in the POWER file (`/etc/passwd`) is shown here:
	
	[_5]
	
	where the fields represent the user account, password, user id, primary group id, comment, home directory and default shell, respectively.
	
	Although highly unlikely, this could represent a serious security risk, since the identity corresponding to user id [_6] is ambiguous.
	
	In order to stop receiving this email in future you have two options. If you believe there to be no security risk involved then you can add the user id [_7] to the whitelist file found in `/etc/opt/powervm-lx86/uid_ignore`. Otherwise, it is recommended that you rectify the problem by changing the user id of [_8] in the x86 World. However, before doing so you MUST first shut down PowerVM Lx86 by running the command:
	`/etc/init.d/powervm-lx86 stop`
	
	You must do this as there may currently be Lx86 processes running as user [_9]. Next, you need to choose a new, unique user id for user [_10], ensuring that this user id does not already exist in either the x86 World or POWER password files. The simplest way to change the user id of user [_11] is to manually edit the x86 World password file, updating the user id accordingly. Finally, you must update the ownership of all files in the x86 World that are owned by user id [_12]. There are several ways to update the ownership on files in the x86 World, however here is one simple command you can run:
	`/bin/chown --from=[_13] -R <new user> [_14]`
	
	CAUTION:
	Any mistakes made at this stage could damage both your x86 World and your POWER system. You should then be able to restart PowerVM Lx86 with the following command:
	`/etc/init.d/powervm-lx86 start`
	
	(This mail was generated by the cron script [_1] and can be disabled by setting the configuration variable `WORLD_CHECK_OR_SYNC=none` in `/etc/opt/powervm-lx86/config`)

Alert message	**New group ('<group>') found in <file>.**
Example e-mail	A new group account has been found in the x86 World group file ([_1]) that is not present in the POWER group file. The relevant group entry found in [_2] is shown here: [_3] where the fields represent the group account, password, group id, and users that are members of this group, respectively. In order to stop receiving this email in future you have two options. The first option is to add the group [_4] to the group whitelist file found in `/etc/opt/powervm-lx86/group_ignore`. In this case you don't need to add the group to the POWER side and this group will be ignored in future. The second option is to add the group on the POWER side yourself. One way of doing this (but not necessarily the most complete) is to execute the following command in a native POWER shell: `/usr/sbin/groupadd [_5]` (This mail was generated by the cron script [_1] and can be disabled by setting the configuration variable `WORLD_CHECK_OR_SYNC=none` in `/etc/opt/powervm-lx86/config`)

Alert message	**Aliased group id ('<groupid>') found in <file>.**

Example e-mail	A group account has been found in the x86 World group file ([_1]) that shares the same group id ([_2]) with an account found in the POWER group file (`/etc/group`). The relevant group entry found in the x86 World file ([_3]) is shown here:
	
	[_4] and the relevant group entry found in the POWER file (`/etc/group`) is shown here:
	
	[_5] where the fields represent the group account, password, group ID, and users that are members of this group, respectively.
	
	Although highly unlikely, this could represent a serious security risk, since the identity corresponding to group id [_6] is ambiguous.
	
	In order to stop receiving this email in future you have two options. If you believe there to be no security risk involved then you can add the group id [_7] to the whitelist file found in `/etc/opt/powervm-lx86/gid_ignore` Otherwise, it is recommended that you rectify the problem by changing the group id of [_8] in the x86 World. However, before doing so you MUST first shut down PowerVM Lx86 by running the command:
	`/etc/init.d/powervm-lx86 stop`
	
	since there may currently be PowerVM Lx86 processes running as group [_9]. Next, you need to choose a new, unique group id for group [_10], ensuring that this group id does not already exist in either the x86 World or POWER group files. The simplest way to change the group id of group [_11] is to manually edit the x86 World group file, updating the group id accordingly. Finally, you must update the ownership of all files in the x86 World that are owned by group id [_12]. There are several ways to update the ownership on files in the x86 World, however here is one simple command you can run:
	`/bin/chown --from=:[_13] -R :<new group id> [_14]`
	
	CAUTION: Any mistakes made at this stage could damage both your x86 world and your POWER system.
	
	You should then be able to restart PowerVM Lx86 with the following command:
	`/etc/init.d/powervm-lx86 start`
	
	(This mail was generated by the cron script [_1] and can be disabled by setting the configuration variable `WORLD_CHECK_OR_SYNC=none` in `/etc/opt/powervm-lx86/config`)

Appendix. Known issues with PowerVM Lx86

There are some known issues with PowerVM Lx86. This section details those issues and any workarounds that are available. See the Release Notes for the most recent updates.

Access times on directories

As part of the operation of PowerVM Lx86, access times of directories may be updated more regularly than normally expected by a running x86 application due to the *jailing* mechanism. This is not expected to have an impact on any applications. This will not be addressed in a future release.

Argument lengths

The PowerVM Lx86 *jailing* mechanism adds the *X86WORLD_ROOT* (e.g. `/i386`) string to some system call arguments. This reduces the maximum length of arguments that can be made by a translated x86 application. This will not be addressed in a future release.

UTF-8 console

If the installer is run from a console that does not support UTF-8, some characters might be shown incorrectly. This will not be addressed in a future release.

Limitations with escapes

It is not possible to move an existing escape directory or file from within the x86 World, for example by using the **mv** command. To move an escape, delete the escape (by deleting the symlink file in the x86 World from a POWER shell), move the underlying POWER file or directory, and then use **linkx86** to create a new escape to the file or directory.

Stopped and zombie processes

If a translated x86 process is stopped, the process will not appear in the x86 `/proc` entries. The process is still running on the system and will be visible from a POWER shell, but will not be visible to x86 tools such as **ps** and **top**. If the process is *continued*, it will reappear in the x86 `/proc` entries and will be visible to the x86 **ps** and **top** commands again.

Local X11 display on RHEL 4

Graphical applications may not work when running on the local X11 display or using VNC due to an error in the X server and VNC Server. Running on the local X display may also require the command **xhost +** to be run from within a POWER shell before running the translated applications. Also, ensure that the display is set to a defined network name, for example, `DISPLAY=localhost:0.0`, not `DISPLAY=:0.0`. If you want to use a local display and connect directly to the local X server (rather than via TCP), an escape can be created for the .X11-unix socket directory. Use **linkx86** to create the socket from a POWER shell by typing the following command:

```
% linkx86 /tmp/.X11-unix
```

Colors in X11 applications

Some graphical applications, such as Adobe® Macromedia Flash, may display colors incorrectly when run under translation if displayed on a local POWER X server.

System resource conflicts

Certain system resources are shared between PowerVM Lx86 and the host POWER system. By default, PowerVM Lx86 and the POWER system use the same IP address. Attempts to bind to a certain port by a translated x86 application will fail if that port is already used by a running POWER application.

For example, if a POWER version of apache (**httpd**) is using port 80, an x86 instance of apache (**httpd**) will be unable to use that port. This can be resolved by running one of the apache instances on a different port number.

Disk performance

Disk performance may be slower on disks formatted using reiserfs than with ext2 or ext3. ext3 should be used as the default disk format for PowerVM Lx86.

Out of memory

If an x86 application uses up all of the available system memory, PowerVM Lx86 may exit with an error.

Accuracy of floating-point instructions

Due to the precision differences in floating point hardware implementations between the x86 and native POWER systems, the precise results of floating point instructions may not be the same as running the application natively on x86 hardware.

reboot command in an x86 shell

The **reboot** command has no effect when run from a translated x86 shell. This is the desired effect. If you wish to reboot the system, please run the **reboot** command from a POWER shell. If you want to restart just the x86 services running on the system, in a similar way to rebooting the x86 components running on the POWER system, restart the PowerVM Lx86 daemon with the following command:

```
% /etc/init.d/powervm-lx86 restart
```

For more details about the PowerVM Lx86 daemon, see “Starting the PowerVM Lx86 daemon” on page 24.

binfmt_misc not supported in x86 environment

The binfmt_misc Linux kernel feature is not supported in the x86 environment. If an x86 application running in the VxE attempts to register with binfmt_misc it will fail and PowerVM Lx86 may error.

PowerVM Lx86 translation process

The PowerVM Lx86 translation process is multi-stage and iterative.

Once an x86 application is loaded into memory, it undergoes a continuous process of translation and optimization. This is shown in the following diagram.

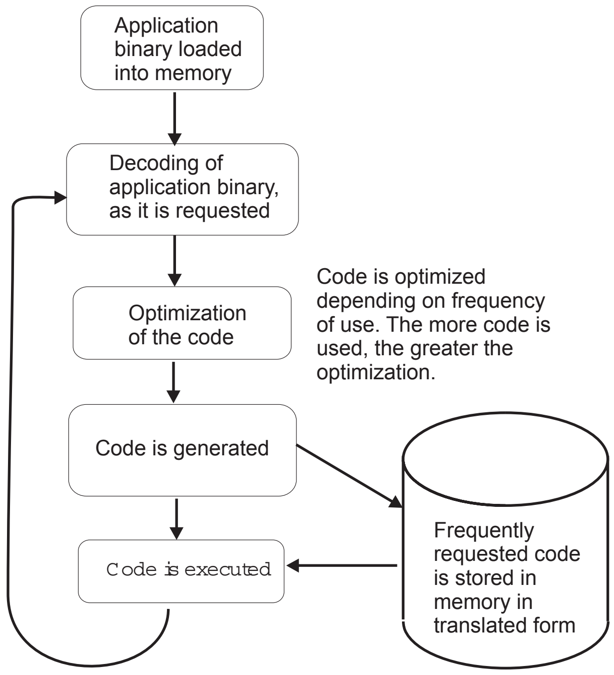

Application binary loaded into memory
Decoding of application binary, as it is requested
Optimization of the code
Code is optimized depending on frequency of use. The more code is used, the greater the optimization.
Code is generated
Frequently requested code is stored in memory in translated form
Code is executed

Figure 4. PowerVM Lx86 translation process

As the x86 application runs, PowerVM Lx86 dynamically translates the x86 code into POWER code. The translation is a three-stage process:

1. Decoding of application binary: x86 binary instructions are decoded as the translator requests them.
2. Optimization: The optimization is iterative, so more optimization is done on frequently-used code.
3. Generation of POWER code: The decoded x86 instructions are now translated into POWER code. Frequently-used code is stored in memory, so it does not need retranslating the next time it runs.

Linux on x86 system calls

x86 applications use system calls to request services from the x86 kernel.

The translator maps x86 system calls to their POWER equivalents.

System resources, binaries, and files

x86 applications need access to system resources, and to be able to access their own data and files as if they reside on an x86 system. They also need to be able to access files in the Linux on POWER system.

System resources such as x86 applications are translated as they are run. This means that the x86 applications can interact with POWER system resources as if they were native POWER applications. System resources include graphics, disk access, users, and network devices.

x86 binaries and libraries are all installed in one location on the same machine as the translator. The translator ensures that x86 applications can access the binaries and libraries that they require. See "PowerVM Lx86 concepts" on page 3 in this guide for more information about x86 World, jailing, and escapes.

Linux on POWER files and resources can be accessed by x86 applications. This may require some configuration.

Glossary

This is a glossary for the PowerVM Lx86 for x86 Linux Applications Administration Guide.

escape A mechanism that allows access to files on the local Linux on POWER file system, which are external to x86 World from the VxE.

host system The POWER system on which PowerVM Lx86 has been installed. It is able to run x86 applications within a VxE.

jailing The restriction of the view of the Linux file system from the VxE. Similar in concept to UNIX chroot.

Linux on POWER system A system with a POWER CPU running the Linux operating system.

Linux on x86 system A system with an x86 CPU running the Linux operating system.

Linux on POWER application A Linux application compiled for a Linux on POWER system.

Linux on x86 application A Linux application compiled for a Linux on x86 system.

native POWER application A Linux on POWER application running natively on a Linux on POWER system.

native x86 application A Linux on x86 application running natively on a Linux on x86 system.

native POWER shell The Linux shell running natively on the Linux on POWER host system.

native x86 shell The Linux shell running natively on a Linux on x86 system.

runx86 The command that runs an x86 binary within a Virtual x86 Environment.

powervm-lx86 The program that translates x86 applications so they can run on POWER systems.

powervm-lx86-daemon The daemon program that PowerVM Lx86 uses to communicate between translated x86 processes on the POWER system.

PowerVM Lx86 A product that enables POWER systems to run x86 applications alongside native POWER applications. No modifications, recompiling, or changes are needed to the x86 applications.

translator The powervm-lx86 program that handles the mapping of instructions and requests from the VxE onto the underlying Linux on POWER system.

Virtual x86 Environment (VxE) The method that PowerVM Lx86 uses to add Linux on x86 compatibility to Linux on POWER systems. Linux on x86 applications are encapsulated so the operating environment appears to be x86, even though the underlying system is POWER. This is achieved using the files and libraries in x86 World, the translator, and selective integration between the VxE and the POWER host system.

x86 application A Linux on x86 application running within a VxE on a Linux on POWER host system.

x86 shell The Linux shell running within a VxE on a Linux on POWER host system. Linux on x86 commands entered from the x86 shell prompt will also be run within a VxE.

x86 World A set of Linux on x86 libraries, commands, applications, and other system files that are installed in a directory on the POWER system.

Accessibility features

Accessibility features help users who have a physical disability, such as restricted mobility or limited vision, to use information technology products successfully.

The following list includes the major accessibility features:
- Keyboard-only operation
- Interfaces that are commonly used by screen readers
- Keys that are tactilely discernible and do not activate just by touching them
- Industry-standard devices for ports and connectors
- The attachment of alternative input and output devices

IBM and accessibility

See the IBM Accessibility Center at http://www.ibm.com/able/ for more information about the commitment that IBM has to accessibility.

Notices

The IBM license agreement and any applicable information on the web download page for IBM products refers You to this file for details concerning notices applicable to code included in the products listed above or otherwise identified as Excluded Components in the License Information document for the above-listed products ("the Program").

Notwithstanding the terms and conditions of any other agreement You may have with IBM or any of its related or affiliated entities (collectively "IBM"), the third party software code identified below are "Excluded Components" and are subject to the terms and conditions of the License Information document accompanying the Program and not the license terms that may be contained in the notices below. The notices are provided for informational purposes.

Please note: This Notices file may identify information or Excluded Components listed in the agreements for the Program that are not used by, or that were not shipped with, the Program as You installed it.

IMPORTANT: IBM does not represent or warrant that the information in this NOTICES file is accurate. Third party websites are independent of IBM and IBM does not represent or warrant that the information on any third party web site referenced in this NOTICES file is accurate. IBM disclaims any and all liability for errors and omissions or for any damages accruing from the use of this NOTICES file or its contents, including without limitation URLs or references to any third party websites.

This information was developed for products and services offered in the U.S.A.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not grant you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing

IBM Corporation

North Castle Drive

Armonk, NY 10504-1785 U.S.A.

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

IBM World Trade Asia Corporation

Licensing

2-31 Roppongi 3-chome, Minato-ku

Trademarks

Terms and conditions

PURPOSE.

Printed in USA

SA38-0650-05