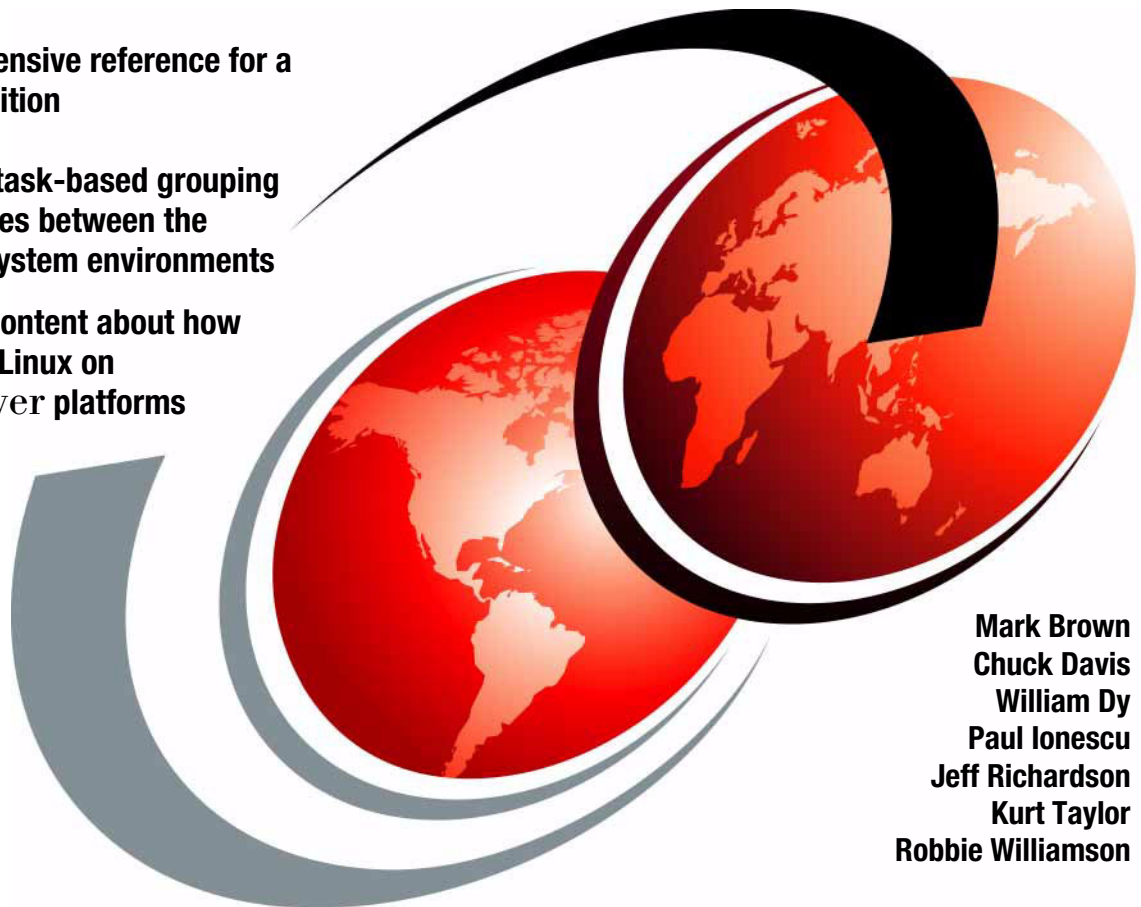


Solaris to Linux Migration: A Guide for System Administrators

A comprehensive reference for a quick transition

Presents a task-based grouping of differences between the operating system environments

Additional content about how to optimize Linux on IBM @server platforms



Mark Brown
Chuck Davis
William Dy
Paul Ionescu
Jeff Richardson
Kurt Taylor
Robbie Williamson



International Technical Support Organization

Solaris to Linux Migration: A Guide for System Administrators

February 2006

Note: Before using this information and the product it supports, read the information in “Notices” on page xiii.

First Edition (February 2006)

© Copyright International Business Machines Corporation 2006. All rights reserved.

Note to U.S. Government Users Restricted Rights -- Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

Contents

Notices	xiii
Trademarks	xiv
Preface	xv
The team that wrote this redbook	xvi
Acknowledgements	xviii
Become a published author	xviii
Comments welcome	xix
Part 1. Background and planning	1
Chapter 1. Introduction	3
1.1 What is Linux?	4
1.2 Linux kernels, distributions, and distributors	4
1.2.1 The Linux kernel	5
1.2.2 The Linux operating system	5
1.2.3 Linux distributions	5
1.3 How this book is organized	6
1.4 Linux administration overview	7
1.4.1 Documentation	8
1.4.2 Utilities	8
1.4.3 Standards	9
1.4.4 Support	10
1.5 Reference materials	10
Chapter 2. Planning for migration	11
2.1 Assemble the stakeholders	12
2.2 Set objectives and define scope	13
2.3 Assess workload and environment	14
2.3.1 Workload types	14
2.3.2 ISV applications	16
2.3.3 Custom application porting	16
2.3.4 Databases	16
2.3.5 Integration with existing network services	16
2.3.6 Other important infrastructure requirements	17
2.3.7 Special hardware	17
2.4 Assess skill requirements	17
2.5 Build a plan	18

Part 2. System administration differences guide	21
Chapter 3. Operating system installation	23
3.1 Basic system installation	24
3.1.1 Graphical or text installation	25
3.1.2 Other installation methods	25
3.1.3 System bundle options	26
3.1.4 Linux LVM consideration	27
3.1.5 Linux firewall	28
3.2 Advanced installation and automation	29
3.2.1 Installing using a serial interface as input/output device	29
3.2.2 Configuring a serial console device	30
3.2.3 Installing using remote display	31
3.2.4 Solaris Jumpstart	32
3.2.5 Red Hat Kickstart	32
3.2.6 SUSE AutoYaST	37
3.2.7 The PXE protocol	47
3.3 Network-based install: Solaris and Linux heterogeneous environments	47
3.3.1 Solaris system serving Linux network installation	47
3.3.2 Linux system serving Solaris network installation	50
Chapter 4. Disks and file systems	51
4.1 Disks and disk partitioning	52
4.1.1 Disks	52
4.1.2 Disk partitions	53
4.2 Disk-based file system management	55
4.3 Virtual file systems	56
4.4 Network File System (NFS)	58
4.5 Swap file systems	58
4.6 File system journaling	59
4.7 File system organization	60
4.8 AutoFSCK	62
4.9 AutoFS	62
4.10 Solaris Volume Manager to Linux LVM	65
4.11 VERITAS VxVM and VxFS	73
Chapter 5. Software management	75
5.1 Packages	76
5.1.1 Package management in Solaris	76
5.1.2 Package management in Linux	76
5.2 Patching	78
5.2.1 Patching in Solaris	79
5.2.2 Patching in Linux	79
5.3 Dependencies	79

5.3.1	Dependency management in Solaris	80
5.3.2	Dependency management in Linux.	80
5.4	Package distribution methods	80
5.5	Automated software management	81
5.5.1	Automated software management in Solaris	81
5.5.2	Automated software management in Linux.	81
5.6	Activating fixes after updating	82
5.6.1	Patch activation in Solaris	82
5.6.2	Patch activation in Linux	82
5.7	Compiling patches in Linux	82
Chapter 6.	Device management	85
6.1	Device access and configuration.	86
6.1.1	Device naming and access	86
6.1.2	Displaying device configuration information	89
6.1.3	Adding a device.	92
6.1.4	Hot-plug devices	93
6.2	Removable media devices	94
6.2.1	Supported types	94
6.2.2	Managing and accessing removable media	94
6.2.3	Formatting removable media	96
6.2.4	CDs and DVDs	96
6.2.5	Tape drives	98
6.3	Terminals and modems.	101
6.3.1	Terminal setup and initialization	102
6.3.2	Modem setup tools	103
6.3.3	Serial port management	103
6.3.4	Port monitoring	104
6.4	Distribution-based device management tools	105
Chapter 7.	Network services	107
7.1	IPv4	108
7.2	IPv6	113
7.3	Mixed IPv4 and IPv6 networks	114
7.4	Static and dynamic routing	114
7.5	IPSec and IKE	116
7.6	Network multipath	116
7.7	Network trunking	116
7.8	IP network services	117
7.8.1	inetd-based versus xinetd-based network services	117
7.8.2	DHCP	119
7.8.3	DNS.	120
7.8.4	NTP.	120

7.8.5 LDAP	120
7.8.6 NIS and NIS+	121
7.8.7 NFS	121
7.8.8 Web proxy and cache servers	122
7.8.9 Mail services	122
7.9 TCP wrappers	123
7.10 IP stateful firewalling	123
Chapter 8. Boot and system initialization	125
8.1 Booting a system	126
8.1.1 Booting types	126
8.1.2 Booting sources	127
8.1.3 Booting process overview	127
8.2 Run levels	131
8.3 Boot configuration files	132
8.3.1 /etc/inittab file	133
8.3.2 Run control files (rc files)	133
8.3.3 Disabling rc scripts	134
8.4 Shutting down	135
8.5 Network booting	135
Chapter 9. Managing system resources	139
9.1 Displaying system information	140
9.2 Resource management	140
9.3 Starting and stopping system services	141
9.3.1 Solaris system services	141
9.3.2 Linux system services	142
9.4 Scheduling and cron services	144
9.5 Quotas	145
9.6 Process accounting	146
9.6.1 Solaris	146
9.6.2 Linux	146
9.7 Remote system management services	147
9.7.1 Web-Based Enterprise Management (WBEM)	147
9.7.2 Simple Network Management Protocol (SNMP)	149
Chapter 10. Printing services	151
10.1 Overview	152
10.2 Linux CUPS	152
10.3 Client and server setup	153
10.4 Printer management using the lp commands	154
10.5 Printer types and CUPS support	156
Chapter 11. Users and groups	159

11.1 Basic user administration	160
11.2 Commands	160
11.2.1 Adding user accounts	160
11.2.2 Changing user account information	161
11.2.3 Removing user accounts	162
11.2.4 Home directory	163
11.3 Directory services	163
11.4 NIS and NIS+	164
11.5 User ID and group ID differences	165
Chapter 12. Monitoring and performance	173
12.1 Processors	175
12.2 Real and virtual memory	175
12.3 Physical media, software RAID, LVM, and file systems	176
12.3.1 Logical volume groups and logical volumes	177
12.3.2 File systems	178
12.4 Network	178
12.5 System and user processes	179
Chapter 13. Backup and restore	181
13.1 Common backup tools	182
13.2 Compression tools	182
13.3 File system backup and restore	183
13.3.1 Solaris overview	183
13.3.2 Linux overview	183
13.4 File system snapshots	184
13.5 AMANDA	185
Chapter 14. Security and hardening	187
14.1 Patching and security updates	188
14.2 Security hardening	188
14.2.1 Hardening tools	188
14.2.2 Auditing: ASET/BSM	189
14.3 Securing and removing services	189
14.3.1 inetd/xinetd	189
14.3.2 TCP wrappers	190
14.3.3 FTP	191
14.4 Kernel tuning for security	192
14.5 Logging	193
14.6 Access control lists	194
14.7 PAM	194
14.7.1 PAM module types	195
14.7.2 Limiting superuser login to secure terminals	196
14.7.3 Restricting user login	197

14.8	umask	197
14.9	SSH	197
14.10	IPSec and IKE	198
14.11	Kerberos	199
14.12	Warning banners	202
14.13	Firewalls	202
Chapter 15. Linux high availability overview		205
15.1	Introduction to Linux-HA	206
15.1.1	Migration scenarios	206
15.1.2	Some features of Linux-HA	206
15.2	Linux-HA services	207
15.2.1	Heartbeat	207
15.2.2	Cluster Resource Manager (CRM)	207
15.2.3	Consensus Cluster Membership (CCM)	208
15.2.4	Local Resource Manager (LRM)	208
15.2.5	STONITH daemon	208
15.3	Linux migration	208
15.4	Cluster environment considerations	209
15.4.1	Data sharing arrangements	209
15.4.2	IBM ServeRAID	209
Chapter 16. Shell scripting		211
16.1	Overview of the shell environment	212
16.1.1	Solaris shell environments	212
16.1.2	Linux shell environments	212
16.2	Public Domain Korn shell	213
16.3	Moving from ksh to bash	213
Chapter 17. Troubleshooting		215
17.1	Troubleshooting the booting process	216
17.2	Core files	216
17.3	Crash dumps	218
17.4	Logs	220
17.5	Permissions: File access problems	222
17.5.1	Problem: Command not found	222
17.5.2	Problem: File access	223
17.6	Printing	223
17.6.1	Troubleshooting remote printer connectivity	224
17.6.2	Troubleshooting local printers	224
17.7	File systems	225
17.7.1	Remote file systems	226
17.7.2	Software RAID	227
17.7.3	Logical volumes	227

17.8 Packages	227
17.9 root password recovery	229
17.10 Network	230
17.11 System and user processes	231
17.12 Diagnostic and debugging tools	232
Part 3. IBM eServer platforms	235
Chapter 18. IBM eServer xSeries hardware platform specifics	237
18.1 Installation considerations	238
18.1.1 xSeries	238
18.1.2 BladeCenter	239
18.2 Hardware and device differences	239
18.2.1 xSeries	240
18.2.2 BladeCenter	244
18.3 References	245
Chapter 19. IBM POWER technology hardware platform specifics	247
19.1 Planning	248
19.1.1 IBM eServer i5 and eServer p5	248
19.1.2 IBM eServer BladeCenter JS20	254
19.2 Booting and system initialization	259
19.2.1 IBM eServer i5 and eServer p5	259
19.2.2 eServer BladeCenter JS20	264
19.3 Linux installation	278
19.3.1 eServer i5 and eServer p5	278
19.3.2 eServer BladeCenter JS20	278
19.3.3 Red Hat	279
19.3.4 Novell SUSE	280
19.3.5 YaBoot OpenFirmware boot loader	283
19.4 eServer i5 and eServer p5 virtualization	284
19.4.1 Create a virtual I/O server partition profile	285
19.4.2 Create a client partition profile	291
19.4.3 Configure Linux on a virtual I/O server	295
19.4.4 Configure Linux on a virtual client partition	302
19.5 POWER technology platform service and productivity tools	303
19.5.1 Red Hat	304
19.5.2 Novell SUSE	304
19.6 References and further readings	304
Chapter 20. IBM eServer zSeries and IBM System z hardware platform specifics	307
20.1 Planning	309
20.1.1 Hardware resources	310

20.1.2	Software resources	310
20.1.3	Networking resources	311
20.1.4	Linux distributions and z/VM	311
20.2	S/390 and zSeries overview	312
20.2.1	Processing units	312
20.2.2	Memory	312
20.2.3	The channel subsystem	313
20.2.4	Tape drives	315
20.2.5	Disk drives	315
20.2.6	Network	316
20.2.7	Printers	319
20.2.8	Logical partition concepts	319
20.2.9	Virtualization	320
20.3	Installation methods and techniques	320
20.3.1	Preparing the z/VM guest resources	321
20.3.2	Server farms or cloned environments	321
20.4	Booting, system initialization, and shutdown	326
20.5	Device management	327
20.5.1	Linux reconfigurability	327
20.5.2	DASD hot-plug example	328
20.6	Performance monitoring and tuning	331
20.6.1	Performance monitoring	331
20.6.2	Performance tuning	333
20.7	Troubleshooting and diagnostics	334
20.7.1	z/VM troubleshooting and diagnostics	334
20.7.2	Linux troubleshooting and diagnostics	335
20.8	S/390 and zSeries-specific Linux commands	335
20.8.1	Packages specifically for the mainframe	336
20.8.2	Commands specifically for the mainframe	336
20.9	High availability	338
20.9.1	Hardware HA	338
20.9.2	z/VM HA	338
20.9.3	Automating Linux guest boot and shutdown	339
20.9.4	Linux-HA	340
20.9.5	Backup and recovery options	340
20.10	Security	341
20.11	References	342
Part 4.	Appendixes	343
	Appendix A. Tasks reference	345
	Packaging	346
	Installing and upgrading tasks	346

Booting and shutting down	348
User management tasks	351
Device management and configuration	352
Network management and configuration	353
NFS management and configuration	354
Managing system resources	354
Managing system services	355
Managing scheduling and cron	356
Managing quota	356
Managing process accounting	357
Printer management and configuration	357
Disk and file system management	358
Swap management	359
Logical volume management	359
General troubleshooting	362
Network troubleshooting	363
Appendix B. Commands and configuration files reference	365
Configuration and other files	366
Comparable commands	367
Common Solaris and Linux commands	369
Appendix C. UNIX to Linux Porting: A Comprehensive Reference (table of contents and sample chapter)	371
Table of contents	372
Chapter 1 Porting Project Considerations	373
Software Application Business Process	373
The Porting Process	374
Defining Project Scope and Objectives	378
Estimating	380
Creating a Porting Project Schedule	385
Porting Process from a Business Perspective	387
Annotated Sample Technical Questionnaire	387
Summary	394
Appendix D. Example: System information gathering script	397
Appendix E. Additional material	401
Locating the Web material	402
Using the Web material	402
Related publications	403
IBM Redbooks	403
Other publications	404

Online resources	405
How to get IBM Redbooks	411
Help from IBM	411
Index	413

Notices

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 give 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.

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law. INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.


This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrates programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs. You may copy, modify, and distribute these sample programs in any form without payment to IBM for the purposes of developing, using, marketing, or distributing application programs conforming to IBM's application programming interfaces.

Trademarks

The following terms are trademarks of the International Business Machines Corporation in the United States, other countries, or both:

AIX 5L™	Lotus®	ServeRAID™
AIX®	Micro-Partitioning™	System p5™
BladeCenter®	OpenPower™	System z9™
DB2®	OS/390®	System z™
developerWorks®	POWER5™	Tivoli®
DFSMSdss™	POWER™	TotalStorage®
Domino®	PR/SM™	VSE/ESA™
Enterprise Storage Server®	pSeries®	WebSphere®
ESCON®	RACF®	Workplace™
@server®	RDN™	xSeries®
FICON®	Redbooks (logo)  ™	z/OS®
HiperSockets™	Redbooks™	z/VM®
i5/OS®	RMF™	z9™
IBM®	RS/6000®	zSeries®
iSeries™	S/390®	

The following terms are trademarks of other companies:

iPlanet, CacheFS, Java, JavaScript, Portmapper, Solaris, Solstice, Solstice DiskSuite, Sun, Sun Microsystems, SunOS, SunScreen, and all Java-based trademarks are trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.

Microsoft, MS-DOS, Windows, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

Intel, Intel logo, Intel Inside logo, and Intel Centrino logo are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States, other countries, or both.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Linux is a trademark of Linus Torvalds in the United States, other countries, or both.

Other company, product, or service names may be trademarks or service marks of others.

Preface

The goal of this IBM® Redbook is to provide a technical reference for IT system administrators in organizations that are considering a migration from Sun™ Solaris™ to Linux®-based systems. We present a system administrator view of the technical differences and methods necessary to complete a successful migration to Linux-based systems, including covering the differences between major Linux distributions and IBM server and storage platforms.

This book is designed primarily to be a reference work for the experienced Sun Solaris 8 or 9 system administrator who will be working with Linux. It is not a Linux administration how-to book for beginning system administrators, but rather a guide for experienced administrators who need to translate a given Solaris system administration task to Linux.

We organize this book into four main parts.

In Part 1, “Background and planning” on page 1, we provide an introduction to the topic and a general purpose planning guide for migrations.

Part 2, “System administration differences guide” on page 21 covers a broad set of system administration topics. Chapters in this part focus on specific configuration management tasks and system functions for which a system administrator is typically responsible. In each chapter, we present the topics with the goal of identifying the major differences between how the tasks and functions are managed in Solaris and Linux. Wherever possible, we also highlight any differences in how those tasks are accomplished between Red Hat Enterprise Linux (RHEL) and SUSE Linux Enterprise Server (SLES). Because it would be impossible to provide a comprehensive reference about each topic in a single volume, we provide links to find more detailed information about the topics presented in each chapter.

In Part 3, “IBM eServer platforms” on page 235, we provide technical implementation details for installing and running Linux on IBM @server® platforms. We do not attempt to provide a complete reference for running Linux on each IBM platform. Instead, we focus on the Linux administration tasks that change due to hardware differences, such as boot configuration or virtualization functions. To accomplish this, we identify and explain how to leverage the additional industry-leading technologies in IBM @server xSeries®, IBM POWER™ technology (iSeries/pSeries®), and IBM @server zSeries® systems

and IBM System z™ that make them very powerful and flexible platforms for hosting Linux-based solutions.

Part 4, “Appendixes” on page 343 provides quick reference tables for looking up Solaris to Linux task equivalencies, comparing system configuration files and locations, and using comparable commands (different name, comparable function) and common commands (same name and purpose). In Part 4, we also include a sample chapter from the following related publication:

Mendoza, et al., *UNIX to Linux Porting: A Comprehensive Reference*,
Prentice Hall, 2006, ISBN 0131871099

Along with the sample chapter, we include link information so that you can order the *UNIX to Linux Porting* book at a discounted price.

The team that wrote this redbook

This redbook was produced by a team of specialists that came from around the world to work together at the IBM International Technical Support Organization in Austin, Texas, in October and November of 2005.

Mark Brown is a Senior Technical Staff Member at IBM Linux Technology Center in Austin, Texas. He is currently the technical lead for UNIX® to Linux porting efforts within the Linux Technology Center, and the IBM corporate representative to the IEEE POSIX and UNIX trademark standards working groups. Mark is a coauthor of the IEEE POSIX 1003.1 and 1003.2 specifications, as well as the Single UNIX Specification. His experience includes more than 20 years in UNIX and Linux operating system development, with an emphasis on C language libraries and ABIs.

Chuck Davis (CISSP-ISSAP) is a Security Architect for IBM Global Services. Chuck’s responsibilities are to author and support the IBM global e-business UNIX security policies, write vulnerability tests for the IBM internal penetration testing application, and support the IBM internal host-based intrusion detection solution. Chuck has been a speaker at information security conferences worldwide. Prior to working for IBM, Chuck spent six years working in the Internet industry as a systems security administrator, webmaster, and Internet engineer. Chuck holds certifications from Sun Microsystems™, Cisco Systems, and (ISC)2.

William Dy has been a member of the Canadian IBM Information Technology Services Americas (formerly known as IBM Global Services) since late 1996. He has worked on various platforms, including Sun SPARC, HP RISC, IBM RS/6000® (now known as pSeries), and other smaller UNIX-based systems. His current main concentration is the Sun Solaris operating system, and he provides

day-to-day operational support for various commercial outsourcing accounts in Canada, as well as some IBM internal servers. His interests include writing scripts and programs to automate specific tasks. He currently resides in Calgary, Alberta, Canada.

Paul Ionescu is a Senior IT Specialist working for IBM Global Services Romania. He has more than 10 years of Linux and 7 years of Sun Solaris systems administration experience focusing on networking and security. He is a Sun Certified System Administrator and Network Administrator for Solaris 9, a Certified Information Systems Security Professional (CISSP), a Cisco Certified Internetworking Expert (CCIE), a Cisco Certified System Instructor, and a Red Hat Certified Engineer (RHCE), and holds several other Linux, networking, and security certifications. He is currently focused on network engineering projects for the ISP, telco, and banking industries, and is also helping customers migrate from other operating systems to the Linux OS. Paul joined IBM in 2000.

Jeff Richardson is an Advisory Software Engineer in the IBM Software Group Linux Integration Center. Jeff has a bachelor of arts degree in Computer Science from Saint Edward's University and is an IBM Certified Database Administrator for IBM DB2® UDB V8.1 for Linux, UNIX, and Microsoft® Windows®. Jeff has worked for IBM for more than 23 years. His first assignment was an OS/MVS and z/VM® system operator and analyst. Since then, he has worked with IBM Middleware products on different operating systems and hardware platforms in functional and system test teams. Jeff currently develops educational material and provides pre-sales technical support for IBM Middleware products running on xSeries, pSeries, and zSeries for Linux.

Kurt Taylor is a Senior Software Engineer at the IBM Linux Technology Center, located in Austin, Texas. Kurt is currently a Technical Team Leader for the Linux System Management Team. He has worked at IBM for 10 years, and has more than 17 years of industry experience architecting, designing, and developing distributed network and system management products. He is currently working with the OpenPegasus project and the OSDL Desktop Linux work group. Kurt graduated from the University of North Texas with a bachelor of science degree in Computer Science.

Robbie Williamson is an Advisory Software Engineer working in the IBM Linux Technology Center (LTC). He is currently the IBM @server Red Hat Linux Test Architect. Robbie has a bachelor's degree in Computer Science and is completing his master's degree in Engineering Management from the University of Texas at Austin. He has more than seven years of UNIX/Linux experience, including IBM AIX® 5L™ technical support and testing, Hewlett-Packard HP-UX development, and Linux development and testing. He is also an open source project maintainer for the Linux Test Project.

Production of this IBM Redbook was managed by:

Chris Almond, an ITSO Project Leader and IT Architect based at the ITSO Center in Austin, Texas. In his current role, he specializes in managing content development projects focused on Linux and AIX 5L systems engineering. He has a total of 15 years of IT industry experience, including the last six with IBM.

Acknowledgements

A complete and detailed Redbook about a topic such as this would not have been possible without the support of the IBM Worldwide Linux team, the IBM Linux Technology Center, and the IBM Systems and Technology Group. The primary sponsors supporting development of this book include:

Terese Johnson, Susanne Libischer, Mark Brown, Susan Greenlee, Connie Blauwkamp, and John Milford

Additional content for this book was provided by **Alan Robertson**, an IBM employee and a lead developer in the High Availability Linux project.

The Redbook team would like to acknowledge the following people for their contribution of draft review feedback in support of this project:

Roland Caris, Shane Kilmon, Cliff Laking, Michael Maclsaac, Stephen Wehr, Elton Rajaniemi, and Jeroen van Hoof

The team would also like to acknowledge the support efforts of **Greg Geiselhart** and **Stephen Hochstetler** from IBM ITSO, **Lucy Ringland** from Red Hat Inc., **Steve Hansen** from Novell Inc., and our editor for this book, **Elizabeth Barnes**, from our ITSO Authoring Services team.

Become a published author

Join us for a two- to six-week residency program! Help write an IBM Redbook dealing with specific products or solutions, while getting hands-on experience with leading-edge technologies. You'll team with IBM technical professionals, Business Partners and/or customers.

Your efforts will help increase product acceptance and customer satisfaction. As a bonus, you'll develop a network of contacts in IBM development labs, and increase your productivity and marketability.

Find out more about the residency program, browse the residency index, and apply online at:

ibm.com/redbooks/residencies.html

Comments welcome

Your comments are important to us!

We want our Redbooks™ to be as helpful as possible. Send us your comments about this or other Redbooks in one of the following ways:

- ▶ Use the online **Contact us** review redbook form found at:

ibm.com/redbooks

- ▶ Send your comments in an e-mail to:

redbook@us.ibm.com

- ▶ Mail your comments to:

IBM Corporation, International Technical Support Organization
Dept. JN9B Building 905
11501 Burnet Road
Austin, Texas 78758-3493



Part 1

Background and planning

This part provides an introduction to the topic and a general-purpose planning guide for migrations.



Introduction

This chapter introduces Linux and the topic of the system administration of the Linux operating system from the perspective of administrators familiar with the Solaris operating system versions 8 or 9 and who are planning their first move to a Linux system.

This chapter includes the following topics:

- ▶ What is Linux?
- ▶ Linux kernels, distributions, and distributors
- ▶ How this book is organized
- ▶ Linux administration overview
- ▶ Reference materials

1.1 What is Linux?

In 1991 Linus Torvalds, back then a university student of computer science at the University of Helsinki, sent a historic post to the MINIX¹ news group announcing that he was working on a new free operating system and was interested in getting feedback about features that should be included in Linux. A few weeks later, Linux version 0.01 was released on the Internet, and its development became a shared effort as developers downloaded code, tested it, tweaked it, and sent it back to Linus, who then integrated them into his project.

New versions came rapidly, and the amount of followers grew as the code went worldwide through FTP sites. Soon Linux became popular among students and developers and the project expanded. Linux was licensed under the GNU General Public License (GPL), ensuring that the source code had to be included and it would be free for everyone to copy, study, and change. You can read the GPL:

<http://www.gnu.org/copyleft/gpl.html>

Although Linux itself was free, commercial vendors moved in soon in order to make money by gathering and compiling software and thus creating a package easy to install and distribute, more like the other, more familiar commercial operating systems.

Today, Linux has been ported to many other platforms, from hand-held devices to mainframes. It is the fastest growing OS in the server market and is used in enterprise as well as single-purpose server installations. The open nature of its development process has led to rapid and frequent additions of features and technologies. Added support for clustering, for example, has enabled highly available and scalable solutions using Linux.

Linux's roots in the UNIX operating system philosophy make it stable, reliable, and expandable, while at the same time, familiar to experienced UNIX users and administrators.

1.2 Linux kernels, distributions, and distributors

The term "Linux" (which is a trademarked term, held by Linus Torvalds) is often used in different ways: the operating system *kernel* itself, the entire *operating system* built around this kernel, or a *distribution*, in other words, a packaged Linux OS product.

¹ Small UNIX clone designed for IBM PCs and compatibles. It is ideal for educational purposes due to its small size, microkernel-base design, and full documentation. Although MINIX comes with the source code, it is copyrighted software.

1.2.1 The Linux kernel

The term *kernel* in this sense is meant in the same sense as in UNIX, that is, the core layer of the OS that controls hardware, manages files, creates processes, and so forth. Also included in this sense are the set of loadable modules and drivers that come with the basic kernel.

The kernel is maintained by a team of volunteers, headed by Linus Torvalds, who work on maintenance and new features. They are aided in this by developers from all over the world (including IBM) who contribute fixes (patches) and new feature code as candidates for inclusion. Although a person or organization is free to make a copy of the Linux kernel source code and make their own version or variation of it, Linus' kernel project is considered the one official Linux kernel.

1.2.2 The Linux operating system

Used in this sense, the term *Linux* includes all of the drivers, modules, libraries, tools, utilities, and applications that make up an operating system based on the Linux kernel. These are almost always entirely made up of open source products:

- ▶ The base C and C++ libraries, compiler, and associated development tools, base utilities (**ls**, **ps**, **mv**, **etc**), and many other packages of libraries and utilities come from Free Software Foundation projects (GNU).
- ▶ Other tools, from the shells (**bash**, **pdksh**) to scripting languages (Perl, PHP) to Internet daemons (**xinetd**, **wu-ftpd**) come from individual open projects.
- ▶ Some parts come from large, organized open projects (X, GNOME desktop, KDE desktop, Apache Web Server).

These packages and applications are almost exclusively distributed for use under open licenses (source code must be made freely available) by anyone who wishes to use them. The most widely known and used license is the Free Software Foundation's General Public License (GPL). Under the GPL or the other licensing terms of open software packages, anyone can produce a "Linux OS" or variant thereof. There is no one official Linux OS.

1.2.3 Linux distributions

Distributions are Linux operating systems, made up of a kernel and a collection of the necessary tools and applications to make a functioning, useful, stable operating system for general use. A distribution has taken this collection and produced a packed, installable product that can be easily distributed. Anyone can create a distribution, just as anyone can create a Linux OS or use the Linux kernel. There is no one official Linux distribution.

Linux distributions all have one thing in common: They provide Linux in source code and binary form. The base code is freely redistributable under the terms of GNU General Public License (GPL) and any non-free and contributed software has to be clearly marked as such. Distributions divide into two main categories:

- ▶ Non-commercial distributions tend to focus on creating technical excellence and follow a design philosophy or manifesto. These distributions do not offer any direct support other than mailing lists and bug reports. They are freely available on the Internet for download, and for the cost of materials and shipping, some can be purchased as a shrink-wrapped product with manuals.
- ▶ Commercial distributions offer training, certification, and support services in addition to their software, which is commercially distributed through official dealer channels or downloadable from the company's Web sites for payment. Most commercial distributions base themselves on non-commercial ones and are available for free download. However, any commercial software bundled with the product is only available on a separate CD for purchase.

Some of the best known and widely-used distributions include Red Hat, SUSE, Mandriva, and Debian. This book describes Linux using the operating system sense and in terms of two major distribution vendors:

- ▶ Red Hat (Red Hat Enterprise Linux 4)

Red Hat is a well-known Linux and open-source provider. It was founded in 1993. Red Hat Enterprise Linux runs on multiple architectures and is certified by enterprise software and hardware vendors.

Red Hat is based on the Linux kernel version 2.6.

- ▶ Novell SUSE (SUSE Linux Enterprise Server 9)

The first distribution of SUSE Linux began in 1993. The company was acquired by Novell in 2004.

SUSE Linux Enterprise Server 9 (SLES9) is a secure, reliable platform for open-source computing, supporting a range of hardware platforms and software packages. It also provides open APIs and other development tools to help simplify Linux integration and customization.

SLES9 includes the Linux kernel version 2.6.

1.3 How this book is organized

This book is designed primarily to be a reference work for the Solaris 8 or 9 system administrator who will be working with Linux. It is not a Linux administration how-to book for beginning system administrators, but rather a

guide for experienced administrators that need to translate a given Solaris administration task to Linux. We organize this book into four main parts:

- ▶ Part 1, “Background and planning”
- ▶ Part 2, “System administration differences guide”
- ▶ Part 3, “IBM eServer platforms”
- ▶ Part 4, “Appendixes”

Each section of the book attempts to outline the considerations for the experienced Solaris system administrator to keep in mind, the new and different tools and features, and where to go for further in-depth information about the topic. The system administration differences guide section provides task-based technical topics in terms of two major Enterprise Linux server distributions (Red Hat Enterprise Linux and SUSE Linux Enterprise Server) where a “generic” hardware platform is assumed. The IBM eServer platforms part outlines the administration tasks that change due to hardware differences, such as boot configuration or virtualization functions. This book also includes appendixes, containing direct side-by-side task, command, and configuration file comparisons, for use as a quick reference.

1.4 Linux administration overview

As mentioned earlier, the Linux kernel and operating system are based on the UNIX philosophy and practice. Also, Linux contains almost all of the features described by the IEEE POSIX 1003.1-2001 standard for interfaces, commands, and utilities. These two facts will make Linux seem very familiar to a Solaris system administrator.

Some of the many basic features in common between Solaris and Linux include:

- ▶ Users, groups, and a “root” superuser
- ▶ File system and directory hierarchies
- ▶ **init**, boot configuration/rc.* scripts
- ▶ Processes and POSIX threads
- ▶ Networking stacks and services
- ▶ ISO C/C++ library, basic utilities, and basic shells
- ▶ vi, emacs, and other editing tools
- ▶ Accounting and log files
- ▶ X Window System

This is just a short list of commonalities. Almost all of a Solaris administrator's skills and experience should transfer easily to Linux, with a short learning curve to pick up the differences in the common tools (in most cases small) or learn the equivalent new tools (in some cases).

1.4.1 Documentation

Documentation for Linux comes in several forms, the most common of which are manual pages (**man**) and HTML (either on the system or over the network). Almost all commands can be referenced through the **man** command, which works in the same way as on Solaris and is organized in the same manner (sections 1,2, and so on). System guides are usually provided in HTML or PDF format.

Some documentation, particularly for parts of the system that come from Free Software Foundation projects, is documented using the **info** tool. The C library (GNU libc, or glibc) is one example of this. This is a command line tool that operates as a full-screen, text-based GUI. It has an extensive built-in help system, but requires some time to learn. For basic information about the **info** tool, use the command **man info**.

1.4.2 Utilities

Some of the most common software packages that are used to manage your Linux servers include:

- ▶ YaST and YaST2 for SUSE Linux

YaST stands for yet another system tool and is a system management GUI. YaST can be started by typing the following command as root on the Linux command line:

```
~> yast
```

It contains modules for managing the installation and configuration of the system features.

- ▶ Red Hat **setup** and **system-*** utilities

setup is the Red Hat implementation of the Linux configuration tool. It is used to configure and manage your Linux server. As system administrator (root) on the Linux command line, start **setup** by typing the following command on the command line (it will ask for the root password):

```
~> setup
```

There are also individual GUI-based utilities for configuration of various system features in the `/usr/bin` path, as shown in Table 1-1 on page 9.

Table 1-1 Red Hat system configuration GUI tools

system-cdinstall-helper	system-config-printer-gui
system-config-authentication	system-config-printer-tui
system-config-boot	system-config-rootpassword
system-config-date	system-config-samba
system-config-display	system-config-securitylevel
system-config-httpd	system-config-securitylevel-tui
system-config-keyboard	system-config-services
system-config-language	system-config-soundcard
system-config-lvm	system-config-time
system-config-mouse	system-config-users
system-config-netboot	system-control-network
system-config-network	system-install-packages
system-config-network-cmd	system-logviewer
system-config-network-druid	system-switch-im
system-config-nfs	system-switch-mail
system-config-packages	system-switch-mail-nox
system-config-printer	

► VNC

Virtual Network Computing (VNC) is free software that enables you to view and interact with another computer, which might or might not have the same operating system. It gives you a remote desktop view of the other system. VNC or its equivalent is shipped with most Linux distributions. However, the default mode in VNC on Linux provides the connecting party with a new desktop screen, not the console screen as under Microsoft Windows.

1.4.3 Standards

Much like the Solaris OS, there are specifications that cover operations in the Linux space as well. The Linux Standard Base (LSB, <http://www.linuxbase.org/>) covers the implementation of a conforming Linux environment in much the same way that the Single UNIX Specification (SUS, Open Group) covers the UNIX environment. It covers application programming interfaces (APIs), utilities, software packaging, and installation, and additionally

specifies ABIs for the most common Linux platforms. We recommend that you investigate LSB support when choosing a Linux distribution for use.

Linux offers IEEE POSIX compliance where possible in both the API and utilities sets. There is also significant overlap in conformance with the Single UNIX Specification, though Linux cannot be called UNIX.

1.4.4 Support

Because the distributions themselves are made up of free software, the business model of Enterprise Linux distribution vendors typically emphasizes the selling of support and services related to their offerings. These services range from self-help advice forums (for the lower-end or desktop systems) to full 24x7x365 enterprise-level maintenance with support level contracts. Additionally, third parties (including IBM) offer support services related to Linux.

Consider support and updates when choosing a distribution. Some of the larger distributors also offer Linux system administration courses.

1.5 Reference materials

There are many sources for technical information about Linux. Some additional sources related to topics discussed so far in this book include:

- ▶ Linux and Linux solutions in general:
 - Siever, E., et al., *Linux in a Nutshell*, O'Reilly Media, Inc., 5 Edition, 2005, ISBN 0596009305
 - IBM Redbook *Linux Handbook A Guide to IBM Linux Solutions and Resources*, SG24-7000
- ▶ The Linux Documentation Project, online guides to almost any Linux topic:
<http://www.tldp.org/>
- ▶ Open source licenses and how they work:
 - The Free Software Foundation:
<http://www.fsf.org/>
 - The Open Source Initiative:
<http://www.opensource.org/>
- ▶ Application development:
Linux Standard Base Team, *Building Applications with the Linux Standard Base*, IBM Press, 2004, ISBN 0131456954



Planning for migration

This chapter provides some useful information regarding the non-technical and technical issues involved with a migration to a new platform in an established IT environment. Migrations, whether in a data center or on client workstations, involve considerations beyond simple OS operation. In many cases, the move to (or addition of) a new platform involves organizational and strategic changes as well. In this chapter, we cover some of the important planning steps involved in a migration to a Linux platform.

This chapter includes the following topics:

- ▶ Assemble the stakeholders
- ▶ Set objectives and define scope
- ▶ Assess workload and environment
- ▶ Assess skill requirements
- ▶ Build a plan

2.1 Assemble the stakeholders

As the name implies, these are the people with a “stake” in the migration. This group is not limited just to the people responsible for directly maintaining the systems. End users of the services (especially if it is a client workstation) and business managers have an interest in a successful move that meets the goals of the organization.

A meeting of stakeholders (or representatives of large groups of stakeholders) is a good way to set expectations and to perform some of the other planning considerations put forward in this chapter. A meeting of this group of people will be a good way to identify if any IT environment or administrator, manager, and user skills changes are needed, for example. These will also be the people to whom status and results milestones are reported.

Many times as a system administrator, you will not be the one proposing or funding the migration; instead, you are usually the implementor of a migration drive decided by others. In order to make sure that all interests are taken into account, it is still a good idea to request a meeting of the key people asking for and affected by the move.

Communicating the change

Such meetings are also good ways to open communication channels. The effect of a good communications plan will be to flatten out the negative aspects of the acceptance curve, as shown in Figure 2-1.

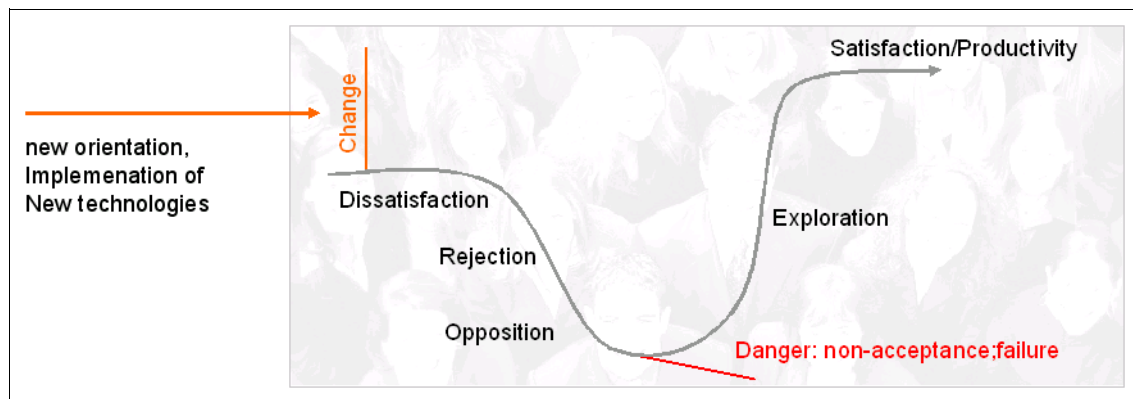


Figure 2-1 Acceptance of new technologies

A communications plan coupled with proper training on the new system should minimize the number of users that fall into the rejection/opposition mode, cause

users to start out with acceptance instead of dissatisfaction as the initial response, and lead to a quick transition into exploration and productive use.

Regarding the IT support team, these same issues have even more importance. A strategic decision to switch the OS can cause the impression of disapproval of the work they have done so far. It might give staff the impression that their current skills are being devalued. It probably will not be easy to convince an administrator of Solaris systems that migrating to Linux is a good strategy unless you can also convince them that the organization is prepared to make a significant investment in upgrading their skills as well.

2.2 Set objectives and define scope

You should be able to articulate the objectives for your Linux migration and relate them to your key business drivers. Whether you are trying to gain efficiencies by reducing costs, increasing your flexibility, improving your ability to support and roll out new application workloads, or some other key business drivers, set up objectives that line up with these. Even the smallest of migrations should be able to do this, and it will help guide your planning.

Defining metrics (increased performance? more uptime?) up-front during the project will help your team stay focused. Be sure that you will have a means of tracking the metrics. Getting stakeholder agreement on your metrics early in the project will help ensure executive and user support.

Often, the migration to Linux will be accompanied by other objectives. For instance, some customers upgrade their database at the same time to get the latest features and performance enhancements and to obtain support that lines up well with the latest distributions of Linux. As with any project, the scope must be well defined to prevent project overrun, but it is also important that you have a means to manage additions to the plan as business needs dictate.

Because cost is often a key motivator for migrating to Linux, give careful consideration to identifying where cost reduction is targeted. Identify metrics for defining return on investment prior to beginning migration activities.

Questions to ask in this area include:

- ▶ What are the tactical goals of the migration?
- ▶ What measurements of success will validate achieving those goals?
- ▶ What are the financial aspects of the migration (budget, return on investment, justification, and so on)?
- ▶ What are the strategic goals for the migration? How does the migration fit into the overall IT strategy for the organization?

The system administrator's perspective

While the objectives and scope of the migration will determine the size of the work involved, it is meeting the metrics (goals) that will guide everything from the hardware choices to the configuration of the various subsystems in the OS. Most of the considerations can be tackled the same way as in a move to another flavor of commercial UNIX. However, for many areas (security and firewalls, mail service, and databases are a few), there are multiple tools and packages available to achieve your goals.

Even though this book outlines the differences between Solaris and Linux (from a system administrator point of view) and lists many of the choices (in those areas that have them) as well, we strongly advise that you continue your research beyond this book as necessary to determine which of the variety of tools for a given task best fits your metrics. For example, methods for evaluation of PostgreSQL, MySQL, or a commercial database such as IBM DB2 to determine which is best for your use is beyond the scope of this book. But as administrator of the system running the database, such a choice can strongly affect how you initially configure the system and what types of tools you use for ongoing management of the system.

2.3 Assess workload and environment

The decision regarding which workloads to migrate needs to be tied to your overall objectives. A Linux migration usually does not have to be an “all or nothing” approach. Many organizations choose to start small to gain experience with Linux before migrating larger or mission-critical services to Linux. The following sections present some general considerations that should be factored into this decision process.

2.3.1 Workload types

The following sections show some sample workloads, roughly arranged from simple to complex, from the system administrator's perspective. These can vary in order of complexity depending on a given organization's environment; however, these are some general guidelines, based on IBM Global Service's experience with migrations.

File and print infrastructure

This type of workload can be moved to Linux, as Linux is easily configured to provide file and print service to a wide range of clients, from UNIX-based workstations to Windows. Linux distributions ship with all of these tools available in their default installations.

Security and directory infrastructure

Linux offers full LDAP, Kerberos, Network Information Service (NIS), firewalling, and other tools in this area that meet the appropriate industry standards. In fact, some of the most modern versions of security services were available from the major Linux distributors earlier than on other non-Linux based systems. LDAP schemas might differ from Solaris to Linux, and work to configure the proper integration between Kerberos levels might also be needed. If you are using NIS+ on Solaris, plan on transitioning to LDAP or NIS.

E-mail

Linux offers full sendmail, IMAP, and POP3 support, but at least one major Linux distributor (Novell SUSE Linux Enterprise Server) now offers the new postfix MTA as the default package (while still offering sendmail as an option). The IMAP and POP daemons (and clients) will each have a learning curve for configuration and use. End-user clients should not see a difference between the Solaris and Linux mail service.

Database serving

This type of migration will not only involve an operating system change, but will also involve moving a major application and its data. Both IBM DB2 and Oracle have Linux offerings, in addition to the open-source MySQL and PostgreSQL packages. The IBM Redbook *Up and Running with DB2 for Linux*, SG24-6899, might be helpful for information about this topic.

Web and application serving

A full suite of Web services utilities are shipped with most Linux distributions, with all of these containing the Apache HTTP server package. Perl, PHP, and Java™ (versions from IBM as well as Sun) are all available natively. You will need to research the needs of the application/service to be offered and how to make those things (such as the middleware and all of its prerequisites) available on the Linux platform.

Messaging

Messaging systems incorporate some elements of high availability systems and introduce scalability and networking factors as well. Research will be needed to properly specify your Linux equivalent system requirements. IBM offers an entire suite of messaging middleware (our WebSphere® and WebSphere MQ lines) for Linux.

Enterprise resource planning (ERP)

ERP systems tend to be cross-functional and enterprise-wide. Therefore, there are many interactions with other systems and processes for which to plan and account. The system administrator will need to carefully plan for successful integration in addition to the software and OS changes.

Business Intelligence

These systems incorporate several types of software (such as data mining and decision support applications) and use data from several different sources. In addition to the software research involved, the system administrator will need to plan for system communication and workload management issues.

2.3.2 ISV applications

If you are running ISV applications on systems that you are targeting for migration, you need to determine if the vendors provide compatible versions that support the distribution of Linux that you plan to use. Many ISV applications have other third-party dependencies. Be sure to map out all ISV dependencies, including middleware. Most leading middleware products are available on Linux, and there are many open source alternatives.

2.3.3 Custom application porting

Applications developed in-house might require porting and testing on Linux. The effort involved can vary greatly depending on how portable the application code is. Open source and commercial tools are available to help with tasks such as assessing the portability of your applications. IBM Global Services, as part of their migration services offerings, uses tools developed in cooperation with IBM Research to help with code assessment and conversion.

2.3.4 Databases

Consider whether you want to move the database and the application, or just one or the other. Database administration is a different but related skill to system administration. These skills are as equally transferable to Linux due to the similarity to UNIX-based systems and the fact that the most popular database offerings on the market are also available for Linux now.

2.3.5 Integration with existing network services

A part of your planning will be based on whether you are migrating a client or a server platform. While Linux can work equally well in either role, the differences include what type of distribution you install and what hardware configuration you use. While this book will serve you well in either capacity for the purposes of system administration, the IBM Redbook *Linux Client Migration Cookbook A Practical Planning and Implementation Guide for Migrating to Desktop Linux*, SG24-6380, might be a helpful reference as well.

Integration with other Solaris systems

Typically, the Linux migration will occur slowly or as a part of a pilot project, and thus the Linux system will be introduced into an environment where the interaction with Solaris systems is a requirement. You can reference this book in the research stage in order to work out the planning for interoperability in the administration tools and automation.

Integration with other Windows systems

Linux offers several tools for managing integration with Microsoft Windows systems. There are many resources available for further research to plan for integration points such as file sharing, printing, and so on.

2.3.6 Other important infrastructure requirements

You need to closely analyze how your existing functional requirements have defined the design of your current infrastructure. How will you meet these requirements in a Linux-based environment? Factor in your requirements for security, availability, operations, maintenance, and so on. What infrastructure is needed to support your specific workloads? Many of the same infrastructure products available for UNIX are also supported on Linux.

2.3.7 Special hardware

Are there special devices that will be supported? Printers, storage, or other hardware necessary to your operation must be investigated for Linux support.

2.4 Assess skill requirements

One of the key factors that is helping to accelerate migrations from UNIX to Linux is the availability of skills and the ease of transferring UNIX skills into a Linux environment. UNIX and Linux are similar in many respects, enabling system administrators to easily switch to a Linux environment. Also, many of the same tools that they use on UNIX are available on Linux. Still, a skills assessment should be a part of your migration planning so that you can identify areas where skills gaps may exist and plan for how you will cover any skills gaps. Several avenues for Linux training are available, including IBM, Linux distributors, and others.

Hardware-specific skills

Although almost all of the skills related to managing a Linux system are the same regardless of platform, some of the activities related to the interaction of Linux with the underlying hardware and maintenance of the hardware itself are different

based upon the hardware type. This is especially true for larger systems, including IBM @server POWER and zSeries platforms. Administrator education might be required for these systems.

A training option for client users

Many Linux distribution vendors are now providing live or bootable CD-ROM-based distributions. One notable live CD distribution is provided by Knoppix:

<http://www.knoppix.com>

The following description from Knoppix further explains what a live CD provides:

KNOPPIX is a bootable CD with a collection off GNU/Linux software, automatic hardware detection, and support for many graphics cards, sound cards, SCSI and USB devices and other peripherals. KNOPPIX can be used as a Linux demo, educational CD, rescue system, or adapted and used as a platform for commercial software product demos. It is not necessary to install anything on a hard disk. Due to on-the-fly decompression, the CD can have up to 2 GB of executable software installed on it.

A live CD can be used to provide users with the ability to run a bootable Linux system on their desktop. They can use it to test and evaluate the UI, applications, and other facets of the Linux client, before an actual client migration occurs. And this can be done without harming the host operating system installation on the local hard disk. Another benefit of using a live CD as part of a migration plan is for detection of hardware and device problems. The live CD helps validate proper hardware detection and device support and identifies any issues prior to actual client migration.

2.5 Build a plan

A key success factor is a well-built plan. Understanding your objectives and having metrics with stakeholder involvement and agreement help provide a good base from which to build a plan. Be sure to build in all key requirements, such as timing, resource needs, and business commitments such as service level agreements (SLAs). Also, be sure to include related aspects of the migration, such as new workloads, infrastructure, and consolidation.

As with any change in an organization, cultural change might need to be managed. Anticipating change and involving affected teams early are good ways to manage culture change issues. For example, support staff for hardware might be comfortable with UNIX-related hardware support, where to go for help, and so on. The “experts” in the prior environment might be less open to change if they feel threatened by new ways of doing things where they are not the “expert.”

Build in requirements

Consider the following areas:

- ▶ Resources

What hardware will be required? Software? Identify what housing issues are required (are the electrical and cooling inputs equal?). What staff will be required to help with the crossover?

- ▶ Education

Does the staff have adequate Linux education? Are there special skills required for the hardware or Linux/hardware combination?

- ▶ Service level agreements

While the installation, configuration, and testing of the change is taking place, what are the support mechanisms (both yours and those of any vendors)? What are your commitments to current stakeholders while you are performing the migration?

- ▶ Related aspects to project

Be sure to set out what other things are happening in addition to the basic system changeover.

- ▶ Identify skills-related requirements

Include documentation time

When installing and configuring the new system, documenting (even if it is just a short set of notes) will help in reproducing the process and in troubleshooting problems should they appear.

Add a testing period

After the new system is installed, initialized, and configured, there should be time in the schedule for testing the system. Does it interoperate correctly? Can it handle the expected load? Your plan should have some metrics for judging the success of the test period.

Identify and time crossover

It is important to understand what the expected impact will be to the expected usage of the system during the actual migration period. Many administrators schedule crossovers to new systems during “low-load” periods to minimize disruption of services. In other environments, such as those with no “light load” times, there will be a period where both new and old systems both remain online until the load can be moved entirely to the new system. This can also be used as the “testing period” for the migration in these environments.



Part 2

System administration differences guide

In this part of the book, we provide chapters describing system administration differences, where each chapter focuses on a specific functional area. We assume that the reader is an experienced Solaris system administrator who is undertaking (or considering) a migration to Linux. Therefore, each chapter assumes reader knowledge of how the task topic is performed on a Solaris system and compares that with the procedures for doing the similar task on Linux.

In each chapter, we attempt to “match up” Solaris GUI tools, commands, and configuration files with Linux-based counterparts, where they are similar. We also describe differences in or lack of comparable functionality. The chapters include pointers to the appropriate Linux guides and “how-to” documentation.

These chapters are not a substitute for the Linux documentation, but rather a “cross-reference” work for an administrator trying to find out what tools are available on Linux to perform a given job.

In this part, we focus on the aspects of system administration in Linux that are hardware-neutral. Part 3, “IBM eServer platforms” on page 235 covers areas of administration that are unique to a given IBM @server platform (POWER technology-based, xSeries, and zSeries systems). Throughout this section, we make references to two of the leading Linux enterprise server products, Red Hat Enterprise Linux (RHEL) and SUSE Linux Enterprise Server (SLES).

Note: We assume that the reader is an experienced Solaris system administrator who is undertaking (or considering) a migration to Linux. The content in this part of the book should not be used as a substitute for detailed Linux system administration documentation, but rather as a “cross-reference” work for an administrator trying to find out what tools are available on Linux to perform a given job.

The differences guide portion of this book covers the following functional areas:

- ▶ Chapter 3, “Operating system installation” on page 23
- ▶ Chapter 4, “Disks and file systems” on page 51
- ▶ Chapter 5, “Software management” on page 75
- ▶ Chapter 6, “Device management” on page 85
- ▶ Chapter 7, “Network services” on page 107
- ▶ Chapter 8, “Boot and system initialization” on page 125
- ▶ Chapter 9, “Managing system resources” on page 139
- ▶ Chapter 10, “Printing services” on page 151
- ▶ Chapter 11, “Users and groups” on page 159
- ▶ Chapter 12, “Monitoring and performance” on page 173
- ▶ Chapter 13, “Backup and restore” on page 181
- ▶ Chapter 14, “Security and hardening” on page 187
- ▶ Chapter 15, “Linux high availability overview” on page 205
- ▶ Chapter 16, “Shell scripting” on page 211
- ▶ Chapter 17, “Troubleshooting” on page 215



Operating system installation

This chapter describes the difference in the how Sun Solaris and Linux do their installations. Both have the option to use either a text-based or GUI installation format and allow installing from a different media sources.

This chapter contains the following topics:

- ▶ Basic system installation
- ▶ Advanced installation and automation
- ▶ Network-based install: Solaris and Linux heterogeneous environments

3.1 Basic system installation

In this section, we discuss the basic installation procedure with a local CD-ROM or DVD-ROM drive. Both Solaris and Linux offer other types of installation methods, which we cover in 3.1.2, “Other installation methods” on page 25.

Local CD/DVD installations are very much the same for Solaris and Linux. The installation program prompts for information when it is required. A menu of options is presented in some parts so that you can select the appropriate choice or choices.

As with a Solaris installation, Linux will perform the following operations.

Note: The following list is not in any specific order and is a not complete list of the options that will be presented to you. It is a representation of the basic information that will be presented. When using another installation method, these prompts might not be displayed if they are preconfigured in the installation procedure, or you might have extra options.

- ▶ Probe the hardware for devices and load the appropriate device drivers.
- ▶ Assign a host name.
- ▶ Set up a network setup type:
 - DHCP or static IP.
 - If static IP, you will be asked to enter the IP address, subnet mask, default gateway, and DNS servers to use.
- ▶ Select a region type to use for languages.
- ▶ Select a time zone.
- ▶ Select the initial or upgrade installation type.
- ▶ Prompt for a hard disk to use for root partition.
- ▶ Prompt for a automatic or manual disk layout for the file systems
- ▶ Prompt for the software bundle or packages to install. Refer to 3.1.3, “System bundle options” on page 26 for a high-level view of the software selections available.

In Linux, there might be other options that will allow you to set up sound card devices, graphical console resolution settings, and so on. You can proceed with setting these if you want, but most of the default options will be sufficient enough to start with. If not, they can be reconfigured at any time after the installation is done.

After you selected these options, the installation continues by itself to complete setting up the partitions, building the file systems, and then installing the software packages. As the installation process continues, and depending on what software packages have been selected, it might prompt for other CDs or DVDs to be inserted when required.

3.1.1 Graphical or text installation

What is not mentioned in much detail in the previous section is the console device being used. As with Solaris on some SPARC platforms, you have the choice of using a graphical console to do the installation. And even if you are on a graphical console, you also have the choice of selecting a text-based installation.

In Solaris, when installing from a CD set, the *Installation CD* will always start a graphical installation and the *CD 1* will start a text-based installation. On the DVD, you will be prompted for which type of installation to use.

In Linux for IA32, the installation boot CD will try to start the graphical installation, and it will fail back to text mode installation if necessary. You can also specify when you boot off the CD or DVD what type of console to use for the installation. If you have a graphical adapter, but for some reason want a text mode installation, you can specify so before booting the kernel and installation program. In addition, you can choose a different screen resolution if the default one is not working on your monitor.

Note: Depending on the target platform or system, you might not have a graphical console at all. See Part 3, “IBM eServer platforms” on page 235.

3.1.2 Other installation methods

What we have discussed in the previous section is the installation of the software packages from a CD or DVD drive. Solaris and Linux both offer other ways to source the packages for installation.

In Solaris, you have the following additional choices:

- ▶ Custom Jumpstart
- ▶ Web Start Flash
- ▶ Live Upgrade
- ▶ Factory Jumpstart

In Linux, you have the following additional choices:

- ▶ Another local hard drive partition (can be USB)
- ▶ NFS server
- ▶ FTP server
- ▶ HTTP server
- ▶ SMB or TFTP servers (only for SUSE)
- ▶ A customized Kickstart or AutoYaST installation from any of the available sources of installation

For specific information about how to accomplish each of these methods, refer to the detailed installation documentation provided by the vendors.

3.1.3 System bundle options

If you use an installation method that has not been preconfigured to select the software packages for you, the installation program prompts you for what you want to install.

In Table 3-1 on page 27, we outline the different selections available for Solaris and Linux.

The most common choices for server platforms are “Entire Distribution” for Solaris and “Server Install” for Linux.

What comes with each specific collection depends on how the distributor of your package determines what needs to be included. However, most of the common packages for each type of collection are included. But if they are not, you can easily install them later. Refer to Chapter 5, “Software management” on page 75 for information about how to install packages.

Tip: If you are not yet comfortable with software management in Linux and you want to save some time, you can choose the “Full Install” software bundle to install everything on the Linux server as disk space permits.

Table 3-1 Logical groupings of software clusters provided by installation

Solaris	Red Hat	SUSE
<ul style="list-style-type: none"> ▶ Core: Required operating system files ▶ End-user system support: Core plus window environment ▶ Developer system support: End user plus the development environment ▶ Entire Distribution: Developer system plus enhanced features ▶ Entire Distribution Plus OEM: Entire distribution plus third-party hardware drivers (on SPARC only) 	<ul style="list-style-type: none"> ▶ Workstation install (stand-alone client) ▶ Developer install (workstation install, plus developer tools, compiler) ▶ Server install (server services and software) ▶ Full Install (all software/services) ▶ Minimal Install (minimum packages for a Linux system) ▶ Custom (choose packages manually) 	<ul style="list-style-type: none"> ▶ Workstation install (stand-alone client) ▶ Developer install (includes developer tools, compiler) ▶ Server install (server services and software) ▶ Full Install (all software/services) ▶ Custom (choose packages manually)

3.1.4 Linux LVM consideration

In Solaris, the installer only works with a raw device in a UFS file system. It neither creates a SVM volume nor does it try to create any type of RAID structure. You are expected to do this step after the OS has been installed completely.

In Linux, during the installation, you have the option of using Logical Volume Manager (LVM) and RAID when you are defining the hard disk for the root drive. In addition, you can select which hard drives to use.

Important: When you use LVM in Linux, be aware that /boot should not be an LVM structure, but a raw disk partition.

The /root and other file systems can be part of an LVM setup. The reason behind this is that the boot loader does not recognize an LVM structure; therefore, it cannot read anything out of an LVM partition.

Refer to 4.10, “Solaris Volume Manager to Linux LVM” on page 65.

Note: There might be other disk layout limitations depending on the target platform. Check the platform-specific chapters for more information about these restrictions. See Part 3, “IBM eServer platforms” on page 235.

3.1.5 Linux firewall

Linux comes with a firewall based on the **iptables** kernel modules as part of the installation packages, and it is enabled by default if you do not specify otherwise. For detailed information, consult 14.13, “Firewalls” on page 202. We cover the basics here because this question is handled as a part of a Linux installation.

Red Hat Enterprise Linux

During the interactive installation, you have the opportunity to select what services to permit, such as remote login (SSH), Web server (HTTP, HTTPS), file transfer (FTP), mail server (SMTP), and your own protocol/port numbers. Or, you might choose to disable the firewall completely.

Additionally, Red Hat also has SELinux activated by default. SELinux stands for security-enhanced Linux. It allows more granular control of users, programs, and processes. For more information about SELinux, refer to *Red Hat SELinux Guide* or the NSA SELinux information page at the following locations:

<http://www.redhat.com/docs/manuals/enterprise/RHEL-4-Manual/selinux-guide/>
<http://www.nsa.gov/selinux>

If you need to reconfigure or disable the SELinux or the firewall, you can use the **system-config-securitylevel** command. This opens a UI tool that can be used to set up security options. Another way to disable SELinux without using the UI tool is to edit the `/etc/sysconfig/selinux` file and change the line from “SELINUX=enforcing” to “SELINUX=disabled”. This permanently disables SELinux at the next reboot. For the firewall, you can use the **chkconfig** command on the **iptables** service.

SUSE Linux Enterprise Server

The SUSE firewall is installed and enabled by default after the installation. Its configuration file is called `/etc/sysconfig/SuSEfirewall2`. There are three startup script files that set up the firewall software:

- ▶ SuSEfirewall2_init
- ▶ SuSEfirewall2_setup
- ▶ SuSEfirewall2_final

All three files reside in `/etc/init.d` and have a symlink to the appropriate run-level subdirectory in which it is started. You can use the **chkconfig** command or **yast2 firewall** GUI to check the status of the firewall or to disable it.

3.2 Advanced installation and automation

Automating the installation can become necessary in a large enterprise environment. For managing multiple systems, automated installation over a network is often the best solution.

Where Solaris has custom Jumpstart, Red Hat has Kickstart, and SUSE has AutoYaST.

There are no equivalent tools in Linux for the Solaris Web Start Flash and Live Upgrade.

3.2.1 Installing using a serial interface as input/output device

In Solaris, the installation program will use the current console as an I/O device, and if you are using a serial interface, it will use that same serial interface.

In Linux for IA32 (see Part 3, “IBM eServer platforms” on page 235 for other IBM platforms), the default console device used for installation is the graphics card. If you need to install the system using a serial interface, you need to specifically tell the installation program to do so by adding the `nofb console=ttyS0` kernel boot options. For example, if you are installing from a CD/DVD, you need to launch the installation from the graphical terminal boot prompt using `linux nofb console=ttyS0`. You then continue with the installation through the serial interface.

If you want to automatically start the installation over the serial interface, one of the options is to remaster the installation CD/DVD in order to modify the `isolinux.cfg` boot configuration file from the media.

For RHEL, edit the `isolinux/isolinux.cfg` file from the CD/DVD and modify the first line from `default linux` to `default linux nofb console=ttyS0`, and erase the `timeout 600` option.

For SUSE, edit the `boot/loader/isolinux.cfg` file from the CD/DVD and modify the first line from `default harddisk` to `default linux nofb console=ttyS0`, and erase the `timeout 200` option.

We do not describe here the CD/DVD remastering procedure because it is out of the scope of this book. If you are using the serial interface from a remote location, you probably do not want to change the CD or DVD, so the best way to remotely install with a serial console is to use a network installation and a serial console. For this, you need to set up the network boot and install servers as usual and add the `nofb console=ttyS0` options to the kernel line served to clients.

3.2.2 Configuring a serial console device

The serial interface can be configured to act as the serial console in a similar way as on Sun SPARC systems. However, there might be a limitation based on the platform that you are using. Most IA32-based systems do not have the BIOS capability to have the serial interface activated from the point of power-on self-test (POST) to loading the boot loader. You will not be able to access the system up to this point. However, some enterprise servers have a separate management card that can be accessed through a serial or network and can be used to manage that server even if it is not powered on (refer to Part 3, “IBM eServer platforms” on page 235). This is equivalent with the Sun LOM feature.

The GRUB¹ IA32-based boot loader can be configured to be controlled from a serial line, and this is useful if you need to interactively manipulate the booting kernel, the kernel boot parameters, or both. If you are remotely controlling your server through a serial console, it is a good practice to set GRUB to listen to that serial console.

To configure the GRUB boot loader to use the serial console, you need to add the parameters shown in the Example 3-1 at the beginning of its `/boot/grub/menu.lst` configuration file and comment out the `splashimage` or `gfxmenu` options.

Example 3-1 Entries required in `/boot/grub/menu.lst` file

```
serial --unit=0 --speed=9600 --word=8 --parity=no --stop=1
terminal --dumb --timeout=10 serial console
# splashimage OPTIONS (in RHEL)
# gfxmenu OPTIONS (in SLES)
```

In this example, you might need to adjust the serial parameters to match your actual hardware setup. If you have a VT100 compatible terminal, you can remove the `--dumb` option from the terminal line. For more information about those parameters, consult the GRUB documentation.

Important: If you do not comment out or erase the `splashmenu` or `gfxmenu` options from the GRUB's `/boot/grub/menu.lst` configuration file, it initializes the graphical mode and does not work on your serial as expected.

For setting up the serial console for the rest of the Linux system, you have two options:

- ▶ For a temporary solution, at the GRUB boot prompt, edit the boot command line to have `console=ttyS0,9600` for the first serial interface at speed of 9600 bps.

¹ The GRand Unified Bootloader; see <http://www.gnu.org/software/grub>

- ▶ For permanent solution, edit the `/boot/grub/menu.lst` file and add the previous option to the kernel command line.

If you want the console messages to be shown on both the graphical console and on the serial line, you can add the `console=ttyS0 console=tty0` option.

And if you need a login prompt on that serial device, you need to add to the `/etc/inittab` a line similar to the one shown in Example 3-2.

Example 3-2 Enabling a login prompt on the serial line

```
S0:2345:respawn:/bin/agetty -L 9600 ttyS0 vt100
```

Direct root login can be enabled on this serial interface by adding a `ttyS0` to the end of the `/etc/securetty` file.

For more information about configuring console logins over serial ports, refer to 6.3, “Terminals and modems” on page 101.

Some enterprise-level IA32 platforms come with the serial console activated from the POST. Refer to Part 3, “IBM eServer platforms” on page 235 for more information.

3.2.3 Installing using remote display

In Solaris, you can install using the local graphic display or the serial interface. In Linux, you can install using the local graphic display, serial interface, and a few other remote modes.

In RHEL, you have the following options:

- ▶ Add a `display=IP_OF_REMOTE_X:0` argument to the kernel boot line for using a remote X Server for displaying the installation GUI.
- ▶ Add a `vnc vncpassword=MIN_6_CHAR_PASS` argument to the kernel boot line for using a VNC client from another machine to connect to the installation GUI.

See “Additional RHEL installation boot options” on page 280 for more RHEL installation options.

In SLES, you have the following options:

- ▶ Add a `usesh=1 sshpassword=MIN_1_CHAR_PASS` argument to the kernel boot line for starting a `ssh` installation. If you connect with `ssh -X`, you can have graphical GUI.
- ▶ Add a `vnc=1 vncpassword=MIN_5_CHAR_PASS` argument to the kernel boot line for using a `vncclient` from another machine to connect to the installation GUI.

See “Additional SLES install boot options” on page 282 for more SLES installation options.

3.2.4 Solaris Jumpstart

The Jumpstart services enables you to set up a configuration file through the rules file and build customization for each type, class, or group of servers, depending on how you classify your servers in your environment.

Jumpstart has two basic components:

- ▶ A boot server
- ▶ An install server

These components can either reside in the same server or on different servers. It all depends on your network.

Normally, the boot server component of Jumpstart has to be in the same subnet as the client server being installed, but the install server can be anywhere in the network. In an enterprise environment, this might be a bit of an issue. Therefore, new functionalities were built into the Jumpstart procedure that allows the client server to boot from a server outside of its subnet.

Jumpstart can now be configured to rely on the DHCP server to provide some boot parameters to the client server. If you have not had the chance to use this functionality, refer to the “Supporting Solaris Network Installation with the DHCP Service (Task Map)” section in the Solaris *Installation Guide* and the *System Administration Guide: IP Services*.

On the client server, you can type the **boot net - install dhcp** command to tell the boot loader to enable the network and configure its IP settings from the DHCP, and that the DHCP will provide additional parameters that it will need to start the installation procedure.

3.2.5 Red Hat Kickstart

Kickstart is an installation configuration file used by the **anaconda** installation program. There is a wealth of information about **anaconda** on the Web. Red Hat *System Administration Guide* has a very detailed how-to as well. To access the Red Hat documentation, refer to:

<http://www.redhat.com/docs/manuals/enterprise/>

The following list describes the basic actions Kickstart:

- ▶ It uses a configuration file for a specific server or a group of servers. See Example 3-3 on page 34 and Example 3-4 on page 35 for sample Kickstart files.
- ▶ This configuration file can be supplied in a bootable CD-ROM, bootable diskette, on a network through NFS, FTP, and HTTP, or from a USB flash drive.
- ▶ If you want to install from the network, you can either boot from the network if you have a PXE-enabled system and a network boot server configured, or you can boot from local media such CD, DVD, or USB flash and install the rest of the software from the network.
- ▶ You can use Kickstart for initial installs and upgrades.
- ▶ You can specify the Kickstart file location at the install GRUB prompt with the **ks=URL** option, or you can use DHCP to supply a Kickstart URL for your client with the **ks** option.

The following sample command line invokes the installation using Kickstart from an NFS server. This line is entered at the GRUB prompt:

```
linux ks=nfs://IP_ADDRESS_OF_NFS_SERVER/NFS_PATH/sample-ks.cfg
```

IP_ADDRESS_OF_NFS_SERVER is the IP address of the server that has the */NFS_PATH* directory exported through NFS. That subdirectory contains the *sample-ks.cfg* file. There are additional steps required to make the network installation work (for example, copying the contents of the CDs or the ISO CD image files to source hard drive). For these instructions, see *Red Hat Enterprise Linux 4 Installation Guide for x86, Itanium[™], AMD64, and Intel[®] Extended Memory 64 Technology (Intel[®] EM64T)* in the chapters “Installing Red Hat Enterprise Linux” and “Performing a Network Installation.”

The Kickstart configuration file has many options. If used properly, you can do a fully automated, hands-free installation. If you happen to miss an item that the installation needs, it stops the installation at that portion and prompts you to enter what it needs. Items such as disk partitioning, LVM and RAID, software selection, and others can all be defined in the configuration file. You can generate a Kickstart file by using the Kickstart Configurator command **system-config-kickstart**. This is the graphical-based front end. Or, you can have it automatically generate a Kickstart file for you based on an existing server by issuing the following command:

```
# system-config-kickstart --generate <output-file-name>
```

When you use the Kickstart Configurator, you set up the following sections:

- ▶ Basic configuration information: Language support, keyboard, mouse, time zone, assign a root password, and so on.
- ▶ Installation method: New installation or upgrade, media source (CD, NFS, FTP, HTTP, or hard drive).
- ▶ Boot loader options: Install a new boot loader or not, set up the GRUB password, install MBR, and assign kernel parameters.
- ▶ Partition information: Set up disk partitioning information, use RAID, LVM, or both.
- ▶ Network configuration: Set up network device, use DHCP, static IP, or BOOTP.
- ▶ Authentication: Set up what method to use for user authentication, shadow file, NIS, LDAP, Kerberos 5, and so on.
- ▶ Firewall configuration: Enable or disable the firewall, set up trusted devices and services.
- ▶ Display configuration: Configure X Window System, set up the video card and monitor, set up the display resolution.
- ▶ Package selection: Select the groups of packages to install (can only select from groups and not individual packages).
- ▶ Preinstallation and post-installation script: Set up customized scripts to run before or after the installation process.

There are differences between doing a manual Kickstart selection and a system-generated Kickstart file. See the two sample files in Example 3-3 and Example 3-4 on page 35.

Example 3-3 Sample manual Kickstart configuration file

```
#Generated by Kickstart Configurator
#platform=x86, AMD64, or Intel EM64T

#System language
lang en_US
#Language modules to install
langsupport en_US
#System keyboard
keyboard us
#System mouse
mouse
#System timezone
timezone America/Edmonton
#Root password
rootpw --iscrypted $1$IRIk0Ucb$gYVbwzLZ62TsYX8IYEN.60
```



```

#Reboot after installation
reboot
#Use text mode install
text
#Use interactive kickstart installation method
interactive
#Install OS instead of upgrade
install
#Use NFS installation Media
nfs --server=9.3.5.11 --dir=/vm/rhel
#System bootloader configuration
bootloader --location=mbr
#Clear the Master Boot Record
zerombr yes
#Partition clearing information
clearpart --all --initlabel
#Disk partitioning information
part / --fstype ext3 --size 3000
part swap --size 512
#System authorization information
auth --useshadow --enablemd5
#Network information
network --bootproto=dhcp --device=eth0
#Firewall configuration
firewall --disabled
#XWindows configuration information
xconfig --depth=8 --resolution=1024x768 --defaultdesktop=GNOME
#Package install information
%packages --resolvedeps
#@ base-x
#@ kde-desktop
#@ editors
#@ graphical-internet
#@ server-cfg
#@ admin-tools
#@ system-tools
#@ printing
#@ compat-arch-support

```

Example 3-4 Sample system-generated Kickstart file from an existing server

```

#Generated by Kickstart Configurator
#platform=x86, AMD64, or Intel EM64T

#System language
lang en_US
#Language modules to install
langsupport en_US en --default=en_US

```

```

#System keyboard
keyboard us
#System mouse
mouse
#Sytem timezone
timezone America/North_Dakota/Center
#Root password
rootpw --iscrypted $1$74Wx1sHg$WHkNT3cVQjiNQA2u1.HrG.
#Install OS instead of upgrade
install
#Use CDROM installation media
cdrom
#Clear the Master Boot Record
zerombr yes
#Partition clearing information
clearpart --linux
#Package install information
%packages resolvedeps
4Suite
Canna
Canna-devel
Canna-libs
ElectricFence
FreeWnn
FreeWnn-devel
FreeWnn-libs
GConf2
GConf2-devel
.
.
.(long list of packages cutout here)
.
.
xorg-x11-xfs
xpdf
xrestop
xsane
xsane-gimp
xscreensaver
xsri
xterm
yelp
yp-tools
ypbind
ypserv
zip
zisofs-tools
zlib
zlib-devel

```

zsh
zsh-htm1

A manually generated file enables you to customize the information. As for a system-generated one, it picks up only a few of the components. The following components are not gathered by the system-generated Kickstart file. The installation is defaulted to CD-ROM. You might want to change this if doing a network installation.

- ▶ Disk partitioning layout
- ▶ Boot loader setting
- ▶ Authentication method setting
- ▶ Network settings
- ▶ Firewall settings
- ▶ X Window System settings, if required

Depending on what you want to achieve, you can combine the two methods to generate a more complete Kickstart file. If your target server will have the same basic software and package configuration as the current server, you can have the system generate a configuration file and have Kickstart open that file and you can customize all the missing components mentioned earlier.

Refer to the Red Hat *System Administration Guide* for more information about the Kickstart installation.

3.2.6 SUSE AutoYaST

AutoYaST uses the same concepts as the Red Hat Kickstart. The following list describes some of its capabilities:

- ▶ The AutoYaST configuration file, called a control file, automatically controls either certain portions or the whole portion of the installation process.
- ▶ Installation source can be from CD/DVD, HDD, NFS, SMB, FTP, HTTP, and TFTP.
- ▶ AutoYaST can use a network PXE boot and DHCP to provide network installation information.
- ▶ AutoYaST is for initial installs and not for upgrades.
- ▶ You can specify at the install GRUB prompt the AutoYaST file location with the `autoyast=URL` option.
- ▶ AutoYaST can import Red Hat Kickstart files.

A sample command line to invoke the installation using AutoYaST from an NFS server follows. Enter this line at the GRUB prompt:

```
install=nfs://IP_ADDRESS_OF_NFS_SERVER/NFS_PATH/  
autoyast=nfs://IP_ADDRESS_OF_NFS_SERVER/NFS_PATH/sample-autoyast.xml
```

Enter these two lines as one line. The *IP_ADDRESS_OF_NFS_SERVER* is the IP address of the server that has exported through NFS the */NFS_PATH* directory. That subdirectory contains also the *sample-autoyast.xml* file. There are additional steps required to for the network installation, such as copying the contents of the CDs to the hard drive. You can use the **yast2 instserver** program to copy the install media to the server and configure the relevant services such as SLP, NFS, HTTP, and FTP. Refer to the *SUSE Linux AutoYaST Automatic Linux Installation* documentation for more information.

The AutoYaST control file enables very granular control of the installation process. If used properly, you can do a fully automated, hands-free installation. If you happen to miss an item that the installation needs, it stops the installation at that portion and prompts you to enter what it needs. Items, such as disk partitioning, LVM and RAID, software selection, and others, can all be defined in the control file. You can generate an AutoYaST control file by using the YaST command **yast2 autoyast** (graphical-based) or **yast autoyast** (text-based). Unlike Red Hat Kickstart, AutoYaST does not provide a way to automatically generate a control file based on a currently running server.

The control file enables you to have much more granular control of system options, that is, up to the */etc/sysconfig* settings, as well as the settings for the installation itself. The following list of options are available to you:

- ▶ Software setup:
 - Online update (if enabled):
 - What time of day
 - If download patches only or whole packages
 - Software packages: Which group of packages to install. After you have selected your choice from the following list, you also have the option to do a detailed selection of packages:
 - Minimum system
 - Minimum graphical system (without KDE)
 - Full installation
 - Default system
- ▶ Hardware setup:
 - Partitioning for hard drives, use RAID, LVM, or both.

- Printer configuration: Use direct connects, CUPS, LPD-style, and so on.
- Sound card configuration.
- Graphics card and monitor configuration: Enable X Window System, 3D support, color depth, display resolution, and so on.
- ▶ System setup:
 - Boot loader configuration: Set up GRUB, location, and sections.
 - General options: Language support, time zone, hardware clock, keyboard, mouse, and so on.
 - Report and logging options: Enable or disable logging of messages, warnings, and errors.
 - Restore module: Restore files from an archive device or location as part of the installation process.
 - Runlevel editor: Configure the default run level and what services to enable or disable for each run level.
 - /etc/sysconfig editor: Preconfigure kernel values.
- ▶ Network device setup: Set up the type of network device and whether to use DHCP or static IP, set up host name and name server, routing information, and others.
- ▶ Network service setup: Configure DHCP server, DNS server, host names, HTTP server, LDAP server/client, NFS server/client, NIS server/client, Samba server/client, and others.
- ▶ Security and users setup: Configure CAs, certificates, firewalls, VPN, security settings, and create and edit users.
- ▶ Miscellaneous setup: Configure or preload customized application configuration files, set up custom preinstallation and post-installation scripts.

When doing the partitioning, note that the default size it uses when you allocate a size for a partition is in kilobytes (KB). You can also enter mb, MB, gb, or GB to the end of the partition size.

Example 3-5 shows a sample AutoYaST file.

Example 3-5 Sample AutoYaST file

```
<?xml version="1.0"?>
<!DOCTYPE profile SYSTEM "/usr/share/autoinstall/dtd/profile.dtd">
<profile xmlns="http://www.suse.com/1.0/yast2ns"
xmlns:config="http://www.suse.com/1.0/configns">
  <configure>
    <networking>
```

```

<dns>
  <dhcp_hostname config:type="boolean">>false</dhcp_hostname>
  <dhcp_resolv config:type="boolean">>false</dhcp_resolv>
  <domain>labtest.com</domain>
  <hostname>sles9serv</hostname>
  <nameservers config:type="list">
    <nameserver>9.3.5.2</nameserver>
    <nameserver>9.3.5.254</nameserver>
  </nameservers>
  <searchlist config:type="list">
    <search>labtest.com</search>
  </searchlist>
</dns>
<interfaces config:type="list">
  <interface>
    <bootproto>static</bootproto>
    <broadcast>9.3.5.255</broadcast>
    <device>eth0</device>
    <ipaddr>9.3.5.7</ipaddr>
    <netmask>255.255.255.0</netmask>
    <network>9.3.5.0</network>
    <startmode>onboot</startmode>
  </interface>
</interfaces>
<modules config:type="list">
  <module_entry>
    <device>static-0</device>
    <module></module>
    <options></options>
  </module_entry>
</modules>
<routing>
  <ip_forward config:type="boolean">>false</ip_forward>
  <routes config:type="list">
    <route>
      <destination>default</destination>
      <device>-</device>
      <gateway>9.3.5.1</gateway>
      <netmask>-</netmask>
    </route>
  </routes>
</routing>
</networking>
<printer>
  <cups_installation config:type="symbol">server</cups_installation>
  <default>local_lp0</default>
  <printcap config:type="list">
    <printcap_entry>
      <ff config:type="boolean">>false</ff>
    </printcap_entry>
  </printcap>
</printer>

```

```

<info>Local parallel printer</info>
<location>Local direct connection to LPT1</location>
<manufacturer>HP</manufacturer>
<model>HP LaserJet 5</model>
<name>local_lp0</name>
<nick>HP LaserJet 5 Foomatic/hpijs (recommended)</nick>
<options config:type="list"/>
<ppd_options config:type="list">
  <ppd_option>
    <key>Copies</key>
    <value>1</value>
  </ppd_option>
  <ppd_option>
    <key>Duplex</key>
    <value>None</value>
  </ppd_option>
  <ppd_option>
    <key>Economode</key>
    <value>Off</value>
  </ppd_option>
  <ppd_option>
    <key>InputSlot</key>
    <value>Default</value>
  </ppd_option>
  <ppd_option>
    <key>MPTray</key>
    <value>First</value>
  </ppd_option>
  <ppd_option>
    <key>PageSize</key>
    <value>A4</value>
  </ppd_option>
  <ppd_option>
    <key>PrintoutMode</key>
    <value>Normal</value>
  </ppd_option>
  <ppd_option>
    <key>Quality</key>
    <value>FromPrintoutMode</value>
  </ppd_option>
  <ppd_option>
    <key>REt</key>
    <value>Medium</value>
  </ppd_option>
  <ppd_option>
    <key>TonerDensity</key>
    <value>5</value>
  </ppd_option>
</ppd_options>

```

```

        <raw config:type="boolean">>false</raw>
        <uri>parallel:/dev/lp0</uri>
    </printcap_entry>
</printcap>
<server_hostname></server_hostname>
<spooler>cups</spooler>
</printer>
<security>
    <console_shutdown>reboot</console_shutdown>
    <cracklib_dict_path>/usr/lib/cracklib_dict</cracklib_dict_path>
    <cwd_in_root_path>yes</cwd_in_root_path>
    <cwd_in_user_path>yes</cwd_in_user_path>
    <displaymanager_remote_access>no</displaymanager_remote_access>
    <enable_sysrq>no</enable_sysrq>
    <fail_delay>3</fail_delay>
    <faillog_enab>yes</faillog_enab>
    <gid_max>60000</gid_max>
    <gid_min>1000</gid_min>
    <kdm_shutdown>all</kdm_shutdown>
    <lastlog_enab>yes</lastlog_enab>
    <obscure_checks_enab>yes</obscure_checks_enab>
    <pass_max_days>99999</pass_max_days>
    <pass_max_len>7</pass_max_len>
    <pass_min_days>0</pass_min_days>
    <pass_min_len>5</pass_min_len>
    <pass_warn_age>7</pass_warn_age>
    <passwd_encryption>des</passwd_encryption>
    <passwd_use_cracklib>yes</passwd_use_cracklib>
    <permission_security>secure</permission_security>
    <run_updatedb_as>nobody</run_updatedb_as>
    <system_gid_max>499</system_gid_max>
    <system_gid_min>100</system_gid_min>
    <system_uid_max>499</system_uid_max>
    <system_uid_min>100</system_uid_min>
    <uid_max>60000</uid_max>
    <uid_min>500</uid_min>
    <useradd_cmd>/usr/sbin/useradd.local</useradd_cmd>
    <userdel_postcmd>/usr/sbin/userdel-post.local</userdel_postcmd>
    <userdel_precmd>/usr/sbin/userdel-pre.local</userdel_precmd>
</security>
<x11>
    <color_depth config:type="integer">16</color_depth>
    <configure_x11 config:type="boolean">>true</configure_x11>
    <display_manager>kdm</display_manager>
    <enable_3d config:type="boolean">>false</enable_3d>
    <monitor>
        <display>
            <frequency config:type="integer">60</frequency>
            <height config:type="integer">768</height>

```



```

        <width config:type="integer">1024</width>
    </display>
    <monitor_device>1024X768@60HZ</monitor_device>
    <monitor_vendor> LCD</monitor_vendor>
</monitor>
<resolution>1024x768</resolution>
<window_manager>kde</window_manager>
</x11>
</configure>
<install>
  <bootloader>
    <activate config:type="boolean">>true</activate>
    <global config:type="list">
      <global_entry>
        <key>color</key>
        <value>white/blue black/light-gray</value>
      </global_entry>
      <global_entry>
        <key>default</key>
        <value config:type="integer">0</value>
      </global_entry>
      <global_entry>
        <key>timeout</key>
        <value config:type="integer">8</value>
      </global_entry>
      <global_entry>
        <key>gfxmenu</key>
        <value>/boot/message</value>
      </global_entry>
    </global>
    <loader_device>/dev/sda</loader_device>
    <loader_type>grub</loader_type>
    <location>mbr</location>
    <repl_mbr config:type="boolean">>false</repl_mbr>
    <sections config:type="list">
      <section config:type="list">
        <section_entry>
          <key>title</key>
          <value>Linux</value>
        </section_entry>
        <section_entry>
          <key>kernel</key>
          <value>/boot/vmlinuz root= vga=0x314 selinux=0 splash=silent
resume=
showopts elevator=cfq</value>
        </section_entry>
        <section_entry>
          <key>initrd</key>
          <value>/boot/initrd</value>

```

```

        </section_entry>
    </section>
    <section config:type="list">
        <section_entry>
            <key>title</key>
            <value>Hard Disk</value>
        </section_entry>
        <section_entry>
            <key>root</key>
            <value>(hd0)</value>
        </section_entry>
        <section_entry>
            <key>chainloader</key>
            <value>+1</value>
        </section_entry>
    </section>
    <section config:type="list">
        <section_entry>
            <key>title</key>
            <value>Failsafe</value>
        </section_entry>
        <section_entry>
            <key>kernel</key>
            <value>/boot/vmlinuz root= showopts ide=nodma apm=off acpi=off
vga=n
normal noresume selinux=0 barrier=off nosmp noapic maxcpus=0 3</value>
        </section_entry>
        <section_entry>
            <key>initrd</key>
            <value>/boot/initrd</value>
        </section_entry>
    </section>
</sections>
</bootloader>
<general>
    <clock>
        <hwclock>localtime</hwclock>
        <timezone>US/Central</timezone>
    </clock>
    <keyboard>
        <keymap>english-us</keymap>
    </keyboard>
    <language>en_US</language>
    <mode>
        <confirm config:type="boolean">true</confirm>
        <forceboot config:type="boolean">true</forceboot>
    </mode>
    <mouse>
        <id>probe</id>

```

```

</mouse>
</general>
<partitioning config:type="list">
  <drive>
    <device>/dev/hda</device>
    <initialize config:type="boolean">>false</initialize>
    <partitions config:type="list">
      <partition>
        <crypt>twofish256</crypt>
        <filesystem config:type="symbol">reiser</filesystem>
        <format config:type="boolean">>true</format>
        <loop_fs config:type="boolean">>false</loop_fs>
        <mount>/</mount>
        <partition_id config:type="integer">131</partition_id>
        <size>4000</size>
      </partition>
      <partition>
        <crypt>twofish256</crypt>
        <format config:type="boolean">>false</format>
        <loop_fs config:type="boolean">>false</loop_fs>
        <mount></mount>
        <partition_id config:type="integer">130</partition_id>
        <size>512</size>
      </partition>
    </partitions>
    <use>free</use>
  </drive>
  <drive>
    <device>/dev/hdb</device>
    <initialize config:type="boolean">>false</initialize>
    <partitions config:type="list">
      <partition>
        <crypt>twofish256</crypt>
        <format config:type="boolean">>false</format>
        <loop_fs config:type="boolean">>false</loop_fs>
        <mount>/var</mount>
        <partition_id config:type="integer">142</partition_id>
        <size>2048</size>
      </partition>
      <partition>
        <crypt>twofish256</crypt>
        <format config:type="boolean">>false</format>
        <loop_fs config:type="boolean">>false</loop_fs>
        <mount>/usr</mount>
        <partition_id config:type="integer">142</partition_id>
        <size>2048</size>
      </partition>
      <partition>
        <crypt>twofish256</crypt>

```

```

        <format config:type="boolean">false</format>
        <loop_fs config:type="boolean">false</loop_fs>
        <mount>/home</mount>
        <partition_id config:type="integer">142</partition_id>
        <size>3000</size>
    </partition>
</partitions>
    <use>free</use>
</drive>
</partitioning>
<software>
    <addons config:type="list">
        <addon>Base-System</addon>
        <addon>Basis-Devel</addon>
        <addon>Basis-Sound</addon>
        <addon>File-Server</addon>
        <addon>Gnome</addon>
        <addon>HA</addon>
        <addon>Kde-Desktop</addon>
        <addon>LAMP</addon>
        <addon>LSB</addon>
        <addon>Linux-Tools</addon>
        <addon>Mail_News-Server</addon>
        <addon>Print-Server</addon>
        <addon>SuSE-Documentation</addon>
        <addon>Various-Tools</addon>
        <addon>WBEM</addon>
        <addon>X11</addon>
        <addon>YaST2</addon>
        <addon>analyze</addon>
        <addon>auth</addon>
        <addon>dhcp_DNS-Server</addon>
    </addons>
    <base>Full-Installation</base>
</software>
</install>
</profile>

```

For more information about how to use AutoYaST, refer to the “Special Installation Procedure” section in the Novel SUSE *Administration Guide* and *AutoYaST Automatic Linux Installation and Configuration with YaST2*. To access the SUSE documentation, refer to:

<http://www.novell.com/documentation/sles9/>

3.2.7 The PXE protocol

PXE stands for Pre-boot Execution Environment. It is a protocol designed by Intel that allows IA32 computers to boot using the network interface. PXE is available for newer network adapters that have execution codes stored in their ROM device. When the computer boots up, the BIOS loads the PXE ROM into the memory and runs it. A menu is then presented on the display and enables the user to boot the loaded operating system through the network.

PXE relies on a DHCP or BOOTP server to provide an IP address and other network information. A TFTP server is also needed to supply the boot loader to the client. For basic functionality, you do not need a PXE server.

Linux supports PXE for network booting using the `syslinux` package. The `syslinux` package is bundled in both RHEL and SLES. For more information about this topic, read 8.5, “Network booting” on page 135 and the following section.

3.3 Network-based install: Solaris and Linux heterogeneous environments

Depending on your migration plan, you might have an environment where you will need to configure a Solaris system to serve a Linux network installation, or a Linux system to serve a Solaris network installation.

3.3.1 Solaris system serving Linux network installation

To configure a Solaris system to serve a Linux network installation, you have to enable and configure the DHCP, TFTP, and NFS services on the Solaris server for booting and installing a Linux network client. Instead of NFS, you can also use FTP or HTTP protocols for a package distribution. You can set up the Solaris server to serve standard Linux network installs and also custom *Kickstart* or *AutoYaST* installs. There are no ready-to-use tools for this task, so you need to configure all these services by hand. But if you follow the Linux vendor instructions for setting up a Linux server for network installations, you will be able to replicate that scenario for a Solaris server. We provide a little more technical how-to information in this section, because this topic is not covered by any vendor documentation.

Solaris system serving RHEL network installation

Here is a short outline of the needed steps to configure a Solaris system to serve RHEL for IA32 network installations:

1. Get the `pxelinux.0` file from PXELINUX tools at <http://syslinux.zytor.com> or from the RHEL `syslinux` package (`/usr/lib/syslinux/pxelinux.0` in RHEL) and

place it in the TFTP share directory of the Solaris system, which by default is `/tftpboot`. This file is an IA32 PXE Linux loader that is needed for the network boot process.

2. Copy the `vmlinuz` and `initrd.img` files from the `/isolinux` directory found in the CD1 of RHEL to the same `/tftpboot` folder on the Solaris server.
3. Create the `pxelinux.cfg` configuration directory in the same place you put the `pxelinux.0` file, and create a *default* file for the PXELINUX configuration. (See Example 3-6.)

Example 3-6 Sample default PXE configuration file

```
default linux
label linux
kernel vmlinuz
append initrd=initrd.img ramdisk_size=65536 ks=KS_URL
```

For more information about syntax, check the PXELINUX documentation, and look at the `isolinux.cfg` file from the same `/isolinux` directory on the CD1.

4. Edit `/etc/inetd.conf` file to enable the `tftpd` daemon if not already active, and send a HUP signal to `inetd` to reload the configuration if you modified it.
5. Configure and start the DHCP daemon in the normal way, and add the following standard DHCP options:

```
BootFile=pxelinux.0
BootSrvA=IP_address_of_your_Solaris_server
```
6. Copy the RHEL CD images into a local directory and share that directory through NFS.
7. Optionally, create a Kickstart file for the clients and place it in the NFS shared directory and reference it in the default configuration file of the `pxelinux`, for example:

```
ks=nfs:IP_OF_NFS_SERVER:/SHARE/ks.cfg
```

Now, you are ready to boot and install your Linux RHEL clients from this Solaris server.

This example is for installing a similar configuration on all clients, but you can make a custom configuration for each separate client by adding a specific MAC-based or IP-based PXE configuration file for each client and then customize it by specifying the right Kickstart file for the installation configuration. For platforms other than IA32, the process is similar, but you will need to use another boot loader instead of `pxelinux.0`.

Solaris system serving SLES network installation

Here is a short outline of the needed steps to configure a Solaris system to serve SLES for IA32 network installations:

1. Get the pxelinux.0 file from PXELINUX tools at <http://syslinux.zytor.com> or from the SLES syslinux package (/usr/share/syslinux/pxelinux.0 in SLES) and place it in the TFTP share directory of the Solaris system, which by default is /tftpboot. This file is an IA32 PXE Linux loader that is needed for the network boot process.
2. Copy the linux and initrd files from the /boot/loader directory found in the CD1 of SLES to the same /tftpboot folder on the Solaris server.
3. Create the pxelinux.cfg configuration directory in the same place you put the pxelinux.0 file, and create a *default* file for the PXELINUX configuration (see Example 3-7).

Example 3-7 Sample default PXE configuration file

```
default linux
label linux
kernel linux
append initrd=initrd ramdisk_size=65536 install=URL autoyast=URL
```

For more information about its syntax, check the PXELINUX documentation, and look at the isolinux.cfg file from the same /boot/loader directory on the CD1.

4. Edit /etc/inetd.conf to enable the **tftpd** daemon if not already active, and send a HUP signal to **inetd** to reload the configuration if you modified it.
5. Configure and start the DHCP daemon in the normal way, and add the following standard DHCP options:

```
BootFile=pxelinux.0
BootSrvA=IP_address_of_your_Solaris_server
```

6. Copy the CDs or DVD contents of the SLES distribution in a local directory, share that directory through NFS, and point the installation option to the right location, for example:

```
install=nfs://IP_OF_NFS_SERVER/NFS_SHARE/
```

7. Optionally, create a AutoYaST file for the clients and place it in a NFS shared directory and reference it in the default configuration file of the pxelinux, for example:

```
autoyast=nfs://IP_OF_NFS_SERVER/AUTOYAST_SHARE/autoyast.cfg
```

Now, you are ready to boot and install your Linux SLES clients from this Solaris server.

This example is for installing the same configuration on all clients, but you can make a custom configuration for each separate client by adding a specific MAC-based or IP-based PXE configuration file for each client and then customize it by specifying the right AutoYaST file for the installation configuration.

3.3.2 Linux system serving Solaris network installation

To configure a Linux system to serve a Solaris network installation, you have to decide which type of network initialization you will use: the default RARP+Bootparams, or the more streamlined DHCP. If you use the RARP+Bootparams services, you need to install, configure, and activate these services, because they are not normally used in Linux by default. If you choose the DHCP service, you need to configure and start this service and configure the Solaris client to ask for the DHCP service instead of RARP+Bootparams. Then, you need to configure the TFTP and NFS services for serving the kernel and the installation files. There are no ready to use tools for this task, so you need to configure all these services by hand. Our recommendation is to use the DHCP protocol instead of RARP+Bootparams, because all needed services are in the main distribution of RHEL and SLES.



Disks and file systems

This chapter provides the information required to understand how to define, configure, and set up disks and file systems for use on Linux.

This chapter describes the following topics:

- ▶ Disks and disk partitioning
- ▶ Disk-based file system management
- ▶ Virtual file systems
- ▶ Network File System (NFS)
- ▶ Swap file systems
- ▶ File system journaling
- ▶ File system organization
- ▶ AutoFSCK
- ▶ AutoFS
- ▶ Solaris Volume Manager to Linux LVM
- ▶ VERITAS VxVM and VxFS

4.1 Disks and disk partitioning

In Solaris, partitions and slices are used interchangeably to mean the same thing: raw disk slices. Linux uses the term *partitions*, but some documentation also mentions “slices” when discussing the same thing.

4.1.1 Disks

There is no distinction made in this section as to the type of attachment (that is, direct, SAN, or other), as long as the operating system can recognize that it is attached to the server.

Task table

Table 4-1 illustrates device creation commands.

Table 4-1 Device creation commands

Device creation task	Solaris	Linux
Create device files for new device	<code>devfsadm</code>	<code>udev</code>

Disk recognition on Solaris

Before a disk can be sliced for use, the disk has to be recognized by the operating system. For new SCSI devices in the system, the `sd.conf` file in the `/kernel/drv` directory has to be set up to recognize a specific target and LUN number. In addition, a reconfiguration reboot has to be completed before the device will be recognized.

If the `sd.conf` file already has the entry before the last reboot, the `devfsadm` command is run to generate the appropriate device files.

Disk recognition on Linux

Linux also requires that first the disk needs to be recognized by the operating system. This is done automatically by the kernel at the boot time if there are no hardware conflicts between disks (check for SCSI ID conflicts, or IDE master/slave conflicts). However, if *hot plug* is used to add new disks, we need to tell the kernel that a new disk is available for use. For more information about hot plug, refer to 6.1.4, “Hot-plug devices” on page 93.

In RHEL and SLES, the device files are created every time a system boots by the `udev` daemon, so a reconfiguration reboot is not necessary.

Multipath

Both Red Hat and SUSE began supporting multipathing in their latest releases of RHEL and SLES. Refer to the vendor documentation and support channels for the latest updates about how to set up and use multipathing. SUSE has an online support page describing the setup tasks for SLES9, available at:

http://portal.suse.com/sdb/en/2005/04/sles_multipathing.html

See the following link for information about multipath support in RHEL4 Update 2:

<https://rhn.redhat.com/errata/RHEA-2005-280.html>

4.1.2 Disk partitions

In Solaris, *slices* and *partitions* are used interchangeably to mean the same thing. For Linux, most references use the word *partition*. Disk partitioning is a step that needs to be done before any new disk can be used. Each slice or partition will have to be created to the appropriate size as per your requirement.

Task table

Table 4-2 illustrates disk partitioning commands.

Table 4-2 Disk partitioning commands

Disk partitioning task	Solaris	Linux
Create partitions on the disk	format or fmthard	fdisk or parted
Display partition table	prtvtoc	fdisk or parted
Maximum usable partitions	7	15 for SCSI 63 for IDE

Solaris

Use the **format** or **fmthard** commands to create the appropriate slice sizes in the disk. To display the partition table, use the **prtvtoc** command.

Linux

You can use the **fdisk** or **parted** commands to create the appropriate partition sizes in the disk. To display the partition table, use the **fdisk -l** command. Refer to the man pages for the **fdisk** and **parted** commands for more information.

Other considerations

In Solaris, you can create and use a maximum of seven slices per disk excluding the WholeDisk slice number 2.

In Linux, we have primary partitions and logical partitions contained in (primary) extended partitions.

Primary partitions and extended partitions can be created only in the Master Boot Record (MBR) of the disk and there is room for only four such partitions in MBR. Logical partitions are partitions created in extended partitions on the disk. The extended partition cannot be used directly, but only by means of the logical partitions created in it. To summarize, we can have up to four primary partitions, or one extended partition and up to three primary partitions defined in the MBR.

Using either the **fdisk** or **parted** command, you only need to create primary and logical partitions, and they then make the extended partitions automatically.

Another thing to note is that, in Linux, each partition has a partition type, which describes its intended usage. This type can be Linux, Linux LVM, Linux RAID, and so on. When creating a Linux partition, you decide what partition type to use. This depends on for what you intend to use the partition. If it is for data file system data storage, you will have to select ext2, ext3, Reiserfs, LVM, or one of the other options available. Refer to the man pages for the **fdisk** and **parted** commands for more information. The interactive **fdisk** command displays the available partitions types when you use the **l** subcommand. See Example 4-1 and Example 4-2 on page 55.

Example 4-1 Sample prtvtoc output

```
# prtvtoc /dev/rdisk/c0t1d0s2
* /dev/rdisk/c0t1d0s2 partition map
*
* Dimensions:
*   512 bytes/sector
*   424 sectors/track
*   24 tracks/cylinder
* 10176 sectors/cylinder
* 14089 cylinders
* 14087 accessible cylinders
*
* Flags:
*  1: unmountable
* 10: read-only
*
*
* Partition  Tag  Flags      First      Sector      Last
*           Tag  Flags      Sector      Count      Sector  Mount Directory
*   0       2    00    1058304  20972736  22031039  /
*   1       3    01           0   1058304   1058303
*   2       5    01           0 143349312 143349311
*   3       8    00  22031040 121308096 143339135  /space
*   7       0    00 143339136   10176 143349311
```

Example 4-2 Sample fdisk -l output

```
# fdisk -l
Disk /dev/sda: 9100 MB, 9100369920 bytes
255 heads, 63 sectors/track, 1106 cylinders
Units = cylinders of 16065 * 512 = 8225280 bytes

   Device Boot      Start         End      Blocks   Id  System
/dev/sda1                1           64     514048+   82  Linux swap
/dev/sda2    *          65        1106     8369865   83  Linux
```

4.2 Disk-based file system management

This section discusses the different methods to manage a disk-based file system.

Task table

Table 4-3 illustrates file system management commands.

Table 4-3 File system management commands

File system management task	Solaris	Linux
To create a file system on disk	newfs or mkfs	mkfs , mkfs.ext3 , mk2fs -j , parted , or mkfs.reiserfs (SUSE only)
Check file system for consistency	fsck	fsck , fsck.ext3 , e2fsck

Overview of Solaris

Solaris natively only supports the UFS file system for hard disks. Use the **newfs** or the **mkfs** command to create the file system and the **fsck** command to check and fix the file system for consistency.

The other types of disk-based file systems are HSFS (for CD-ROM), PCFS (for diskettes), and UDF (for DVD).

Bad block management uses the command **addbadsec** to map out the bad sector.

Linux

In Linux, there are several disk file systems from which to choose. Depending on its intended use, one file system might be better than another. If you do not know which to choose, a safe guideline is to use the default file system of that distribution. On Red Hat, the default disk file system is ext3, and on SUSE, it is reiserfs, and both file systems are journaled.

For creating a file system we have the following commands:

- ▶ **mkfs.ext3** or **mke2fs -j** for creating ext3 file systems
- ▶ **mkfs.reiserfs** or **mkreiserfs** for creating reiserfs file systems (SUSE only)

You can also still use the **mkfs** command, but you need to specify the file system type option with **-t *fstype***.

For checking the file system consistency, we have the following commands:

- ▶ **fsck.ext3** or **e2fsck** for ext3 file systems
- ▶ **fsck.reiserfs** or **reiserfsck** for reiserfs file systems

You can also still use the **fsck -t *fstype*** command. If you do not use the **-t *fstype*** option, the default file system will be used, ext3 for Red Hat and reiserfs for SUSE.

Linux can also access UFS partitions, if needed for migration purposes.

Table 4-4 shows the different disk file systems in Solaris and Linux.

Table 4-4 File system types

File system type	Solaris	Red Hat	SUSE
UNIX file system	ufs	Source available but not compiled	ufs (Refer to the mount command's man page for support statements.)
Reiser file system	N/A	Source available but not compiled	reiserfs
Ext3 file system	N/A	ext3	ext3
CD-ROM file system	hfs	iso9660	iso9660
DVD file system	udf	udf	udf
MS-DOS® file system	pcsf	msdos	msdos
Windows 95 and later file system	pcfs	vfat	vfat

4.3 Virtual file systems

These types of file systems (except CacheFS™) do not have a 1:1 corresponding disk slice used to store the data, but instead, use the system RAM

to store information. Anything that gets stored in here will not be persistent across a reboot.

Overview of Solaris

There are several different types and uses for virtual file systems in Solaris:

/tmp Based on the TMPFS file system. This file system is used for storing temporary files generated by the operating system and applications. When this space is filled up to a certain portion, the physical swap is used as the backing store in which to move these files. This space is also used as a buffer space for file system reads and writes in order to increase access time.

/var/run A place to store temporary system files that are currently in use, but that are not required across a reboot. This space also provides access to special kernel information and facilities.

CacheFS Used for caching slow devices on local harddrive, such as simple or double-speed CD-ROMs or NFS over slow networks.

Loopback file system

A way for you to create a virtual file system by using a different path to access the files within that file system.

/proc This file system contains all the active processes in the system by process number.

Linux

There are many options for virtual file system types in Linux. We only discuss a few of them that have a Solaris equivalent:

/proc In Linux, this contains all the information about each process, information about the running kernel, devices, networking stack, and so on. See “/proc” on page 89 for more information.

Loopback file system

An equivalent in Linux does not exist. But for similar functionality, you can investigate the **mount** command with the **--bind**, **--rbind**, and **--move** options.

/tmp	Unlike Solaris, in Linux /tmp is a disk file system and is persistent across reboots. There is no /var/run equivalent. If you need a fast file system that will not be required to be persistent across a reboot, you can create a tmpfs or a ramfs.
CacheFS	An equivalent to the Solaris CacheFS is not available in Linux.

4.4 Network File System (NFS)

Network File System (NFS) is available for both Solaris and Linux. You can use it in heterogeneous networks with both Solaris and Linux operating systems, with each of them being NFS server or client.

For more information about NFS, refer to 7.8.7, “NFS” on page 121.

4.5 Swap file systems

Swap space is handled similarly to Solaris, in that a swap space is created at the time the disk is partitioned to handle paging from physical memory.

Task table

Table 4-4 on page 56 illustrates swap commands.

Table 4-5 Swap commands

Swap task	Solaris	Linux
Create a swap space	<code>swap -a</code>	<code>mkswap</code>
Enable a swap space	<code>swap -a</code>	<code>swapon</code>
Enable all swap spaces	N/A	<code>swapon -a</code>
Disable a swap space	N/A	<code>swapoff</code>
Disable all swap spaces	N/A	<code>swapoff -a</code>
Delete a swap space	<code>swap -d</code>	N/A
Display the swap space usage	<code>swap -s</code>	<code>swapon -s</code>
Display the swap space status	<code>swap -l</code>	N/A
Set swap priority level	N/A	<code>swapon -p</code>

Overview of Solaris

swap is the only swap configuration command on Solaris. You can add, remove, and list the available swaps. Swap can be defined in dedicated disk slices, which is the preferred way, or as files within file systems, which is usually done for emergency and temporary purposes.

Linux

In Linux, we can also have swap defined in dedicated disk partitions, which is the preferred way, or as files within file systems, which is usually done for emergency and temporary purposes.

In Linux, there are several commands for managing swap space:

- ▶ **mkswap** to format the swap space in order to be used by the system
- ▶ **swapon** to activate a swap space after it was formatted with **mkswap**
- ▶ **swapoff** to deactivate a swap space that is in use

In addition, you can use the **free** command to see how much swap is used by the system.

4.6 File system journaling

File system journaling is a way to store transactions before they are written to the file system itself. After it is stored, a transaction can be applied to the file system at a later time. When a hard reboot is encountered, the operating system will only have to check and replay the journal log and will not be required to check the whole file system, thereby reducing the time it will take to perform the checks at boot up.

Overview of Solaris task

The SVM default is logging enabled for any UFS file system unless specifically disabled by itself due to available free space or specifically disabled by the system administrator.

SVM logging is all internal to the mounted file system and does not make use of any external device to keep its log entries.

Linux task

The Linux default file systems for both RHEL and SLES have logging capabilities as well. You do not need any extra steps to activate the logging with the ext3 and reiserfs file systems. Both file systems use internal journaling for storing the

transaction logs by default. It is possible to use an external device for journaling if needed.

For journal mode information for the reiserfs and ext3 file systems, see the following IBM publications:

- ▶ *Tuning Red Hat Enterprise Linux on IBM @server xSeries Servers*, REDP-3861
- ▶ *Tuning SUSE LINUX Enterprise Server on IBM @server xSeries Servers*, REDP-3862

The Linux ext3 default for journal mode is “ordered.” Meaning, all data is being forced to be written to the file system before the metadata has been committed to the journal.

There are two other modes that ext3 supports, journal and writeback:

Journal mode	Commits the metadata into the journal before any the file system is written. This is the safest mode.
Writeback mode	Data ordering is not preserved. The data can be written to the file system after the metadata has been committed to the journal. This is considered to provide better performance than the previous two modes, and still provides high system integrity, but there is a tendency for older data in files to appear after the server has crashed and journal recovered.

Novell SUSE has two additional modes, `user_xattr` and `acl`:

user_xattr	Enables extended user attributes. Refer to the attr(5) man page for more information.
acl	Enables POSIX access control lists. Refer to the acl(5) man page for more information.

SUSE Linux also provides ReiserFS. ReiserFS is designed to provide improved features for disk space utilization, access performance, and crash recovery. For more information about ReiserFS, see *SUSE Linux Enterprise Server 9 Administration and Installation Guide*.

4.7 File system organization

The Linux file system organization is standardized in the Filesystem Hierarchy Standard (FHS, <http://www.pathname.com/fhs/>) and is laid out as follows:

/bin Essential command binaries

/boot	Static files of the boot loader
/dev	Device files
/etc	Host-specific system configuration
/etc/opt	Configuration files for add-on packages
/etc/X11	Configuration for X
/home	User home directories
/lib	Essential shared libraries and kernel modules
/media	Mount point for removable media
/mnt	Mount point for mounting a file system temporarily
/opt	Add-on application software packages
/proc	Kernel and process information
/root	Root's home directory
/sbin	Essential system binaries
/srv	Data for services provided by this system
/tmp	Temporary files
/usr	Secondary hierarchy
/usr/bin	Most user commands
/usr/include	Header files included by C programs
/usr/lib	Libraries
/usr/local	Local hierarchy (empty after main installation)
/usr/sbin	Non-vital system binaries
/usr/share	Architecture-independent data
/usr/X11R6	X files
/var	Variable data
/var/account	Process accounting logs (optional)
/var/cache	Application cache data
/var/crash	System crash dumps (optional)
/var/lib	Variable state information
/var/local	Variable data for /usr/local
/var/lock	Lock files
/var/log	Log files and directories
/var/opt	Variable data for /opt

<code>/var/mail</code>	User mailbox files (optional)
<code>/var/run</code>	Data relevant to running processes
<code>/var/spool</code>	Application spool data
<code>/var/tmp</code>	Temporary files preserved between system reboots
<code>/var/yp</code>	Network Information Service (NIS) database files (optional)

4.8 AutoFSCK

In Linux, the ext2 and ext3 file system types have an option to force the checking of the consistency even if the file system was cleanly unmounted and is not marked as dirty.

This is useful when you have an ext3 journaled file system that will never be dirty, but you want to check it from time to time just to make sure that there are no inconsistencies on it. You have two variables, one that counts the number of mounts, and other that is time based. Both of them can be manipulated with the **tune2fs** program.

To disable this feature, you need to set both of these variables to 0.

There is no equivalent functionality for this task in Solaris.

4.9 AutoFS

Also known as automount or automountd, this is a way to automatically mount a Network File System (NFS) when a specific directory path is being accessed by the user. When there is no activity on these subdirectories after a certain period of time, the automountd daemon unmounts the file system.

Task table

Table 4-6 illustrates the AutoFS information.

Table 4-6 *AutoFS files*

AutoFS task	Solaris	Linux
Auto mounter daemon	automountd	automount
Master configuration file	<code>/etc/auto_master</code>	<code>/etc/auto.master</code>
Other common configuration file	<code>/etc/auto_home</code>	<code>/etc/auto.misc</code>

Solaris

The automount is configured in the `/etc/auto_master` (and its other corresponding) file and is activated by the startup script `/etc/rc2.d/S74autofs`, which runs during the server boot to run level 2 or 3. After automountd is running as a daemon, all NFS mounting and unmounting is automatically handled.

Linux

In Linux, the automount is configured in the `/etc/auto.master` file, as opposed to Solaris `/etc/auto_master` file, and is activated by the startup script `/etc/init.d/autofs`. After automountd is running as a daemon, all NFS mounting and unmounting is automatically handled.

Sample contents of configuration files

The following examples provide the sample contents of configuration files.

Example 4-3 Solaris /etc/auto_master

```
# Master map for automounter
#
+auto_master
/net          -hosts          -nosuid,nobrowse
#/home       auto_home      -nobrowse
/xfn         -xfn
```

Example 4-4 Solaris /etc/auto_home

```
# Home directory map for automounter
#
+auto_home
```

Example 4-5 Linux /etc/auto.master

```
#
# $Id: auto.master,v 1.3 2003/09/29 08:22:35 raven Exp $
#
# Sample auto.master file
# This is an automounter map and it has the following format
# key [ -mount-options-separated-by-comma ] location
# For details of the format look at autofs(5).
#/misc /etc/auto.misc --timeout=60
#/misc /etc/auto.misc
#/net /etc/auto.net
```

Example 4-6 Linux /etc/auto.misc

```
#
# $Id: auto.misc,v 1.2 2003/09/29 08:22:35 raven Exp $
#
# This is an automounter map and it has the following format
# key [ -mount-options-separated-by-comma ] location
# Details may be found in the autofs(5) manpage

cd                -fstype=iso9660,ro,nosuid,nodev :/dev/cdrom

# the following entries are samples to pique your imagination
#linux           -ro,soft,intr          ftp.example.org:/pub/linux
#boot            -fstype=ext2            :/dev/hda1
#floppy          -fstype=auto            :/dev/fd0
#floppy          -fstype=ext2            :/dev/fd0
#e2floppy        -fstype=ext2            :/dev/fd0
#jaz             -fstype=ext2            :/dev/sdc1
#removable       -fstype=ext2            :/dev/hdd
```

Example 4-7 Linux /etc/auto.net

```
#!/bin/sh

# $Id: auto.net,v 1.5 2003/09/29 08:22:35 raven Exp $

# Look at what a host is exporting to determine what we can mount.
# This is very simple, but it appears to work surprisingly well

key="$1"

# add "nosymlink" here if you want to suppress symlinking local filesystems
# add "nonstrict" to make it OK for some filesystems to not mount
opts="-fstype=nfs,hard,intr,nodev,nosuid"

# Showmount comes in a number of names and varieties. "showmount" is
# typically an older version which accepts the '--no-headers' flag
# but ignores it. "kshowmount" is the newer version installed with knfsd,
# which both accepts and acts on the '--no-headers' flag.
#SHOWMOUNT="kshowmount --no-headers -e $key"
#SHOWMOUNT="showmount -e $key | tail +2"

# Newer distributions get this right
SHOWMOUNT="/usr/sbin/showmount --no-headers -e $key"
$SHOWMOUNT | sort +0 | \
    awk -v key="$key" -v opts="$opts" -- '
    BEGIN          { ORS=""; first=1 }
```

```

$1, key ":" $1 }
END
{ if (first) { print opts; first=0 }; print " \\n\t"
{ if (!first) print "\n"; else exit 1 }

```

4.10 Solaris Volume Manager to Linux LVM

The Sun Solstice™ DiskSuite™ (SDS) has been changed to Solaris Volume Manager (SVM) since the introduction of Solaris 9. The functionality and concept is still very much the same, but SVM introduces the new feature called soft partitions.

All of the following references to SVM refer to both SDS and SVM. Where appropriate, a more detailed distinction between SDS and SVM is noted.

Task table

Table 4-7 illustrates volume management commands.

Table 4-7 Volume management commands

Volume management task	Solaris	Linux
Initialize a disk for VM	Create partition area/size metainit	Create partition area/size pvcreate mkraid, mdadm
Create a volume or volume group (VG)	metainit volumename raidtype devices...	lvcreate LVname vgcreate VGname devicename mkraid, mdadm
Enable the volume or volume group	N/A	lvchange -a y LVname vgchange -a y VGname raidstart
Disable the volume or volume group	N/A	lvchange -a n LVname vgchange -a n VGname raidstop, mdadm
Delete the volume or volume group	metaclear	lvremove LVname vgremove VGname
Add a device to the volume or volume group	metattach or metainit	lvextend LVname vgextend VGname newdevname
Delete a device from the volume or volume group	metadetach	lvreduce LVname vgreduce VGname devicename raidreconf

Volume management task	Solaris	Linux
Create a soft partition or logical volume (no RAID)	metainit -p	lvcreate -Lsize -nLVname VGname
Create a soft partition or logical volume (RAID 0)	metainit with RAID 0 on devices first, then metainit -p to create the soft partition volume	lvcreate -iNumOfStripes -IStripeSize -nLVname VGname mdadm, mkraid
Create a soft partition or logical volume on a specific device	Same as above, but the second metainit -p has the device name at the end of the command line	Same as above, but add the device name to the end of the command line
Delete a soft partition or logical volume	metaclear	lvremove /dev/VGname/LVname raidreconf
Extend a volume or logical volume	metattach volume (size or device name)	lvextend -Lsize /dev/VGname/LVname raidreconf
Extend a file system after volume has been grown	growfs	<ul style="list-style-type: none"> ▶ Red Hat ext2/ext3: resize2fs ▶ SUSE ext2: resize2fs ▶ Red Hat reiser: resize_reiserfs <p>Note: View the man pages for unmount requirement, and check if online version is available.</p>
Reduce a volume or logical volume	metadetach Volname devicename	<ul style="list-style-type: none"> ▶ ext2: resize2fs, lvreduce ▶ reiser: resize_reiserfs, lvreduce <p>Note: View the man pages for unmount requirements.</p>

The RAID and LVM2 commands installed on your system will be in either the /sbin or /usr/sbin directory. The packages that provide logical volume management are mdadm*, raidtools*, and lvm2*.

Table 4-8 on page 67 presents a quick review of the equivalent commands.

Table 4-8 Equivalent volume management commands

Volume management task	Solaris	Linux
Set up or display metadb	metadb	vgdisplay, lvdisplay, lsraid
Display metadevice status	metastat	vgdisplay, lvdisplay cat /proc/mdstat
Initialize raw devices to metadevices	metainit	pvcreate
Attach metadevices	metattach	vgchange, lvchange
Detach metadevices	metadetach	vgchange, lvchange
Clear and remove metadevices	metaclear	vgremove, lvremove
Replace a metadevice	metareplace	raidreconf
Rename a volume	metarename	raidreconf
Check metadevice ID configuration	metadevadm	mdadm
Manage hot spares	metahs	mdadm, raidhotadd, raidhotremove
Set submirrors offline	metaoffline	mdadm
Set submirrors online	metaonline	mdadm
Change volume parameters	metaparam	vgchange, lvchange
Back up volume group metadata		vgcfgbackup
Recover soft partition configuration information	metarecover	vgcfgrestore
Set up root file system for mirroring	metaroot	N/A
Administer disk sets for sharing	metaset	N/A
Resynchronize volume during reboot	metasync	N/A
Expand a file system size	growfs	vgextend, lvextend, resize2fs, resize_reiserfs, raidreconf
Has entries that starts up the kernel metadevice modules	/etc/system	N/A
Sets the default number of available volumes	/kernel/drv/md.conf	/etc/sysconfig/lvm /etc/raidtab

Solaris

SVM uses the state database to store a copy of the configuration. It requires a minimum of half + 1 copies, called a quorum, to be in good state in order to boot up the operating system.

Prior to soft partitions, Solaris created a new file system the same size as the metadevice. So, if an SVM volume is a concatenation of two disks or a RAID 5 structure of four disks, the file system will be created to the maximum space allowed by it. With soft partitions, you can partition the whole SVM volume into smaller partitions as you see fit, and the file system that created on these soft partitions will be the size that you create.

The creation and management of the SVM volumes uses some of the subcommands such as **metainit**, **metattach**, **metastat**, and **metarename**. Refer to the task table in Table 4-8 on page 67.

Linux

In Linux, the equivalent of SVM is LVM2, which is present in both Red Hat and SUSE. The basic concepts behind LVM for Linux are the same as in SVM. The sequence of operations are simplified:

1. Partition the disk.
2. Initialize partitions.
3. The disk can be used to create a new volume or volume group (VG), or added into an existing volume or volume group.
4. Logical volumes (LV) can be created or extended on this new disk.
5. The file system can be extended to this new disk.

Note: The task table in Table 4-8 on page 67 lists the commands required to do the previous sequence. Refer to the man pages for each command for detailed information about other options available. For a good guide about LVM configuration and its use on Linux, see:

http://www.tldp.org/HOWTO/html_single/LVM-HOWTO/

Other Linux LVM tasks

Aside from the previously discussed normal volume management task, the Linux LVM has the additional following commands:

lvmdiskscan	Scan for devices visible to LVM2
lvscan	Scan disks for logical volumes
pvdiskdisplay	Display physical volume attributes

pvs	Report physical volume information
pvscan	Scan all supported logical volume block devices for physical disk
vgck	Perform consistency check on the volume group
vgscan	Scan all disks for volume group data and rebuild caches
vgimport	Import a volume group to this server
vgexport	Export a volume group out of this server

LVM, by default, does not log errors and debug messages to a file. You can change this by editing the `/etc/lvm/lvm.conf` file under the `log` section.

Linux LVM sample configuration

Example 4-8 shows `df` output for a sample disk configuration. The root disk is being used as a simple raw device. The mount point `/vm` shows that it is using an LVM structure.

Example 4-8 Sample output of `df`

```
# df -h
Filesystem      Size  Used Avail Use% Mounted on
/dev/sda1        9.7G  6.0G  3.2G  66% /
none            2.0G   0    2.0G   0% /dev/shm
/dev/sda6        1.4G   38M  1.3G   3% /home
/dev/sda5        2.9G   37M  2.7G   2% /opt
/dev/mapper/DataVG-LogVol100
                100G   12G   84G  13% /vm
```

Example 4-9 shows `vgdisplay` output, without any parameters passed to it, for the volume group (VG). If you have multiple VGs, this command will include each one of them in the list output. To display only a specific VG, provide the VG name as a parameter to the command.

Example 4-9 Sample default output of `vgdisplay` with no options

```
# vgdisplay
--- Volume group ---
VG Name          DataVG
System ID
Format           lvm2
Metadata Areas   4
Metadata Sequence No 2
VG Access        read/write
VG Status        resizable
MAX LV          0
```

Cur LV	1
Open LV	1
Max PV	0
Cur PV	4
Act PV	4
VG Size	101.62 GB
PE Size	32.00 MB
Total PE	3252
Alloc PE / Size	3248 / 101.50 GB
Free PE / Size	4 / 128.00 MB
VG UUID	egqMUy-dNyj-DSKG-zupE-V22E-cPHn-2cLLrn

Example 4-10 shows the output of the `vgdisplay` command in abbreviated format (using the `-s` option).

Example 4-10 Sample output of `vgdisplay -s`

```
# vgdisplay -s
"DataVG" 101.62 GB [101.50 GB used / 128.00 MB free]
```

Example 4-11 shows what a verbose display looks like. It also shows the information about the logical volumes (LVs) defined within the VG and the corresponding physical volumes (PVs) that are used for this VG.

Example 4-11 Sample output of `vgdisplay -v`

```
# vgdisplay -v
  Finding all volume groups
  Finding volume group "DataVG"
  --- Volume group ---
  VG Name                DataVG
  System ID
  Format                  lvm2
  Metadata Areas         4
  Metadata Sequence No   2
  VG Access               read/write
  VG Status               resizable
  MAX LV                 0
  Cur LV                  1
  Open LV                 1
  Max PV                  0
  Cur PV                  4
  Act PV                  4
  VG Size                 101.62 GB
  PE Size                 32.00 MB
  Total PE                3252
  Alloc PE / Size        3248 / 101.50 GB
```

```

Free PE / Size      4 / 128.00 MB
VG UUID             egqMUy-dNyj-DSKG-zupE-V22E-cPHn-2cLLrn

--- Logical volume ---
LV Name             /dev/DataVG/LogVol100
VG Name             DataVG
LV UUID             pmIJYM-0wVm-ZpqN-Z4dR-9TQ1-Gtg0-xCU5kS
LV Write Access     read/write
LV Status           available
# open              1
LV Size             101.50 GB
Current LE          3248
Segments           4
Allocation          inherit
Read ahead sectors  0
Block device        253:0

--- Physical volumes ---
PV Name             /dev/sdb1
PV UUID             6j6Ld4-tyMR-Swe5-oCpM-N38w-yTYZ-HTubDo
PV Status           allocatable
Total PE / Free PE 1084 / 0

PV Name             /dev/sdc1
PV UUID             NU5VjA-N7LU-SagN-u91f-xAmI-ysbY-5AbvK7
PV Status           allocatable
Total PE / Free PE 1084 / 0

PV Name             /dev/sdd1
PV UUID             CtXfvq-yrmD-SVHb-17x3-fKUh-pm5m-Gys0F7
PV Status           allocatable
Total PE / Free PE 542 / 0

PV Name             /dev/sde1
PV UUID             2WJB4T-YTQX-ibjC-r0e5-HcBC-8QoE-ywFCiJ
PV Status           allocatable
Total PE / Free PE 542 / 4

```

Example 4-12 shows the information provided by the `pvdisplay` command for a specific physical disk partition. It gives a little bit more information than the previous output.

Example 4-12 Sample output of `pvdisplay`

```

# pvdisplay /dev/sdc1
--- Physical volume ---
PV Name             /dev/sdc1
VG Name             DataVG

```

```
PV Size          33.88 GB / not usable 0
Allocatable     yes (but full)
PE Size (KByte) 32768
Total PE        1084
Free PE         0
Allocated PE    1084
PV UUID         NU5VjA-N7LU-SagN-u91f-xAmI-ysbY-5AbvK7
```

Example 4-13 shows the information provided by the `lvdisplay` command for a logical volume.

Example 4-13 Sample default output of `lvdisplay`

```
# lvdisplay /dev/DataVG/LogVo100
--- Logical volume ---
LV Name          /dev/DataVG/LogVo100
VG Name          DataVG
LV UUID          pmIJYM-0wVm-ZpqN-Z4dR-9TQ1-Gtg0-xCU5kS
LV Write Access  read/write
LV Status        available
# open           1
LV Size          101.50 GB
Current LE       3248
Segments         4
Allocation       inherit
Read ahead sectors 0
Block device     253:0
```

Example 4-14 shows a lot more information for an LV, including which physical disk device partitions the LV resides in and how it is laid out. The sample LV shows that it uses a “linear” structure across the four PVs. Note that the logical extents numbering from the first PV to the last PV are in sequence. This is equivalent to the concatenation model for SVM. This shows the mapping of logical extents to physical volumes and physical extents.

Example 4-14 Sample output of `lvdisplay --maps`

```
# lvdisplay --maps /dev/DataVG/LogVo100
--- Logical volume ---
LV Name          /dev/DataVG/LogVo100
VG Name          DataVG
LV UUID          pmIJYM-0wVm-ZpqN-Z4dR-9TQ1-Gtg0-xCU5kS
LV Write Access  read/write
LV Status        available
# open           1
LV Size          101.50 GB
Current LE       3248
```

```
Segments          4
Allocation        inherit
Read ahead sectors 0
Block device      253:0

--- Segments ---
Logical extent 0 to 1083:
  Type            linear
  Physical volume  /dev/sdb1
  Physical extents 0 to 1083

Logical extent 1084 to 2167:
  Type            linear
  Physical volume  /dev/sdc1
  Physical extents 0 to 1083

Logical extent 2168 to 2709:
  Type            linear
  Physical volume  /dev/sdd1
  Physical extents 0 to 541

Logical extent 2710 to 3247:
  Type            linear
  Physical volume  /dev/sde1
  Physical extents 0 to 537
```

4.11 VERITAS VxVM and VxFS

There are VERITAS VxVM and VxFS products for Linux systems. You can also use the Linux LVM2, which is part of the software bundle for both Red Hat and SUSE. See 4.10, “Solaris Volume Manager to Linux LVM” on page 65. For more information about the VERITAS products (now Symantec), visit:

<http://www.veritas.com/>



Software management

This chapter describes how to manage software packages and patches in Linux. Software management, in this context, includes the operating system and any software that runs on the operating system.

This chapter discusses the following topics:

- ▶ Packages
- ▶ Patching
- ▶ Dependencies
- ▶ Package distribution methods
- ▶ Automated software management
- ▶ Activating fixes after updating
- ▶ Compiling patches in Linux

5.1 Packages

A package is a collection or group of program files and subdirectories that make up a software product. A package can be part of the operating system or an add-on software for specific functionality.

Task table

Table 5-1 illustrates package management commands.

Table 5-1 Package management commands

Package management task	Solaris	Linux
Install packages	<code>pkgadd</code>	<code>rpm -i</code>
Display installed packages	<code>pkginfo</code> or <code>pkgparam</code>	<code>rpm -qa</code>
Remove software package	<code>pkgrm</code>	<code>rpm -e</code>
Upgrade/install a package	<code>pkgadd</code>	<code>rpm -U</code>
Verify correct installation	<code>pkgchk</code>	<code>rpm -V</code>
List the contents of an installed package	Look in <code>/var/sadm/install/contents</code>	<code>rpm -ql</code>
Check which file to which a package belongs	Look in <code>/var/sadm/install/contents</code>	<code>rpm -qf</code>
Check package information	Look in <code>/var/sadm/pkg/PKGNAME/pkginfo</code>	<code>rpm -qi</code>

5.1.1 Package management in Solaris

The Solaris package format is based on the System V Interface Definition for Application Binary Interface (ABI) and has tools for managing these packages. These management tools include the ability to add and remove packages, check for consistency, and display package information. If system administrators use the `pkgadd` and `pkgrm` tools to manage packages, the tools update the software product database accordingly.

5.1.2 Package management in Linux

Red Hat and Novell SUSE use the Red Hat Package Manager (RPM), which includes features for package management such as dependency and signature checking and other advanced options. For more information about RPM, go to the official site:

<http://www.rpm.org>

Refer to the following Web sites for distribution-related information:

- ▶ Red Hat: Chapter 15, “Package Management with RPM,” in *Red Hat Enterprise Linux 4: Reference Guide*, available at:
<http://www.redhat.com/docs/manuals/enterprise/>
- ▶ Novell SUSE: The “RPM --- the Package Manager” section in the “Installation” chapter of *SUSE LINUX Enterprise Server 9 Administration Guide for OES*, available at:
<http://www.novell.com/documentation/sles9/>

Example 5-1, Example 5-2, and Example 5-3 on page 78 provide sample output from using the `rpm` command.

Example 5-1 Sample output: List all packages in server

```
# rpm -aq
filesystem-2.3.0-1
bzip2-libs-1.0.2-13
gdbm-1.8.0-24
libsepol-1.1.1-2
bash-3.0-19.2
popt-1.9.1-9_nonpt1
vim-minimal-6.3.046-0.40E.4
gawk-3.1.3-10.1
openssl-0.9.7a-43.1
shadow-utils-4.0.3-41.1
cracklib-dicts-2.7-29
...
mozilla-nspr-1.7.7-1.4.2
intltool-0.31.2-1
libcroco-0.6.0-4
vim-enhanced-6.3.046-0.40E.4
gail-1.8.0-2
metacity-2.8.6-2.1
```

Example 5-2 Sample output: List files in a particular package

```
# rpm -ql GConf2-2.8.1-1
/etc/gconf/2
/etc/gconf/2/path
/etc/gconf/gconf.xml.defaults
/etc/gconf/gconf.xml.mandatory
/usr/bin/gconf-merge-tree
/usr/bin/gconftool-2
/usr/lib/GConf
/usr/lib/GConf/2
/usr/lib/GConf/2/libgconfbackend-olddxml.so
```

```
/usr/lib/GConf/2/libgconfbackend-xml.so
....
/usr/share/locale/yi/LC_MESSAGES/GConf2.mo
/usr/share/locale/zh_CN/LC_MESSAGES/GConf2.mo
/usr/share/locale/zh_TW/LC_MESSAGES/GConf2.mo
/usr/share/man/man1/gconftool-2.1.gz
/usr/share/sgml
/usr/share/sgml/gconf
/usr/share/sgml/gconf/gconf-1.0.dtd
```

Example 5-3 Sample output: Package information

```
# rpm -qi GConf2-2.8.1-1
Name           : GConf2                      Relocations: (not relocatable)
Version        : 2.8.1                      Vendor: Red Hat, Inc.
Release        : 1                          Build Date: Tue 12 Oct 2004
12:16:24 PM CDT
Install Date: Thu 13 Oct 2005 03:15:55 PM CDT      Build Host:
tweety.build.redhat.com
Group          : System Environment/Base       Source RPM: GConf2-2.8.1-1.src.rpm
Size           : 4050741                      License: LGPL
Signature      : DSA/SHA1, Wed 05 Jan 2005 03:47:04 PM CST, Key ID
219180cddb42a60e
Packager       : Red Hat, Inc. <http://bugzilla.redhat.com/bugzilla>
URL            : http://www.gnome.org
Summary        : A process-transparent configuration system
Description    :
GConf is a process-transparent configuration database API used to
store user preferences. It has pluggable backends and features to
support workgroup administration.
```

5.2 Patching

A patch is a subset of a package that adds new functionality to a package or fixes a problem with the installed package.

Task table

Table 5-2 illustrates patch management commands.

Table 5-2 Patch management commands

Patch management task	Solaris	Linux
Install a patch	patchadd	rpm -F (upgrades only if already present)
Remove a patch	patchrm	N/A
Display installed patches	showrev -p	N/A

5.2.1 Patching in Solaris

Patches in Solaris are handled using the following commands: **patchadd**, **patchrm**, and **showrev**. The last two digits of the patch file name show the revision level.

5.2.2 Patching in Linux

In comparison to Solaris, Linux manages software updates and fixes in a different way. Instead of creating patches, vendors create full packages of fixed software to simplify package maintenance. If a system administrator needs to install a new package, the process only requires installing the latest version. This method is in contrast to the process of requiring the installation of the base version package first, followed by all the associated patches.

For RPM-based distributions a system administrator can use the **rpm -U** command to upgrade a package in Linux. In addition, administrators can use the **rpm -F** command to update only if the package is already present.

Note: The concept of a “patch” differs in Linux as compared to Solaris. A patch in Linux refers to a text file that contains code that must be applied to a preexisting source code or script file. Patching source code files also requires the user to recompile the binaries in order to benefit from the update. See 5.7, “Compiling patches in Linux” on page 82 for more information.

5.3 Dependencies

In certain situations, a package or patch will have a dependency on one or more packages or patches before it can be installed.

5.3.1 Dependency management in Solaris

A Solaris system administrator has several options available for resolving package dependencies:

- ▶ Manual (that is, an individual package or patch).
- ▶ A patch cluster, which already contains most of what is needed for the core operating system.
- ▶ An automated patch manager (see 5.5, “Automated software management” on page 81).

Each of these options has certain considerations that need to be taken into account according to the needs of the system administrator. For example, to target a specific issue, only downloading the required patches and its prerequisites is required. However, consider a cluster patch when there is a need to ensure that the key components of the operating system are all up to date.

5.3.2 Dependency management in Linux

In an RPM-based distribution of Linux, you can use two methods for resolving package dependencies:

1. Manual: Using the `rpm -i` command to install a package provides information about what dependent packages are missing, if any.
2. Automatic: Enterprise Linux distribution vendors typically provide tools for automatic dependency management. Some examples include:
 - Red Hat: `up2date`
 - Novell SUSE: `you`

Refer to 5.5.2, “Automated software management in Linux” on page 81 for more information.

5.4 Package distribution methods

Solaris has the SunSolve Web site for retrieving distributed packages and patches. Red Hat and Novell also provide facilities for package distribution:

- ▶ Red Hat: The Red Hat Network provides package updates and files:
<https://rhn.redhat.com>
- ▶ Novell SUSE: The SUSE Linux portal provides package updates and fixes (requires registration):
<https://portal.suse.com>

5.5 Automated software management

Automated software management is an easy way to update a system without having to individually check through each patch or package for dependencies and requirements.

5.5.1 Automated software management in Solaris

There are several options available from the SunSolve Web site for automated software management. The software in this context refers to patches only and not packages. Those options include:

- ▶ Patch Manager for Solaris 8 and 9
- ▶ PatchPro Interactive
- ▶ PatchPro Expert

These tools generate a complete list of patches based on your current system, which includes patches for software components that are installed on the system but are not being used. Some of these tools enable you to pick from the generated list so that you can target only specific software components.

5.5.2 Automated software management in Linux

In Linux, each distributor has their own way of performing automated software management. With both systems, you can configure a default policy for software upgrades and have a daemon perform the updates in the background based on the schedule defined in the policy.

Red Hat

The default tool for automated dependency and update management is **up2date**. This tool connects to the Red Hat Network and checks for available updates for your system. After **up2date** gathers a list of updates, it presents you with the option to select which packages to download and update. After the selections step, the tool completes the remaining installation and update steps. For more information, refer to the *Red Hat Network 4.0 Reference Guide*, available at:

<http://www.redhat.com/docs/manuals/RHNetwork/>

Novell SUSE

The default tool for automated dependency and update management is **you** (YaST online update), a YaST2 module that has a similar functionality to the Red Hat **up2date** tool. For more information, see the “YaST Online Update” section in

the “Installation” chapter of *SUSE LINUX Enterprise Server 9 Administration Guide for OES*, available at:

<http://www.novell.com/documentation/sles9/>

5.6 Activating fixes after updating

Certain packages and patches require a reboot to activate the update or fix, especially for kernel and library patches.

5.6.1 Patch activation in Solaris

Occasionally, a system will have to regenerate its devices during the reboot if the fix is for driver components. In most kernel or library patches, Sun recommends installing in single user mode. This allows the system to be immediately rebooted after applying the update, while minimizing the potential for a loss of user data.

5.6.2 Patch activation in Linux

In Linux, there is no such thing as reconfiguration reboot or device regeneration. Sometimes it is possible to update kernel modules without rebooting the system by removing the affected module from kernel, updating the kernel module, and reinserting the new kernel module. For information about this, see “Installing device driver modules” on page 92. In addition, you can update the system with a new kernel while it is running and wait to boot the new kernel during a maintenance window. The system continues to run with the old kernel, without interruption, until the system is rebooted.

5.7 Compiling patches in Linux

For a Solaris system administrator, there is no need to compile the kernel, driver, or any of its core tools because there is no source code available in all versions of Solaris before version 10. Compilation is usually done when GNU freeware is required and has to be installed. Even in this situation, there are usually downloadable precompiled binaries.

In Linux, it is usually possible to compile programs because the majority of the applications come with source code. All the tools necessary for compiling, assembling, and linking are usually present in the operating system. If not, then those tools can be installed.

Some driver or third-party applications might require compilation to allow them to work with the systems kernel. Most source code and third-party applications

come with instructions about how to compile and install. If you still need more help, you can contact the provider of the source or third-party application to guide you through this process.

The following tutorials are available in the open source section of the IBM developerWorks® Web site:

<http://www.ibm.com/developerworks/opensource>

- ▶ “Compiling sources and managing packages,” available at:

<http://www.ibm.com/developerworks/edu/1-dw-linux-lpir25-i.html>

- ▶ “Configuring and compiling the kernel,” available at:

<http://www.ibm.com/developerworks/edu/1-dw-linux-lpir26-i.html>



Device management

This chapter provides a description of common tasks for managing devices in Linux for the Solaris system administrator.

This chapter includes the following topics:

- ▶ Device access and configuration
- ▶ Removable media devices
- ▶ Terminals and modems
- ▶ Distribution-based device management tools

6.1 Device access and configuration

This section outlines the specific tasks needed to access and configure devices in Linux. We describe the following topics:

- ▶ Device naming and access
- ▶ Displaying device configuration information
- ▶ Adding a device
- ▶ Hot-plug devices

Task table

Table 6-1 illustrates device management commands.

Table 6-1 *Solaris versus Linux device management commands*

Device management task	Solaris	Red Hat	SUSE
Run multiple tasks in a GUI environment	<code>admintool</code>	<code>system-config-*</code> (keyboard, mouse, printer, network, soundcard, and so on)	<code>yast2</code>
Configure a device	<code>devfsadm</code>	<code>/dev/MAKEDEV</code> <code>kudzu</code>	<code>/dev/MAKEDEV</code> <code>hwinfo</code>
Define a device	<code>devfsadm</code>	<code>mknod</code> <code>kudzu</code>	<code>mknod</code> <code>hwinfo</code>
Remove a device	<code>devfsadm -C</code> or <code>-c</code>	<code>/dev/MAKEDEV</code>	<code>/dev/MAKEDEV</code>
List devices	<code>sysdef</code> or <code>prtconf</code>	<code>ls /sys/devices</code> <code>cat /proc/devices</code>	<code>ls /sys/devices</code> <code>cat /proc/devices</code>
Install a device driver kernel module	<code>modload</code>	<code>insmod</code>	<code>insmod</code>
List installed device driver kernel modules	<code>modinfo</code>	<code>lsmod</code>	<code>lsmod</code>
Remove a device driver kernel module	<code>modunload</code>	<code>rmmmod</code>	<code>rmmmod</code>

6.1.1 Device naming and access

Linux is similar to Solaris in the way it represents devices on the system by using logical and physical device files.

Table 6-2 Solaris versus Linux device file location

Solaris	Linux	Description
/dev	/dev	Contains logical device files
/devices	/sys/devices	Contains physical device files
devfsadm	udev	Command that creates and manages the physical device files and the associated logical device files

Solaris and Linux both use the logical device names in the /dev file system for many of the disk-related system administration tasks. However, the file and directory naming convention differs between the two operating systems.

Solaris logical disk devices

Solaris administrators refer to disk devices by selecting the subdirectory that it is linked to (either /dev/rdisk or /dev/dsk), followed by a string of information that indicates the specific controller, disk, and slice:

```
/dev/dsk/c[1]t[2]d[3]s[4]
```

Where:

- ▶ 1: Logical controller number
- ▶ 2: Physical bus target number
- ▶ 3: Driver number
- ▶ 4: Slice or partition number

Linux logical disk devices

The Linux kernel represents disk devices as a pair of numbers <major>:<minor>. Some major numbers are reserved for particular device drivers, others are dynamically assigned to a device driver when Linux boots. A major number can also be shared by multiple device drivers. The device driver uses the minor number to distinguish individual physical or logical devices. Device drivers assign device names to their devices, according to a device driver-specific naming scheme. For example:

- ▶ The first SCSI disk is named /dev/sda, the second disk is named /dev/sdb, the third is /dev/sdc, and so on. After /dev/sdz is reached, the cycle starts over with /dev/sdaa, followed by /dev/sdab, and so on.
- ▶ The first floppy drive is named /dev/fd0.
- ▶ The first SCSI CD-ROM is named /dev/scd0, which is also known as /dev/sr0.
- ▶ The master disk on the IDE primary controller is named /dev/hda and the slave disks is named /dev/hdb, while the master and slave disks of the secondary controller are /dev/hdc and /dev/hdd.

The output in Figure 6-1 is an example of what a Linux system administrator might see when performing an `ls -l` command on `/dev/hda1`.

```
brw-rw---- 1 root disk 8, 1 Mar 15 2005 hda1
```

The diagram shows the following breakdown of the device name `hda1`:

- h: Device Type
- d: Drive Controller and Number
- a: Drive Type
- 1: Minor Number
- 8: Major Number

Figure 6-1 Example of IDE device in Linux

The previous listing represents the first partition of the first drive on the first IDE controller. Linux defines device files with either type “c” for character devices, or type “b” for block devices. All disks, such as the previous one, are represented as block devices in Linux. For an example of a character device, refer to 6.3.3, “Serial port management” on page 103.

This next output is an example of what a Linux system administrator might see when performing an `ls -l` command on `/dev/sda1`, as shown in Figure 6-2.

```
brw-rw---- 1 root disk 8, 1 Mar 15 2005 sda1
```

The diagram shows the following breakdown of the device name `sda1`:

- s: Device Type
- d: Drive Controller
- a: Drive Type
- 1: Minor Number
- 8: Major Number

Figure 6-2 Example of SCSI device in Linux

For more information about device names and numbers, the Linux Assigned Names And Numbers Authority (LANANA) maintains the Linux Device List. This is the official listing of device numbers and `/dev` directory nodes for Linux, available at:

<http://www.lanana.org/>

Accessing devices

In Solaris, there is the concept of having a raw device logical interface name (rsk) and a block device logical interface name (dsk) for the same device. The name a system administrator uses depends on the task. In Linux, there is only one type of logical device name scheme used. Table 6-3 lists a few Solaris versus Linux commonly used commands for accessing devices.

Table 6-3 Commonly used logical device access commands in Solaris and Linux

Solaris example	Linux example
<code>df /dev/dsk/c1t0d0s0</code>	<code>df /dev/sda1</code>
<code>fsck -p /dev/rdisk/c1t3d0s4</code>	<code>fsck -y /dev/sda2</code>
<code>mount /dev/dsk/c1t1d0s0 /mnt/1</code>	<code>mount /dev/hda2 /data</code>
<code>newfs /dev/rdisk/c0t0d1s1</code>	<code>mkfs /dev/sdc1</code>
<code>prtvtoc /dev/dsk/c1t1d0s0</code>	<code>fdisk -l /dev/hdc</code>

Refer to the man pages for these commands for more information.

6.1.2 Displaying device configuration information

In Solaris, there are three commands for displaying system and device configuration information:

- ▶ `prtconf`
- ▶ `sysdef`
- ▶ `dmesg`

There are several methods to obtain similar device-related information in Linux.

/proc

The Linux /proc virtual file system differs from the Solaris implementation in that besides having information about running processes, it also contains runtime system information about areas such as devices mounted, system memory, and hardware configuration. Table 6-4 on page 90 outlines some of the files and directories that contain device-related information.

Table 6-4 Device-related information in the /proc file system

Location	Description
/proc/bus/	Directory containing bus-specific information
/proc/devices	File that lists block and character device drivers configured into the current running kernel
/proc/ide	Directory containing information about all IDE devices known to the kernel
/proc/scsi	Directory containing information about all SCSI devices known to the kernel

Red Hat and Novell SUSE also provide additional information about the /proc file system:

- ▶ In Red Hat, refer to Chapter 5, “The /proc File System,” in *Red Hat Enterprise Linux 4: Reference Guide*, available at:

<http://www.redhat.com/docs/manuals/enterprise/>

- ▶ In Novell SUSE, refer to the “The /proc File System,” section of the “Administration” chapter of *SUSE LINUX Enterprise Server 9 Administration Guide for OES*, available at:

<http://www.novell.com/documentation/sles9/>

Important: Although some device driver information can be found in /proc, using it for this purpose is deprecated in recent Linux distributions, and the sysfs virtual file system (mounted under /sys) should be used instead.

/sys

The sysfs virtual file system provides a hierarchical view in the user space of kernel data structures, their attributes, and the links between them. The **hotplug** command notifies user space when a device is added or removed, keeping the file system dynamic. Features include:

- ▶ sysfs is linked to the kobject infrastructure. Objects in the kernel are represented in a directory tree. Each kernel object is represented as a subdirectory in the tree.
- ▶ sysfs forwards file I/O operations to methods defined for the attributes, providing a means to read and write kernel attributes.
- ▶ sysfs enables a user to determine which device has been assigned to which device file by the kernel. Previously, this information was not easily obtainable.
- ▶ sysfs can be mounted just like other virtual file systems. Only `cat(1)` and `echo(1)` are needed, but `tree(1)` is also good to have for viewing the overall structure.

Table 6-5 outlines some of the directories that contain device-related information.

Table 6-5 Device-related information in the /sys file system

Location	Description
/sys/block/	Has subdirectory for each block device discovered in the system.
/sys/bus/	Has subdirectories for each supported physical bus type.
/sys/class/	Every device class that is registered with the kernel is represented here. An example of a class is sound or printer.
/sys/device/	A hierarchical structure that has every physical device discovered by the bus types seen by the system.

For an in-depth look at sysfs, refer to *The sysfs file system*, by Patrick Mochel, presented in volume 1 of the 2005 Ottawa Linux Symposium proceedings, available at:

<http://www.linuxsymposium.org/2005/>

The lspci and lsusb commands

The **lspci** and **lsusb** commands display information about all PCI and USB buses in the system and all devices connected to them. For more information and options about these commands, refer to the man page.

The dmesg command

This command essentially performs the same function as it does in Solaris. For more information about this command, refer to the man page.

Red Hat and Novell SUSE also provide their own files and commands for obtaining device information:

- ▶ Red Hat:
 - /etc/sysconfig/hwconf: This file contains information about all the hardware detected on the system, which includes the drivers used, vendor ID, and device ID information. For more information, refer to Chapter 4, “The sysconfig Directory,” in *Red Hat Enterprise Linux 4: Reference Guide*, available at:

<http://www.redhat.com/docs/manuals/enterprise/>

- **hwbrowser**: This command invokes an X Window System graphical user interface that enables you to browse the hardware configurations. For more information, refer to Section 4 in Chapter 40, “Gathering System Information,” in *Red Hat Enterprise Linux 4: System Administration Guide*, available at:

<http://www.redhat.com/docs/manuals/enterprise/>

► **Novell SUSE:**

hwinfo: This command probes for the hardware present in the system and can generate a system overview log. Refer to the man page and the readme file located under `/usr/share/doc/packages/hwinfo` for more information about this command.

6.1.3 Adding a device

The process of adding a device (excluding hot-pluggable) involves shutting down the system, physically connecting the device, and powering the system back up. If the device contains its own power source, make sure that the power is turned on before powering up the system.

Device driver modules

The Linux operating system uses kernel modules for device drivers in a manner similar to that used in Solaris. Most device driver modules are located under `/kernel/drivers` in the release directory within the `/lib/modules/` file system. The specific release level is determined by issuing the **uname -r** command. For example, a Red Hat Enterprise Linux system might return the following output, which means `/lib/modules/2.6.9-11.ELsmp/kernel/drivers` contains the loadable device driver modules for the system:

```
# uname -r
2.6.9-11.ELsmp
```

Automatic device configuration

Red Hat and Novell SUSE both support automatic device configuration at boot time. This means that the system will recognize most devices and make them available after powering up the system.

Installing device driver modules

Occasionally, a system administrator might need to update or install new device driver modules. In these circumstances, follow the instructions provided by the device manufacturer or module creator. Refer to Table 6-1 on page 86 for module-related tasks in Solaris and their associations in Linux. Table 6-6 on page 93 lists additional module-related commands. Refer to the command’s man page for information about its use.

Table 6-6 Additional module-related commands

Module-related task	Command
Add and remove modules based on the system configuration	modprobe
Display information about a kernel module	modinfo

6.1.4 Hot-plug devices

Hot-plugging is the adding and removing of devices while the system is running. Linux includes hot-plug support for the Small Computer System Interface (SCSI), Peripheral Component Interconnect (PCI), and Universal Serial Bus (USB), devices just as Sun Solaris, as well as additional devices such as IEEE 1394 (Firewire) support.

While Solaris requires the use of the **cfgadm** command to manage hot-plug devices, a Linux system automatically recognizes and configures a supported hot-pluggable device when connected. Conversely, the system automatically deconfigures a hot-pluggable device when removed. Linux provides the **hotplug** and **udev** commands for managing hot-plug devices, and the man pages contain instructions about using both of these commands. Red Hat and Novell SUSE also provide additional information about Linux hot-plugging:

- ▶ Red Hat: Refer to 7.1.1, “Hotplug Devices,” in the *Red Hat Enterprise Linux 3: System Administration Guide*, available at:
<http://www.redhat.com/docs/manuals/enterprise/>
- ▶ Novell SUSE: Refer to the “The Hotplug System” section in *SUSE LINUX Enterprise Server 9 Administration Guide for OES*, available at:
<http://www.novell.com/documentation/sles9/>

Background information

For general information about Linux hot-plugging, refer to the project Web site at:

<http://linux-hotplug.sourceforge.net/>

Also refer to the following Ottawa Linux Symposium (OLS) presentations by Greg Kroah-Hartman:

- ▶ OLS 2001: “Hotpluggable devices and the Linux Kernel”
- ▶ OLS 2003: “udev – A Userspace Implementation of devfs”

The presentations are available at:

<http://www.kroah.com/linux/>

6.2 Removable media devices

This section covers the following topics:

- ▶ Supported types
- ▶ Managing and accessing removable media
- ▶ Formatting removable media
- ▶ CDs and DVDs
- ▶ Tape drives

6.2.1 Supported types

Linux fully supports many types of removable media, ATAPI, SCSI, and USB connected devices, such as:

- ▶ DVD-ROM/R/RW
- ▶ CD-ROM/R/RW
- ▶ Iomega Zip and Jaz drives
- ▶ Diskettes
- ▶ Tapes

The locations of the device mount points differ from the Solaris operating system, and Table 6-7 lists information for accessing some common removable devices.

Table 6-7 How to access common removable media devices

What to access	Where to access
Files on the first floppy disk	/media/floppy
Files on the second floppy disk	/media/floppy1
Files on the first CD or DVD	/media/cdrom
Files on the second CD or DVD	/media/cdrom1

For questions about Linux support for a specific device, contact the Linux distributor or the device manufacturer.

6.2.2 Managing and accessing removable media

Methods implemented in the operating system for automatic mounting of removable devices vary between Solaris and Linux, and also between different Linux distributions. In addition, depending on which graphical session manager

you optionally choose to run on top of the operating system, the types of device management and access tools available in those session managers vary as well.

Solaris

Solaris uses the volume manager (vold) to actively manage removable media devices. The volume manager simplifies the process for using removable media by automatically performing many of the tasks required to mount and access removable media (on earlier versions of Solaris).

Red Hat

Red Hat Enterprise Linux semi-automatically manages removable media devices. Most removable devices are auto-detected for Nautilus File Manager users, who will see a desktop link to the device upon startup. Shell console users will have to manually mount and unmount removable devices. For more information about accessing removable devices, refer to Chapter 13, “Diskettes and CD,” in *Red Hat Enterprise Linux 4: Red Hat Enterprise Linux Step By Step Guide*, available at:

<http://www.redhat.com/docs/manuals/enterprise/>

Novell SUSE

Novell SUSE Linux Enterprise System operating system automatically manages removable media devices using the submount file system (subfs). The submount program automatically mounts removable media devices when the mount point is accessed, and subsequently unmounts them when they are no longer in use. Issuing the `cd /media/<device>` command initiates the automatic mounting process.

Note: Media cannot be ejected as long as it is being accessed by a program.

For more information about submount, consult the man pages or the project Web site at:

<http://submount.sourceforge.net/>

Access

System administrators access files on mounted removable media devices using the same commands as they do when accessing other file systems. Commands such as `ls`, `cp`, `rm`, and `mv` behave the same way on removable media, with CD/DVD and tape media being the exception (see 6.2.4, “CDs and DVDs” on page 96 and 6.2.5, “Tape drives” on page 98).

Remote systems

The process of allowing remote systems to access removable media file systems is no different than it is for standard file systems. You can export the removable media file system mount points using Network File System (NFS), Samba, or any other supported networked file system protocols. In addition, accessing a remote removable media file system is the same process as it is for standard remote file systems.

The eject command

The **eject** command operates in the same manner in Linux as it does in Solaris. For details about its usage options, refer to the man page.

The mzip command

The **mzip** command is part of the mtools package that enables you to issue Zip and Jaz disk-specific commands on Linux. Refer to the man pages about **mzip** and mtools for more information.

6.2.3 Formatting removable media

Linux does not have an equivalent command to Solaris's **rmformat**. Therefore, superuser access must be attained in order to format removable media from the command line. The standard file system commands, such as **fdisk**, **fsck**, and **mkfs**, are applicable to most removable media devices (see 6.2.4, "CDs and DVDs" on page 96 and 6.2.5, "Tape drives" on page 98 for exceptions).

The fdformat command

The **fdformat** command does a low-level format of the floppy diskette. Consult the man page for further information.

The gfloppy command

This is a GUI desktop interface for formatting floppy disks. Consult the man page for further information.

6.2.4 CDs and DVDs

This section provides an overview of commands and tools for accessing, writing, and erasing CD and DVD optical media.

Accessing

When mounted, both CD and DVD media can be accessed using standard file system commands, with the exception of commands that involve the writing and erasing of data, that is, **rm**, **cp**, and **mv**.

Writing

Both Red Hat and SUSE provide graphical tools for creating audio and data CDs through their desktop applications:

- ▶ Red Hat
 - The Nautilus File Manager has an integrated tool called the CD/DVD Creator.
- ▶ Novell SUSE
 - The K3b application is provided for CD burning. For more information about this application, go to:
<http://k3b.sourceforge.net/>
 - The YaST tool provides a CD creator module. For more information about this application, run the **yast2 cd-creator** command as root.

If you do not want to use one of the previous tools, important command line tools for managing this process include:

- ▶ **cdrecord**: This command is similar to **cdrw** on Solaris and is for writing CD file systems. Linux and **cdrecord** support the ISO 9660 format with Rock Ridge or Joliet extensions. For more information about other supported formats and instructions for using **cdrecord**, refer to the man page or the project Web site at:
<http://freshmeat.net/projects/cdrecord/>
- ▶ **mkisofs**: This command is the same for both Solaris and Linux. Refer to the man page for more information about its use.
- ▶ **dvd+rw-tools**: Red Hat and Novell SUSE ship with this package, which provides the following commands for manipulating +RW/+R and -R[W] DVD media:
 - **dvd+rw-booktype**
 - **dvd+rw-format**
 - **dvd+rw-mediainfo**
 - **dvd-ram-control**
 - **growisofs**

For more information about this package, refer to the **growisofs** man page and the following system documentation:

- Red Hat: `/usr/share/doc/dvd+rw-tools-<version>`
- Novell SUSE: `/usr/share/doc/packages/dvd+rw-tools`

- ▶ **dvdrecord**: Red Hat provides this additional command, which is a stub for the **cdrecord** command. The man page documentation recommends using the **growisofs** command from the dvd+rw-tools package instead of this command.

The previously mentioned CD/DVD Creator (Red Hat) and K3b (Novell SUSE) applications also provide DVD burning capabilities.

Erasing

The **cdrecord** command is capable of erasing CD-RW media. The dvd+rw-tools package provides the ability to format and erase DVD media.

Audio and video

Table 6-8 describes the commands and tools used to extract and play audio data from CD and DVD media.

Table 6-8 Solaris versus Linux audio extraction and playback tools

Task	Solaris	Red Hat	Novell SUSE
Extraction shell command	cdrw	cdparanoia	cdparanoia
Extraction GUI command	xmcd	sound-juicer	k3b
Playback shell command	cda	cdda2wav	cdda2wav
Playback GUI command	xmcd	xmms	xmms

Note: Although Sun recommends **xmcd**, Solaris does not include the tool.

Red Hat and Novell SUSE do not provide tools for video extraction or playback. However, the following Web site provides software and information resources related to using DVDs with Linux:

<http://dvd.sourceforge.net/>

6.2.5 Tape drives

Linux supports an assortment of tape drives. For information about whether or not a particular drive is supported under Linux, refer to the manufacturer's Web site.

Supported formats

The following list outlines the tape formats supported in Linux:

- ▶ Travan

- ▶ Digital Audio Tape (DAT)
- ▶ Digital Linear Tape (DLT)
- ▶ Super Digital Linear Tape (Super DLT)
- ▶ 8 mm
- ▶ Advanced Intelligent Tape (AIT)

Tape device name format

Solaris administrators refer to tape devices by selecting the subdirectory that it is linked to `/dev/rmt`, followed by a string of information that indicates the specific drive number, density, and device options:

```
/dev/rmt/[1][2][3][4]
```

Where:

- ▶ 1: Drive number
- ▶ 2: Optional density flag: [l]ow, [m]edium, [h]igh, [u]ltra, or [c]ompressed
- ▶ 3: Berkeley compatibility flag: [b]
- ▶ 4: No rewind option: [n]

Linux has a different scheme for naming physical tape devices depending on the tape device type. Due to this variance on names, the `/dev/tape` device name is used for simplicity and links to the first tape device. Following sections highlight a few examples of supported tape device types and their naming scheme. For a complete list, see the device list maintained by the Linux Assigned Names And Numbers Authority (LANANA), available at:

<http://www.lanana.org>

SCSI tape device

A generic SCSI tape device is indicated by an `st` in the device name. Most SCSI tape devices use this name scheme. The mode setting is optional and defined by the manufacturer. Figure 6-3 on page 100 shows a SCSI tape example.

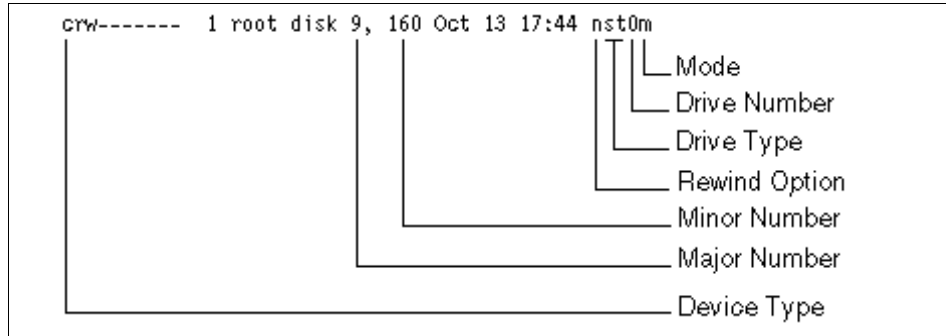


Figure 6-3 SCSI tape example

IDE tape device

An IDE tape device is indicated by an `ht` in the device name. Figure 6-4 shows a IDE tape example.

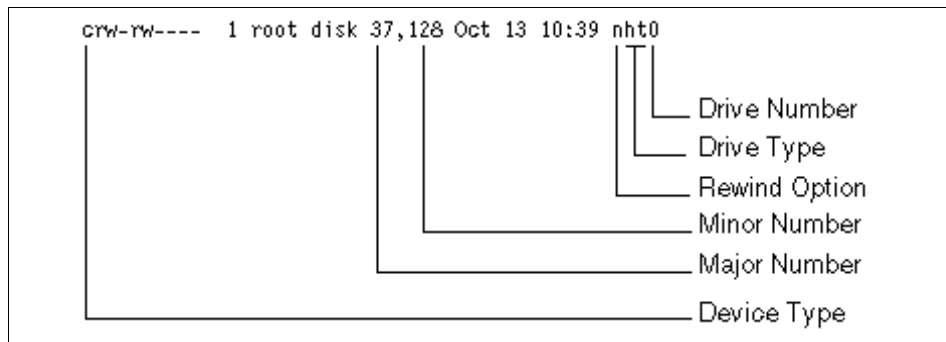


Figure 6-4 IDE tape example

QIC-117 tape device

A QIC-117 tape device is indicated by a `qft` in the device name. The compression part of the name can be left out to indicate no compression. In addition, this same name section can contain the word `raw` to indicate no file marks. Figure 6-5 on page 101 shows a QIC-117 tape example.

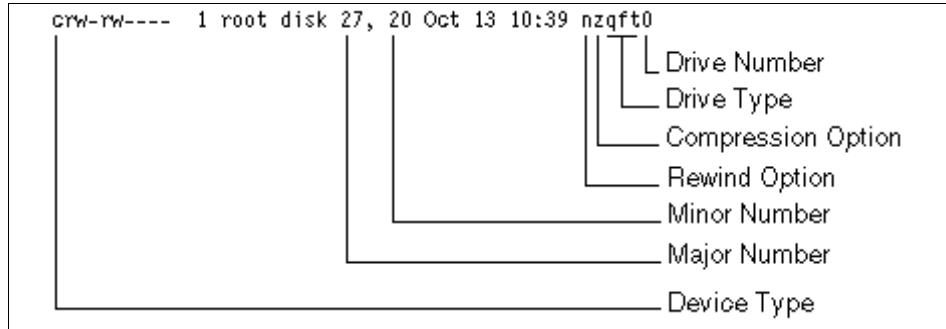


Figure 6-5 QIC-117 tape example

Displaying tape drive status

There are several ways to display the status of a Linux tape device. Issuing a `man -k tape` command lists the tape-related commands available for Red Hat and Novell SUSE. The following commands are a commonly used subset of both distributions' tape-related commands.

The mt command

Both Solaris and Linux have the `mt` command for controlling magnetic tapes. The Linux version contains all the options of the Solaris version and more. Refer to the man page for more information about this command.

The mtx command

The `mtx` command controls SCSI media changers such as tape changers and tape libraries. Refer to the man page for more information about this command.

The tapeinfo command

The `tapeinfo` command reads information from SCSI tape drives that is not usually available by using the tape drivers supplied by vendors. Refer to the man page for more information about this command.

6.3 Terminals and modems

This section covers the following terminal- and modem-related topics:

- ▶ Terminal setup and initialization
- ▶ Modem setup tools
- ▶ Serial port management
- ▶ Port monitoring

6.3.1 Terminal setup and initialization

The Linux operating system uses virtual terminals to establish console access through the keyboard and monitor attached to the system. There are six virtual terminals configured, tty1 through tty6, and each terminal allows user logins. Users can switch between virtual terminals by typing Alt+F1 through Alt+F6. If the system is using X Window System, users must use Ctrl+Alt+F1 through Ctrl+Alt+F6. The /etc/inittab file contains the configuration information for these tty terminals, and the LANANA device list contains information about all tty devices, available at:

<http://www.lanana.org>

Often, a Solaris system administrator needs to configure a terminal to run across a serial port. Sun recommends using Solaris Management Console's (smc) Serial Ports Tool for setting up terminals in Solaris.

The Linux operating system represents serial port devices under the /dev file system as ttyS#, where “#” represents the serial port number. Unfortunately, neither Red Hat nor Novell SUSE provide a tool for configuring serial port terminals; however, the process to accomplish this setup is simple. The following sections provide the steps required to enable this feature in Linux.

Red Hat

Perform the following steps:

1. Open the /etc/inittab file.
2. Locate the following section:

```
# Run gettys in standard runlevels
1:2345:respawn:/sbin/mingetty tty1
2:2345:respawn:/sbin/mingetty tty2
3:2345:respawn:/sbin/mingetty tty3
4:2345:respawn:/sbin/mingetty tty4
5:2345:respawn:/sbin/mingetty tty5
6:2345:respawn:/sbin/mingetty tty6
```

3. Add the following line below this section:

```
S0:12345:respawn:/sbin/agetty -L <baud> <serial device> <TERM>
```

Where:

<baud> = baud rate of connection, for example, 38400

<serial device> = serial port device to which to attach terminal, for example, ttyS0

<TERM> = terminal type to run over connection, for example, xterm

4. Run the **telinit q** command, which tells the **init** program to check, or “query”, the `/etc/inittab` file and make changes as needed.

Novell SUSE

Perform the following steps:

1. Open the `/etc/inittab` file.
2. Locate the following section:


```
#
#S0:12345:respawn:/sbin/agetty -L 9600 ttyS0 vt102
```
3. Remove the “#” and set the baud rate, serial device, and terminal type as needed.
4. Run the **telinit q** command, which tells the **init** program to check, or “query”, the `/etc/inittab` file and make changes as needed.

6.3.2 Modem setup tools

Red Hat and Novell SUSE provide configuration tools and instructions for setting up modems, as described in Table 6-9.

Table 6-9 *Modem configuration*

Linux distribution	Command to start tool	Location of instructions
Red Hat	system-config-network	Refer to Chapter 18, “Network Configuration,” in <i>Red Hat Enterprise Linux 4: System Administration Guide</i> , at: http://www.redhat.com/docs/manuals/enterprise/
Novell SUSE	yast modem	Refer to the “Network Integration” section in the “Services” chapter of <i>SUSE LINUX Enterprise Server 9 Administration Guide for OES</i> , at: http://www.novell.com/documentation/sles9/

6.3.3 Serial port management

Linux provides the **setserial** command for setting and retrieving information about serial ports. The **setserial** command can configure this common list of parameters:

- ▶ **port**: Configures the I/O port number
- ▶ **irq**: Configures the IRQ number
- ▶ **uart**: Configures the universal asynchronous receiver/transmitter (UART) type

- ▶ `baud_base`: Configures the speed of the serial port

Refer to the man page for further information about, as well as options for, the `setserial` command.

Port initialization

To initialize a specific serial port, such as `ttyS0`, run the following command:

```
setserial /dev/ttyS0 autoconfig
```

Novell SUSE provides an additional method for configuring serial ports. Initialization can be accomplished using the `setserial` script, located in `/etc/init.d`. Red Hat Enterprise Linux also provides a `setserial` command for managing initialization and settings. Refer the man page for `setserial` for more information.

Serial console port configuration

Assuming a login is configured on `ttyS0` (see 6.3.1, “Terminal setup and initialization” on page 102), use the following boot option to configure a serial console port in Linux:

```
console=ttyS0
```

This option can be issued at the boot prompt or appended into the bootloader configuration file. For information about booting, refer to 8.1, “Booting a system” on page 126 and 8.3, “Boot configuration files” on page 132.

6.3.4 Port monitoring

Solaris provides `ttymon` for monitoring and managing ports. Table 6-10 lists and describes the various port monitoring programs available in Linux. Consult the man pages for information about their options.

Table 6-10 Port monitoring programs available for Linux

Port monitoring program	Description
<code>agetty</code>	The default “alternative” <code>getty</code> program for Linux, which includes many nonstandard features as compared to <code>getty</code>
<code>mgetty</code>	The smart “modem” <code>getty</code> for use with modems
<code>mingetty</code>	A “minimal” <code>getty</code> for use only on Linux virtual consoles

Novell SUSE

Novell SUSE also provides the original `getty` program for monitoring ports.

Services administration

Solaris provides the **pmadm** command to enable system administrators to list the services of ports associated with a port monitor, add services, and enable or disable a service.

Linux does not have a similar command; however, you can accomplish the same functionality provided by **pmadm**. Table 6-11 lists the tty service-related tasks in Solaris and how to accomplish them in Linux.

Table 6-11 Solaris versus Linux tty port service commands

Port service task	Solaris	Linux
Add a TTY port service	pmadm -a	Add a service line in <code>/etc/inittab</code> . See man page.
View status on TTY port service	pmadm -l	<pre>cat /etc/inittab ps -ef grep tty ps -ewft cat /proc/tty/drivers</pre>
Enable a TTY port service	pmadm -e	Run telinit q after adding the service to the <code>/etc/inittab</code> file.
Disable a TTY port service	pmadm -d	Run telinit q after removing the service from the <code>/etc/inittab</code> file.

6.4 Distribution-based device management tools

Red Hat and Novell SUSE provide their own interactive tools for managing devices:

► Red Hat

kudzu: This tool detects and configures hardware on a system. For information about using this tool, refer to its man page.

► Novell SUSE

yast2: This system configuration utility supplies modules for configuring common hardware, such as disk controllers and displays. For information about using this tool, refer to its man page.



Network services

This chapter discusses the differences in networking and networking services between Solaris and Linux. We do not describe what TCP/IP is and how to plan for your networking services, because this is the same for both operating systems.

This chapter includes the following topics:

- ▶ IPv4
- ▶ IPv6
- ▶ Mixed IPv4 and IPv6 networks
- ▶ Static and dynamic routing
- ▶ IPSec and IKE
- ▶ Network multipath
- ▶ Network trunking
- ▶ IP network services
- ▶ TCP wrappers
- ▶ IP stateful firewalling

7.1 IPv4

IP version 4 is the default networking stack for all recent operating systems, and this is no exception for Solaris and Linux. However, there are a number of small differences between the two operating systems, and in this chapter, we highlight those differences.

IPv4 in Solaris

The installation program of Solaris will usually set the right networking parameters for you, so that after installation, the system is ready to operate in the network. To set or modify the TCP/IP parameters of a Solaris box after installation, you need to edit several files:

<i>/etc/hostname.interface</i>	File containing IP address or host name for each network interface you need to use.
<i>/etc/netmasks</i>	Contains subnetting information.
<i>/etc/nodename</i>	Contains the host name of the system.
<i>/etc/defaultdomain</i>	Contains the fully qualified domain name of the system.
<i>/etc/defaultrouter</i>	Contains an entry for each router that the server will use.
<i>/etc/inet/hosts</i>	Database contains the local IP address to name mappings. The <i>/etc/hosts</i> is a symlink to this file.
<i>/etc/nsswitch.conf</i>	Contains configuration of network name services.
<i>/etc/resolv.conf</i>	Contains the DNS client configuration of the system.
<i>/etc/bootparams</i>	Contains information for serving network booting services.
<i>/etc/ethers</i>	Contains information about the MAC to IP address mapping used by the rarpd daemon.

There are other files used by IPv4 networking (*/etc/inet/services* or */etc/inet/protocols*), but their default values are fine for most applications and usually do not need any modification.

To use **dhcagent**, the DHCP client for configuring a network interface, you need to create an empty file */etc/dhcp.interface* (such as */etc/dhcp.hme0*). You can alter the default **dhcagent** behavior by editing the */etc/default/dhcagent* configuration file.

In addition, you can use the **sys-unconfig** command and reconfigure your Solaris box after a reboot.

IPv4 in Linux

The installation program of Linux also sets up networking parameters for you, so that after the installation process, the system is ready to operate in the network. To set or modify the TCP/IP parameters after the initial installation, you have the option of editing the configuration files or using a GUI. The GUI program for accomplishing this in RHEL is **system-config-network** and in SLES is **yast2**.

The most important configuration files used by Linux networking services are:

► In RHEL:

/etc/sysconfig/network-scripts/ifcfg-interface-name

This file contains all the information needed for configuring a specific interface, including IP, subnet mask, broadcast, custom MAC address, and gateway, or if the interface is to be configured by DHCP or BOOTP protocol. In addition, you can configure whether the interface should be activated at boot time or later. If you choose to activate it later, you can do so with the **ifup** command. Other general networking options can be specified in the */etc/sysconfig/network* file. For more information about the possible options to configure in these files, refer to the */usr/share/doc/initscripts-version/sysconfig.txt* file and RHEL *System Administration Guide*.

► In SLES:

/etc/sysconfig/network/ifcfg-interface-id-mac-address

This file contains all the information needed for configuring that specific interface, including IP, subnet mask, broadcast, custom MAC address, and gateway, or if the interface is to be configured by DHCP or BOOTP protocol. In addition, you can configure whether the interface should be activated at boot time or later. If you choose to activate it later, you can do so with the **ifup** command. Other general networking options can be specified in the */etc/sysconfig/network/config* file. For more information about the possible options to configure in these files, refer to the SLES *Administration Guide*.

Common network configuration files in Solaris and Linux

There are several networking configuration files that are common to both Linux to the Solaris:

<i>/etc/hosts</i>	This file specifies IP address to name mappings.
<i>/etc/nsswitch.conf</i>	Contains configuration of network name services.
<i>/etc/resolv.conf</i>	Contains the DNS client configuration of the system.
<i>/etc/bootparams</i>	Contains information for serving network booting services only to Solaris systems. Linux clients do not need this service.

/etc/services	Contains TCP/UDP port numbers to service names mapping.
/etc/protocols	Contains IP protocol numbers to protocol names mapping.

Other networking differences

The interface naming convention is different. In Solaris, we use the interface type *enum* (for example, `le0`, `le1`, `hme0`, `hme1`, `ce0`, `ce1`), while in Linux we have only *ethnum* (for example, `eth0`, `eth1`), where *num* is the interface number. Linux network interfaces do not need “plumbing;” they are active from the moment their driver is loaded until the moment we unload the driver from the kernel.

For configuring IP networking parameters in Solaris, use the `ndd` command. In Linux, you directly change the parameters in the `/proc/sys/net` directory with the `sysctl` program, or for a permanent change, modify the `/etc/sysctl.conf` configuration file of the `sysctl` program.

Linux provides for network profiles, which means that we can create several network profiles and change them as necessary. However, they are most useful for mobile workstations and computers, rather than servers, so we do not cover them here.

Task table

Table 7-1 illustrates network commands and configuration files in Solaris and their equivalent in Linux.

Table 7-1 Network commands and configuration files

Task	Solaris	Linux
Configure TCP/IP	Edit the following files: <ul style="list-style-type: none"> ▶ <code>/etc/hostname.*</code> ▶ <code>/etc/inet/*</code> ▶ <code>/etc/defaultrouter</code> ▶ <code>/etc/defaultdomain</code> ▶ <code>/etc/nodename</code> ▶ <code>/etc/netmasks</code> 	RHEL: <code>/etc/sysconfig/network</code> and <code>/etc/sysconfig/networking/*</code> SLES: <code>/etc/sysconfig/network/*</code>
Display interface settings	<code>ifconfig</code>	<code>ip</code> <code>ifconfig</code>
Display interface status and statistics	<code>netstat -i</code>	<code>netstat -i</code> <code>ifconfig</code>
Configure interface	<code>ifconfig</code>	<code>ip</code> <code>ifconfig</code>

Task	Solaris	Linux
Check various network statistics	<code>netstat</code>	<code>netstat</code>
Change resolver	<code>vi /etc/resolv.conf</code>	<code>vi /etc/resolv.conf</code>
Configure name services	<code>vi /etc/nsswitch.conf</code>	<code>vi /etc/nsswitch.conf</code>
Display kernel network parameters	<code>ndd /dev/ip \?</code> <code>ndd /dev/tcp \?</code>	<code>sysctl -a grep ^net</code>
Configure kernel network parameters	<code>ndd</code>	<code>sysctl -w variable=value</code>
Check for network link	<code>ndd</code>	<code>ethtool</code> <code>mii-tool</code>
Rename a network interface	N/A	<code>ip</code>
Check routing table	<code>netstat -r</code>	<code>netstat -r</code> <code>route</code>
Testing for connectivity	<code>ping</code>	<code>ping</code>
Check IP path	<code>traceroute</code>	<code>traceroute</code>
Capturing network packets	<code>snoop</code>	<code>tcpdump</code> <code>tethereal</code> <code>ethereal</code>

Example 7-1 shows sample output from `ifconfig` on Solaris.

Example 7-1 Sample Solaris ifconfig output

```
# ifconfig -a
lo0: flags=1000849<UP,LOOPBACK,RUNNING,MULTICAST,IPv4> mtu 8232 index 1
    inet 127.0.0.1 netmask ff000000
eri0: flags=1000843<UP,BROADCAST,RUNNING,MULTICAST,IPv4> mtu 1500 index 2
    inet 9.3.5.145 netmask fffffff0 broadcast 9.3.5.255
    ether 0:3:ba:29:bf:1
```

Example 7-2 on page 112 shows sample output from `ifconfig` on Linux. Note the additional information provided. If the `-a` option is provided, it also lists all the inactive interfaces. Remember that Linux does not have the *plumb/unplumb* feature to enable and disable a network device.

Example 7-2 Sample Linux ifconfig output

```
# ifconfig
eth0      Link encap:Ethernet  HWaddr 00:02:55:7B:02:B0
          inet addr:9.3.5.11  Bcast:9.3.5.255  Mask:255.255.255.0
          inet6 addr: fe80::202:55ff:fe7b:2b0/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:847931 errors:0 dropped:0 overruns:0 frame:0
          TX packets:211207 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:74462901 (71.0 MiB)  TX bytes:280066144 (267.0 MiB)
          Base address:0x2500 Memory:fbfe0000-fc000000

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:4944 errors:0 dropped:0 overruns:0 frame:0
          TX packets:4944 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:3062806 (2.9 MiB)  TX bytes:3062806 (2.9 MiB)

#
# ifconfig -a
eth0      Link encap:Ethernet  HWaddr 00:02:55:7B:02:B0
          inet addr:9.3.5.11  Bcast:9.3.5.255  Mask:255.255.255.0
          inet6 addr: fe80::202:55ff:fe7b:2b0/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:3569066 errors:0 dropped:0 overruns:0 frame:0
          TX packets:4008490 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:426383647 (406.6 MiB)  TX bytes:1746883088 (1.6 GiB)
          Base address:0x2500 Memory:fbfe0000-fc000000

eth1      Link encap:Ethernet  HWaddr 00:02:55:7B:02:B1
          BROADCAST MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:0 (0.0 b)  TX bytes:0 (0.0 b)
          Base address:0x2540 Memory:fbfc0000-fbfe0000

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:5340 errors:0 dropped:0 overruns:0 frame:0
          TX packets:5340 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:3194582 (3.0 MiB)  TX bytes:3194582 (3.0 MiB)
```

```
sit0      Link encap:IPv6-in-IPv4
          NOARP  MTU:1480  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:0 (0.0 b)  TX bytes:0 (0.0 b)
```

7.2 IPv6

We discuss IPv6 differences in this section.

IPv6 in Solaris

The installation program of Solaris asks if you want to configure IPv6 on that server and set the right networking parameters for you, so that after install, the system is ready to operate in an IPv6 network. To manually configure IPv6 on a Solaris system, you need to **touch** the `/etc/hostname6.interface` file for each IPv6 interface and reboot the system. Optionally, you can add IP to name mappings in the `/etc/inet/ipnodes` file. You can use most of the tools from IPv4 with special switches for IPv6.

IPv6 in Linux

The IPv6 stack is present by default in both RHEL and SLES, and you will have a link-local address on all your network interfaces.

To modify the IPv6 stack configuration, you need to edit the following files:

- ▶ In RHEL:

`/etc/sysconfig/network-scripts/ifcfg-interface-name`

The file contains all the information needed for configuring IPv6 on that specific interface. You can also configure whether the interface should be activated at boot time or later. If you choose to activate it later, you can do so with the **ifup** command. Other general networking options can be specified in the `/etc/sysconfig/network` file. For more information about the possible options to configure in these files, refer to the `/usr/share/doc/initscripts-version/sysconfig.txt` file.

- ▶ In SLES:

/etc/sysconfig/network/ifcfg-interface-id-mac-address

This file contains all the information needed for configuring IPv6 on that specific interface. You can also configure whether the interface should be activated at boot time or later. If you choose to activate it later, you can do so with the **ifup** command. You can specify other general networking options in the */etc/sysconfig/network/config* file.

7.3 Mixed IPv4 and IPv6 networks

Because IPv6 is not yet a widely deployed protocol, we might have IPv6 islands separated by IPv4 routers. In order to link those IPv6 islands, we can use tunnels. Both Solaris and Linux support IPv6 tunneling in IPv4.

Solaris tunneling

In Solaris, you can add tunnel interfaces using the **ifconfig** command or adding options to the */etc/hostname6.interface* file.

Linux tunneling

In Linux, you can create tunnels by adding configuration files for them in the */etc/sysconfig/network* directory. For more information, see the documentation for your Linux distribution.

7.4 Static and dynamic routing

We discuss the use of static and dynamic routes in two major cases: when the server is acting as a network router providing routing services to clients, and when the server is acting like a end-user system and it needs a gateway to communicate with the world.

Routing in Solaris

To configure a Solaris box to act as a router, you need at least two network interfaces on that box. Then, you create a */etc/hostname.interface* file for each network interface installed, specifying the host name or the IP address of that interface. Then, you must enter those host names and IP addresses in the */etc/inet/hosts* file for IPv4 and */etc/inet/ipnodes* for IPv6. If this Solaris router is part of any subnetted network, we add the subnets in the */etc/inet/netmask* file. The startup scripts then determine if they need to start a routing daemon for learning and advertising routes (daemons such as RIP or RDISC) or use static routing.

On an end-user Solaris box, we can control the use of a static default route by editing the `/etc/defaultrouter` file. If the file contains a default gateway, the system uses that default gateway and does not launch a dynamic routing protocol such as RIP or RDISC. If the `/etc/defaultrouter` file is empty, the system tries to use a dynamic gateway first, using `/usr/sbin/in.rdisc` if it exists, and if this method fails, it starts the RIP routing protocol in passive, non-advertising mode.

A machine with only one network interface can be forced to be a router by creating an empty `/etc/gateways` file. A machine with more than one network interfaces can be forced to be a multi-homed host instead of a router by creating an empty `/etc/notrouter` file.

For IPv6 routing, you need to use the `ndpd` and `ripngd` daemons.

Routing in Linux

On a Linux box, regardless of how many network interfaces you have, the routing functionality can be enabled by the `sysctl` program. To enable routing, enter the following command in the `/etc/sysctl.conf` file and then run `sysctl -p`:

```
net.ipv4.ip_forward = 1
```

This configures the kernel to permit IP packet forwarding from one network interface to another one. Also check if the IP firewall service is configured properly for your environment.

By default, static routing is used for both network router and end-user configurations. If a DHCP client is used for configuring the network parameters, the default gateway DHCP option is also considered a static route because it is not dynamically modified during the lease period.

If dynamic routing is required, we have several daemons available in the quagga package:

ripd	For RIP routing protocol
ripngd	For IPv6 RIP routing protocol
ospfd	For OSPF routing protocol
ospf6d	For IPv6 OSPF routing protocol
bgpd	For BGP routing protocol
radvd	For router advertisement daemon for IPv6

All these routing protocol daemons have their own configuration file in the `/etc/` directory.

For more details about the configuration of dynamic routing protocols, consult the documentation of the quagga package and the distribution manuals or visit the Web site at:

<http://www.quagga.net/>

7.5 IPSec and IKE

IPSec and IKE protocols are supported by both Solaris and Linux for VPN connectivity.

We cover the topics about configuring and administering IPSec and IKE features in detail in Chapter 14, “Security and hardening” on page 187.

7.6 Network multipath

In Solaris, there is a feature called IP Network Multipath that enables you to create one or more groups of physical interfaces and use them as one virtual interface.

It provides for:

- ▶ Failure detection
- ▶ Repair detection
- ▶ Outbound load spreading

In Linux, we can implement a similar solution with some subsets of the interface bonding feature. See 7.7, “Network trunking” on page 116 for more information about this feature.

7.7 Network trunking

The interface trunking feature in Solaris is almost equivalent to the interface bonding feature in Linux. In order for this feature to work, you need to properly configure the switch at the other end of the network cables. Cisco switches call this feature EtherChannel.

Tip: Linux bonding has more features than the Solaris trunking, and in some configuration modes, it can be used without switch support for EtherChannel.

Solaris trunking

In Solaris, you need to have a special quad network adapter for creating network trunks. Then, you can use several network interfaces from this adapter to create a trunk interface, which is like a big pipe composed of all the individual interfaces. You assign the IP address only to the trunk interface.

Linux bonding

The Linux bonding driver provides a method for incorporating multiple Ethernet network interfaces into a single “bonded” interface. You can create configuration files for them in the `/etc/sysconfig/network` directory. The bonded interfaces can provide either hot standby or load balancing services. In addition, link integrity monitoring can be performed. The `ifenslave` user-level control program is included in both RHEL and SLES distributions. For more information about configuring a bonded interface, refer to the documentation located in the `/Documentation/networking/bonding.txt` file of the kernel source tree or visit the official project Web site at:

<http://sourceforge.net/projects/bonding>

Tip: If you only need more bandwidth, Linux supports 1 Gb and 10 Gb network interfaces.

Linux bridging

In Linux, you can create interface bridges with any Ethernet network interface by using commands from the `bridge-utils` package. For more information, see the Linux distribution and `bridge-utils` package documentation, the `/Documentation/networking/bridging.txt` file of the kernel source tree, and visit the official project Web site at:

<http://bridge.sourceforge.net>

For an example of using the `bridge-utils` package, refer to “Physical and virtual network bridging” on page 296.

7.8 IP network services

This section contains information about a set of commonly used network services.

7.8.1 inetd-based versus xinetd-based network services

In this section, we describe the differences between `inetd` for Solaris and `xinetd` for Linux.

Solaris inetd server

In Solaris, standard network services, such as Telnet, FTP, Rlogin, TFTP, and others, are started on demand by the **inetd** network daemon through the `/etc/inetd.conf` configuration file. The `/etc/services` configuration file contains the portname to portnumber mappings for network applications that can be used with the **inetd** daemon. You can activate usage logging and TCP wrappers by specifying the following parameters in the `/etc/default/inetd` configuration file:

```
ENABLE_CONNECTION_LOGGING=YES
ENABLE_TCPWRAPPERS=YES
```

All services to be run by the **inetd** daemon are defined in the `/etc/inetd.conf` configuration file.

Linux xinetd server

In Linux, the **xinetd** network daemon is the extended Internet services daemon. The **xinetd** daemon general behavior is controlled by its `/etc/xinetd.conf` configuration file. However, unlike the previous **inetd** daemon, **xinetd** has all the network services defined in separate files, one file per service, in a separate directory. The default directory for services configuration is `/etc/xinetd.d/`, but it can be changed to something else in the `/etc/xinetd.conf` global configuration file.

Here are some basic services and their associated configuration files:

```
Telnet          /etc/xinetd.d/telnet
Remote shell  /etc/xinetd.d/rsh
TFTP          /etc/xinetd.d/tftpd
```

The syntax of the **xinetd** configuration files is also different than **inetd**. Example 7-3 shows an example of a configuration file for Telnet services.

Example 7-3 Example xinetd service configuration

```
service telnet
{
    socket_type = stream
    protocol = tcp
    wait = no
    user = root
    server = /usr/sbin/in.telnetd
    disable =yes
}
```

Note that the default state for an **xinetd** service is “disabled.” With **xinetd**, it is possible to configure specific logging and network access control for each separate service. The network access control feature can be implemented in two

ways: with the built-in support of **xinetd**, or with the standard **tcpd** program from the **tcp-wrappers** package.

If any of the configuration files are modified while **xinetd** is running, we need to send a **SIGHUP** signal to **xinetd** in order to reread its configuration; otherwise, it will still run with the old configuration.

For more information about **xinetd**, check the man page of **xinetd**, **xinetd.conf**, and **xinetd.log**.

Tip: It is not uncommon that services in one system are **inetd** or **xinetd** based, and in other systems are stand-alone services. This depends on your distribution and release level.

7.8.2 DHCP

The Dynamic Host Configuration Protocol (DHCP) server from Solaris and Linux have comparable features:

- ▶ Support for previous BOOTP clients
- ▶ Support for local and remote clients by using DHCP relay services
- ▶ Large network support
- ▶ Network booting information for DHCP clients
- ▶ Support for vendor options
- ▶ DDNS updates

However, the server itself is very different in terms of management scripts and configuration files.

Solaris DHCP server

In Solaris, there are several utilities for controlling the **in.dhcpd** server daemon such as **dhcpconfig**, **dhtadm**, **pntadm**, and the GUI **dhcprmgr**.

The configuration is stored in the **/etc/inet/dhcpsvc.conf** file and some other tables, such as **dhcptab**, **macros**, and so on.

Linux DHCP server

In Linux, the controlling script of the DHCPD server is **/etc/init.d/dhcpd**, which supports the **start**, **stop**, **restart**, and **status** arguments among others. The daemon configuration file is **/etc/sysconfig/dhcpd**, and the database containing all the information is in **/etc/dhcpd.conf**. Leases are kept in **/var/lib/dhcpd.leases**. There is no GUI or tool for configuring DHCP in RHEL; the only way to configure

it is to edit the files as needed and restart the daemon with `/etc/init.d/dhcpd restart`. In SLES, there is a DHCP server configuration module in the YaST2 program.

Important: In order to use the DHCP server with IBM `@server i5` and `@server p5` OpenPower™ firmware, you need to run a different DHCP server than the supplied with SLES9. By default, the SLES9 DHCP server uses Berkeley Software Design, Inc. (BSD) type sockets. This DHCP server always sends a broadcast response that the OpenPower firmware does not understand. The Linux Packet Filter (LPF) type sockets send a unicast response, to which the firmware will reply. Because of this caveat, SLES9 ships with two DHCPD binaries, `dhcpd` and `dhcpd.lpf`. In order to use the LPF DHCP server, you need to change the `DHCPD_BINARY` variable in `/etc/sysconfig/dhcpd` to:

```
DHCPD_BINARY="/usr/sbin/dhcpd.lpf"
```

For more information about DHCP, check the man pages for `dhcpd` and `dhcpd.conf` and the documentation that came with your distribution.

7.8.3 DNS

Both operating systems use BIND as the Domain Name System (DNS) server. Modern Linux distributions are based on BIND 9, so if you are familiar with BIND 8 or later on Solaris, you will be able to configure BIND 9 on Linux.

The main configuration file is the same: `/etc/named.conf` in both Solaris and Linux.

7.8.4 NTP

Linux supports both a client and server Network Time Server (NTP) daemon service. While the configuration files for this service are in `/etc/inetd/*` on Solaris, Linux configures this using the `/etc/ntp.conf` file.

7.8.5 LDAP

Both Solaris and Linux support Lightweight Directory Access Protocol (LDAP) as a client and server.

For information about this topic, see 11.3, “Directory services” on page 163.

7.8.6 NIS and NIS+

Both Solaris and Linux support Network Information Service (NIS) and NIS+ protocols.

For information about this topic, see 11.4, “NIS and NIS+” on page 164.

7.8.7 NFS

The Network File System (NFS) is used on both Solaris and Linux for sharing file systems across a network. Both operating systems can act as servers and as clients, even at the same time.

NFS server setup in Solaris

In Solaris, you add the subdirectory paths for NFS sharing in the `/etc/dfs/dfstab` file, and start the NFS server with `/etc/init.d/nfs.server start`. If you already have an NFS server running and you update the content of the `dfstab` file, you can run the `shareall` command to update the NFS shares without restarting the service.

NFS client setup in Solaris

To mount a shared NFS resource, you can either do it manually with the `mount` command or automatically at boot time by placing an entry in the `/etc/vfstab` file.

NFS server setup in Linux

In Linux, you add the subdirectory paths for NFS sharing in the `/etc/exports` file, and start the NFS server with `/etc/init.d/nfs start` on RHEL and `/etc/init.d/nfsserver start` on SLES. The Linux equivalence of the `share` and `shareall` commands is the `exportfs` command, which performs the same thing as the Solaris counterparts. The syntax of `/etc/dfs/dfstab` in Solaris is also different from the syntax of `/etc/exports` in Linux.

Example 7-4 shows a sample `/etc/exports` file.

Example 7-4 NFS /etc/exports sample

```
/pub (ro,insecure,all_squash)
/home/joe joe_pc(rw,all_squash,anonuid=150,anongid=100)
/usr *.local.domain(ro) @trusted(rw)
/projects proj*.local.domain(rw)
/ master(rw) trusty(rw,no_root_squash)
```

Consult the man page for `exportfs` for information about the options in the `/etc/exports` file.

Both Red Hat (**system-config-nfs**) and SUSE (**yast2**) offer GUI-based NFS server configuration.

NFS client setup in Linux

To mount a shared NFS resource, you can either do it manually with the **mount** command or automatically at the boot time by placing an entry in the `/etc/fstab` file.

Table 7-2 lists the main differences between the Solaris and Linux NFS tasks and configuration files.

Table 7-2 Solaris NFS and Linux NFS differences

Tasks and configuration files	Solaris	Linux
Manually start the NFS server	<code>/etc/init.d/nfs.server start</code>	<code>/etc/init.d/nfs start</code> (in RHEL) <code>/etc/init.d/nfsserver start</code> (in SLES)
Mount a resource on a NFS client	<code>mount -F nfs server://resource /mnt/point</code>	<code>mount server://resource /mnt/point</code>
NFS server general config file	<code>/etc/default/nfs</code>	<code>/etc/sysconfig/nfs</code>
NFS logging options config file	<code>/etc/default/nfslogd</code>	<code>/etc/sysconfig/nfs</code>
Share all exported file systems	<code>shareall</code>	<code>exportfs -a</code>
Share a new file system	<code>share</code>	<code>exportfs -o</code>
Config file of shared file systems	<code>/etc/dfs/dfstab</code>	<code>/etc/exports</code>

7.8.8 Web proxy and cache servers

Solaris offers a follow-on product (Web Proxy Server) for this capability. Linux ships by default the squid package for proxy and Web cache services.

7.8.9 Mail services

Solaris uses sendmail as the mail transfer agent.

RHEL uses sendmail as mail transfer agent as well, so you are already familiar with this MTA. postfix is available as an alternative. SLES uses postfix as the mail transfer agent, but sendmail is available for this distribution as well.

Both Linux distributions have POP3 and IMAP4 servers available by default.

7.9 TCP wrappers

Wrappers (tcpd) functionality is offered by both Solaris and Linux in the same manner. On Linux, the configuration of `/etc/xinetd.conf` (instead of `/etc/inetd.conf`) is required. For more information, refer to 7.8.1, “inetd-based versus xinetd-based network services” on page 117 and 14.3, “Securing and removing services” on page 189.

7.10 IP stateful firewalling

Solaris has the SunScreen™ product for IP firewalling, and Linux has the *iptables*-based firewall. Read 14.13, “Firewalls” on page 202 for more information about firewalls.



Boot and system initialization

This chapter describes the differences in system booting and initialization between Solaris and Linux.

We discuss the following subjects:

- ▶ Booting a system
- ▶ Run levels
- ▶ Boot configuration files
- ▶ Shutting down
- ▶ Network booting

8.1 Booting a system

This section provides information about specific ways to boot a system, different booting sources, and ends with an overview of the booting process.

8.1.1 Booting types

A server can be booted in several ways, depending on the tasks the system administrator needs to perform.

Where the Sun SPARC platform has the OpenPROM firmware to control and initiate the system booting, Linux has several platform-specific boot loaders for performing the task of booting a system. The current default Linux boot loader for the IA32 platform is GRand Unified Bootloader (GRUB). Prior to this, there was Linux Loader (LILO) and LOADLIN.

Note: GRUB is standard on Intel-based and AMD-based servers and is used in combination with BIOS on PC-based hardware. For other platform-specific boot loaders, read Part 3, “IBM eServer platforms” on page 235.

Table 8-1 presents the equivalence between the Solaris and Linux boot processes.

Table 8-1 Boot types

Boot type	Solaris	Linux
Default boot mode	boot Or if auto-boot is set to true, it is automatic. Note: The commands listed in this column assume that you are at the OpenPROM prompt.	Automatic or press Enter at the grub prompt. Note: The commands listed in this column assume that you are at the GRUB prompt.
Single user mode	boot -s	Add -s , single , S , or 1 at the end of boot line.
Reconfiguration mode	boot -r	Not needed.
Verbose boot mode	boot -v	in RHEL, remove rhgb quiet from the boot command line. In SLES, press Esc when booting.
Interactive boot mode	boot -a	Edit grub boot line to specify other kernel and options.

Boot type	Solaris	Linux
Recovery boot mode	boot cdrom boot net	Boot from a rescue CD-ROM or from the network.
Interactive start of services while booting	N/A	Possible, but vendor dependent.
Emergency boot mode	N/A	Add emergency or the -b option at the end of boot line.

8.1.2 Booting sources

Solaris and Linux can both boot from different media types.

Table 8-2 presents the possible media sources for Solaris and Linux.

Table 8-2 Boot sources

Boot source	Solaris	Linux
Hard disk	boot Or automatic if auto-boot is set to true.	Automatic if set in BIOS (on PC-based hardware).
CD-ROM/DVD	boot cdrom	Automatic if set in BIOS, selection at boot time (on PC-based hardware).
Network	boot net	Need DHCP (PXE) and TFTP server configured for network boot.
Floppy and various USB devices	Not supported.	Automatic if BIOS supports booting from these devices (on PC-based hardware).

8.1.3 Booting process overview

Here we describe how a system is started up.

Table 8-3 shows how Solaris and Linux control the booting process.

Table 8-3 Boot process

Boot process	Solaris	Linux
Boot control mechanism	OpenPROM	BIOS to MBR to GRUB
Boot control location	NVRAM	/boot/grub/menu.lst Note: This is platform dependent. Applicable to IA32 systems.

Overview of SPARC Solaris booting process

In Solaris for SPARC, booting a kernel is the task of the OpenPROM firmware, which:

1. Runs POST.
2. Identifies and initializes the proper device for booting.
3. Loads **bootblk**.
4. This then loads and executes **ufsboot**.
5. Then **ufsboot** loads and executes the core kernel.
6. Initializes the core kernel data structures.
7. Loads other kernel modules based on the `/etc/system` file.
8. Finally, starts the `/sbin/init` program that brings up the system based on information in the `/etc/inittab` file.

See the Table 8-4 on page 130 for the kernel location in Solaris.

Overview of Linux for IA32 booting process

On the IA32 platform, the BIOS performs the following tasks:

1. Initializes the system and peripheral devices.
2. Locates and runs the MBR code, which is in the first disk sector from the first BIOS defined disk drive, or can be configured to boot from a CD-ROM, floppy, or network device.

The Master Boot Record (MBR) is unique to Intel Architecture (IA) systems and contains a description of the disk (primary partitions, boot partition) and a little program called boot loader that further loads the operating system.

In Linux for the IA32 platform, the default boot loader is GRand Unified Bootloader (GRUB), which can be used to boot not only Linux systems, but also other operating systems, such as DR-DOS and Microsoft Windows. You can also use GRUB to boot other kernel versions in the same Linux system or a same kernel with different boot parameters.

The configuration file for GRUB is `/boot/grub/menu.lst`, and it tells GRUB where to find the kernel to boot from and what the parameters are for the kernel.

In Linux, the kernel to boot is placed in the `/boot` partition or directory along with the necessary kernel modules for proper booting. These kernel modules along with some early scripts are packed in a special format called the **initrd** image. Check the man page for **initrd** for more information about this file.

The kernel starts initializing devices and loads additional device drivers from the **initrd** image. It then starts the **/sbin/init** program, which brings up the system based on information in the **/etc/inittab** file.

The GRUB boot loader is also interactive, so you can modify what kernel to boot and its parameters before booting the system. This is equivalent to the interactive booting of Solaris with the **boot -a** command, but allows more control.

At the GRUB prompt, you can change the automatic startup and edit the boot option strings. To edit the current boot entry in GRUB, you need to press **e** in Red Hat or **Esc** in SUSE.

When in the GRUB interactive prompt, you can change the boot parameters before the kernel is loaded. Example 8-1 shows some sample boot strings for RHEL.

Example 8-1 Changing boot parameters in GRUB

```
root (hd0,0)
kernel /boot/vmlinuz-2.6.9-11.ELsmp ro root=LABEL=/1 rhgb quiet
initrd
/boot/initrd-2.6.9-11.ELsmp.img
```

For example, if you want a verbose boot for the above boot string, remove **rhgb quiet**.

For more information about GRUB, refer to the Linux man pages or the project Web site at:

<http://www.gnu.org/software/grub/>

Whenever you update the Linux kernel, a new boot entry is created for you in the GRUB configuration file and a new **initrd** image is created to match the new kernel.

If you need to create a custom **initrd** image, refer to the man pages for the **mkinitrd** command.

All kernel modules that can be used for a specific kernel are in the **/lib/modules/`uname -r`/** directory.

Table 8-4 on page 130 shows the major differences between booting in Solaris and Linux.

Table 8-4 Booting process

Booting information	Solaris	Linux
Boot process	<p>Phases:</p> <ol style="list-style-type: none"> 1. OpenPROM: Displays system information, runs POST, loads bootblk, locates ufsboot. 2. Boot programs: bootblk loads and executes the ufsboot. 3. Kernel initialization: ufsboot loads and executes the core kernel, initializes core kernel data structures, loads other kernel modules based on the <code>/etc/system</code> file, starts <code>/sbin/init</code> program. 4. <code>init</code>: Starts other processes based on the <code>/etc/inittab</code> file. 5. <code>/sbin/rcX</code> runs the corresponding <code>/etc/rcX.d</code> scripts to start up other components. 	<p>Phases:</p> <ol style="list-style-type: none"> 1. BIOS: Checks the system and peripheral devices. Locates and runs the Master Boot Record (MBR). 2. MBR loads GRUB (GRand Unified Bootloader). 3. GRUB boots the kernel using information in <code>/boot/grub/menu.lst</code>. 4. Kernel starts initializing devices and loads additional device drivers from <code>initrd</code> image, and then starts <code>init</code> process from <code>/sbin/init</code>. 5. <code>init</code>: Starts other processes based on the <code>/etc/inittab</code> file. 6. <code>/etc/rc.d/rc</code> in RHEL or <code>/etc/init.d/rc</code> in SLES runs the corresponding <code>/etc/rcX.d</code> scripts to start up other components.
Default kernel location	<ul style="list-style-type: none"> ▶ <code>/kernel</code> - Contains common components. ▶ <code>/platform/'platform-name'/kernel</code> - Contains the platform-specific kernel components. ▶ <code>/platform/'hardware-class-name'/kernel</code> - Contains kernel components specific to this hardware class. ▶ <code>/usr/kernel</code> - Contains kernel components common to all platforms within a particular CPU set. 	<p><code>/boot</code> contains the kernels and <code>initrd</code> images.</p>

Booting information	Solaris	Linux
Default kernel modules location	<p>Each of the subdirectories mentioned in the previous cell might contain the following subdirectories:</p> <ul style="list-style-type: none"> ▶ drv - Contains loadable device drivers. ▶ exec - Contains modules that run programs stored in different file formats. ▶ fs - Contains file system modules. ▶ misc - Contains miscellaneous system-related modules. ▶ sched - Contains operating system schedulers. ▶ strmod - Contains loadable System V STREAMS modules. ▶ sys - Contains loadable system calls. ▶ cpu - Contains CPU type-specific modules. ▶ tod - Contains time-of-day hardware modules. ▶ sparcv9 - Contains 64-bit versions of specific modules. 	<p><code>/lib/modules/`uname -r`/</code></p>

8.2 Run levels

The concept of *run levels* is the same in both Solaris and Linux, and both have `/etc/inittab` and `rc.d`-style initialization scripts.

However, there are some differences between the use of run levels in Solaris and Linux, as you can see in Table 8-5 on page 132.

Table 8-5 Run levels

Run level	Solaris	Linux
S or s	Single user level (boot -s). Only some file systems are mounted, as opposed to run level 1.	Minimal single user mode on SUSE. In Red Hat, it is the same as init level 1
0	Shuts down the operating system. Does not attempt to turn off the power. Returns to OpenPROM ok prompt.	Power-down state. Shuts down the operating system and tries to turn power off if supported by the system. Equivalent to Solaris run level 5
1	Administrative single user mode. Can access all available file systems. User logins disabled	Single user mode.
2	Multiuser without NFS resources shared.	Multiuser without NFS resources shared. Text-only login.
3	Multiuser with NFS resources shared (default run level for Solaris). The login is text or graphical, depending on the console capabilities.	Multiuser with NFS resources shared. Text-only login.
4	N/A (alternate multiuser).	N/A (alternate multiuser).
5	Power-down state. Shuts down the operating system and tries to turn power off if supported by the system.	Full multiuser, NFS resources shared, and graphic login with X display manager.
6	Reboot.	Reboot. Same as in Solaris.
emergency	N/A.	Ignore the <code>/etc/inittab</code> file and launch the default shell.

In both the Solaris and Linux operating systems, you can check out the current run level with the **who -r** command, and in Linux, you can additionally use the **runlevel** command for this task.

Note: In Linux, the default run level is 5 (full multiuser with graphic login) unless there is no graphic card available, and then the default run level is 3 (full multiuser with text login).

8.3 Boot configuration files

Differences in how **inittab** and run control files are used are discussed in this section.

8.3.1 /etc/inittab file

The `/etc/inittab` file has the same purpose in both operating systems, that is when the kernel loads the `/sbin/init` program, it reads this file to see what is the default run level and what scripts need to run. After initialization, the information in this file is used whenever the administrator changes the run level of the system.

If you modify the `/etc/inittab` file and want to instruct `init` to reload it, you can do so with the `init q` or `init Q` commands.

The syntax of this file is almost the same in both Solaris and Linux. For more in-depth information about this file, refer to the man page for `inittab`.

Note: The `/etc/inittab` file is configured in Solaris to reach run level 3 by first running the scripts for run level 2 and then the scripts for the run level 3. But in Linux, there is only one set of scripts for each run level, meaning that for run level 3, the `init` program will run only the scripts for run level 3. It does not run the lower run level scripts to get to its intended run level. Pay extra attention to where you manually place an rc script.

8.3.2 Run control files (rc files)

Each run level has a set of scripts associated with it called run control files or rc files.

Note: We also discuss the control of the system services through rc files in detail in 9.3, “Starting and stopping system services” on page 141.

In Solaris, each run level has a master script `/sbin/rcX`, where X is the run level, and an associated directory in `/etc/rcX.d/` where X is again the run level.

When entering a run level, the `init` program starts the corresponding `/sbin/rcX` script, which in turn, executes the associated scripts from `/etc/rcX.d/`.

In Linux, there is one single master script: `/etc/rc.d/rc` in Red Hat and `/etc/init.d/rc` in SUSE. This script the run level to start as an argument.

The scripts directory associated with each run level is `/etc/rcX.d/` in Red Hat and `/etc/init.d/rcX.d` in SUSE, where X is the run level, and usually the scripts in those directories are mostly symbolic links to the `/etc/init.d/` main service scripts.

For instance, in Red Hat, `/etc/rc3.d/S80sendmail` is a symbolic link to the `../init.d/sendmail` script and `/etc/rc1.d/K30sendmail` is a symbolic link to the same

../init.d/sendmail script, and depending on the name of this link, the sendmail script will start or stop the service.

Also new in Linux is the ability to restart or check the status of most of the services by passing the argument **restart** or **status** to the service script in the /etc/init.d directory:

```
# /etc/init.d/sendmail restart
```

And:

```
# /etc/init.d/sendmail status
```

Running the service script without options lists the supported options for that specific script:

```
# /etc/init.d/sendmail
Usage: /etc/init.d/sendmail {start|stop|restart|condrestart|status}
#
```

Tip: Red Hat provides the **service** script in /sbin, which enables you to run service scripts without specifying the file path:

```
# service sendmail status
```

Novell SUSE links many of the /etc/init.d/ service scripts to /usr/sbin/rc* files, which also removes the file path restriction:

```
# rcnfsserver
Usage: /usr/sbin/rcnfsserver
{start|stop|status|try-restart|restart|force-reload|reload}
```

8.3.3 Disabling rc scripts

At times, there is a need to prevent certain services from running.

In Solaris, the suggested method is to rename the starting script by adding a “_” character in front of the name, for example:

```
mv /etc/rcX.d/S30someservice /etc/rcX.d/_S30someservice
```

In Linux, we can use the same method as in Solaris for a temporary change, but the proper way is to use the **chkconfig** command or GUI programs such as **system-config-services** in Red Hat, or the *System/Runlevel Editor* module from **yast2** in SUSE.

For more information about **chkconfig**, refer to its man page or the Red Hat and Novell SUSE documentation.

8.4 Shutting down

Shutting down a system in Solaris and Linux is almost the same.

The **shutdown**, **reboot**, **halt**, **poweroff**, **init**, and **telinit** commands exist in both Solaris and Linux, but there can be minor differences in options, so check the man pages of those commands before using them.

One thing you need to remember is that in Solaris run level 5 is for shutting down the system and turning off the power. In Linux, run level 5 is associated with a full multiuser running system with graphical login.

Moreover, run level 0 in Solaris just brings down the operating system returning to the OpenPROM ok prompt, while in Linux it shuts down the operating system and also turns off the power if possible.

8.5 Network booting

Sometimes it is useful to boot from the network. Both Solaris and Linux support network booting. This section describes how to set up a machine to act as a network boot server and how to boot a Linux system from the network as a client.

Solaris network boot server setup

In Solaris, you need to configure the services shown in Table 8-6.

Table 8-6 Solaris network boot services

Service	Description
rarp	For serving IP address information
bootparams	For boot information
dhcp	Can be used instead of rarp and bootparams for network boot configuration
tftp	For serving the proper kernel from /tftpboot
nfs	For sharing file systems

This can be done manually by editing the configuration files, or automatically by using the **add_install_client** script from the installation CD or DVD.

Linux network boot server setup

In Linux, you need to configure the services shown in Table 8-7 on page 136.

Table 8-7 Linux network boot services

Service	Description
dhcp	For network boot configuration
tftp	For serving the kernel and <code>initrd</code> image to boot
nfs	For sharing file systems

Again, you can execute this task by modifying the configuration files or by using a GUI. The GUI is called **system-config-netboot** in RHEL, and in SLES, you can use the *Network Services* module from **yast2** to configure dhcp/tftp/nfs services.

You might need to manually configure support for network booting. Table 8-8 provides a list of the configuration files for controlling the server processes.

Table 8-8 Linux network boot configuration files

File	Description
/etc/dhcpd.conf	For DHCP services
/etc/xinetd.d/tftp	For TFTP services
/etc/exports	For NFS services

Then, make sure that the services are restarted and the firewall policy is not blocking any of the needed protocols.

Network boot services differences

Table 8-9 lists the differences between services needed for booting a Solaris system and those needed for booting a Linux system.

Table 8-9 Network boot services

Services for network boot	Solaris	Linux
RARP	Needed if not using DHCP.	Not needed; uses DHCP.
Bootparams	Needed if not using DHCP.	Not needed; uses DHCP.
DHCP	Not needed if using RARP and Bootparams.	Needed for client network configuration.
TFTP	Needed for serving kernels.	Needed for serving kernels and <code>initrd</code> images.

Services for network boot	Solaris	Linux
NFS	Needed for remote mounting of file systems.	Not needed if the <code>initrd</code> image is enough or using other means for network file access.

Solaris network boot client

On Solaris, it is usually as simple as typing `boot net` at the OpenPROM `ok` prompt, but there are some exceptions. When the server was configured with the `add_install_client` script, the server is actually configured for serving a network install to the client that was specified. If this is not what you want, you need to boot with `boot net -s` to boot from the network in single user mode. In addition, if the server is configured for DHCP instead of RARP+Bootparams, you need to configure OpenPROM to use the DHCP protocol.

Linux network boot client

For booting an Intel-based Linux client from the network, determine if your network card has support for PXE booting. In addition, check the BIOS or system firmware to make sure that the network boot support is enabled. After the system is enabled to boot from the network, it automatically searches for a DHCP server, and then the DHCP server assigns an IP address and transfers a network file to boot from a TFTP server. This network boot file loads and presents a booting menu for choosing a specific kernel to boot, or it can automatically load and boot a default kernel, depending on the configuration.

Tip: It is also possible to boot a Solaris client from a Linux server or a Linux client from a Solaris server, but this setup is possible only with manual configuration. For more information, read 3.3, “Network-based install: Solaris and Linux heterogeneous environments” on page 47.



Managing system resources

This chapter describes the differences in managing system resources from Solaris to Linux. We describe these differences in the system resources in context with the tasks and commands used to control them.

In this chapter, we discuss the following topics:

- ▶ Displaying system information
- ▶ Resource management
- ▶ Starting and stopping system services
- ▶ Scheduling and cron services
- ▶ Quotas
- ▶ Process accounting
- ▶ Remote system management services

9.1 Displaying system information

Table 9-1 shows some commands used to gather and report system information. On Linux, you can also use GUI tools to view this information. On Red Hat, you can use the **hwbrowser** command, and on Novell SUSE, use the **yast** command to launch the GUI tools.

Task table

Table 9-1 illustrates system information commands.

Table 9-1 System information commands

System information task	Solaris	Linux
List system information	<code>uname</code>	<code>uname</code>
List processor information	<code>prinfo</code>	<code>cat /proc/cpuinfo</code>
List system memory size	<code>prtconf grep Memory</code>	<code>cat /proc/meminfo grep MemTotal</code>
List disk space usage	<code>df</code>	<code>df</code>
List file space usage	<code>du</code>	<code>du</code>
List physical disks	<code>format</code>	<code>fdisk -l</code>
Show the system's host name	<code>hostname</code>	<code>hostname</code>
List system's network configuration	<code>ifconfig</code>	<code>ifconfig</code>
List active processes	<code>prstat</code>	<code>ps -e</code>
List system configuration	<code>prtconf</code>	In Red Hat: <code>lshw</code> In Novell SUSE: <code>hwinfo</code>
List system diagnostic information	<code>prtdiag</code>	In Red Hat: <code>lspci</code> and <code>/proc/*</code> files In Novell SUSE: <code>hwinfo</code>

9.2 Resource management

In this section, we discuss optional tools for resource management.

Solaris Resource Manager

Solaris Resource Manager is a tool that helps the Solaris system administrator to manage the system resources for better system utilization and resource

availability. If this is a product that you have used on Solaris systems, you might consider using the Class-based Kernel Resource Management framework in Linux, or IBM Tivoli® Workload Scheduler.

Linux CKRM

Linux Class-based Kernel Resource Management (CKRM) is a framework for allocating system resources (such as CPU, memory, I/O, and network) based on user-defined classifications of work. CKRM gives a user-level application the ability to allocate hardware resources to applications as defined by the system administrator. CKRM is currently shipped with SUSE SLES9.

For more information about how to use CKRM, visit the CKRM project Web site:

<http://ckrm.sourceforge.net/>

Tivoli Workload Scheduler

IBM Tivoli Workload Scheduler is an automated workload management solution that helps manage and operate mission-critical applications within a secure, fault-tolerant, and scalable IT infrastructure. It analyzes the status of the production work and drives the processing of the workload according to your business policies. It supports a multiple-end-user environment, enabling you to distribute processing and control across sites and departments within your enterprise. For more information, see:

<http://www.ibm.com/software/tivoli/sw-bycategory/>

9.3 Starting and stopping system services

System services are defined as processes or daemons that usually run in the background, are not associated with any user login, and do have a user interface.

9.3.1 Solaris system services

In Solaris, the `/etc/rc*.d/` directory contains all the Solaris system startup and shutdown scripts, some of which are soft links to the actual file that resides in `/etc/init.d/`. Startup script names start with an “S” and shutdown scripts start with a “K”. To disable a service from running at startup time, a script is simply renamed so that it does not begin with a capital S. To manually start or stop a service, use the `/etc/init.d/service {start, stop}` commands.

Network services and inetd

Both Solaris and Linux configure the network service ports through the `/etc/services` file. Solaris uses the `/etc/inetd.conf` file to configure the Internet `inetd` processes.

9.3.2 Linux system services

On Linux, you can use the commands listed in this section, or use the user interfaces to manage system services. On Red Hat, use the `system-config-services` command to launch the GUI. On Novell SUSE, use the `yast` command.

The `/etc/init.d` directory is the default location for system service startup scripts used to start and stop the service. The scripts listed there are named the same as the system service. Most startup scripts at least provide start, restart, and stop functionality and usually support a status option, for example:

```
/etc/init.d/dhcpd start
/etc/init.d/dhcpd stop
/etc/init.d/dhcpd restart
/etc/init.d/dhcpd status
```

SUSE provides a set of soft links to these scripts for managing system resources. The linked files are `/usr/sbin/rc*`. These files correspond to the scripts in `/etc/init.d`. You can use either of these commands to check the status of the `cron` daemon:

```
/etc/init.d/cron status
rccron status (in path from /usr/sbin)
```

Red Hat provides the `service` command. You can use either of these commands to check the status of the `cron` daemon:

```
/etc/init.d/crond status
service crond status
```

Another useful version of the `service` command is:

```
service --status-all
```

Example 9-1 provides a sample output of the previous command.

Example 9-1 Sample output from `service --status-all` (shortened)

```
# service --status-all
acpid (pid 2242) is running...
amd is stopped
..
..
```

```
xinetd (pid 2319) is running...
ypbind is stopped
rpc.yppasswdd is stopped
ypserv is stopped
```

Any of the previous services can be started or stopped by issuing the command:

```
/sbin/service smbd start
```

Or:

```
/sbin/service smbd start
```

Note that the **service** command is only available for Red Hat. The **chkconfig** command has the same function as the **service** command and is available on both Red Hat and SUSE.

The **chkconfig** command

The **chkconfig** command is used to add and enable system services to run at boot time. The **chkconfig** command exists and performs the same basic function on both Red Hat and Novell SUSE, but behaves slightly differently. For more information about the **chkconfig** command, refer to the man page. Example 9-2 shows a sample output of a **chkconfig --list** command.

*Example 9-2 Sample output of **chkconfig --list** command (shortened)*

```
# chkconfig --list
kudzu          0:off 1:off 2:off 3:on 4:on 5:on 6:off
dhcp6s        0:off 1:off 2:off 3:off 4:off 5:off 6:off
vsftpd        0:off 1:off 2:off 3:off 4:off 5:off 6:off
anacron       0:off 1:off 2:on 3:on 4:on 5:on 6:off
..
..
diskdump      0:off 1:off 2:off 3:off 4:off 5:off 6:off
inn           0:off 1:off 2:off 3:off 4:off 5:off 6:off
ntpd          0:off 1:off 2:off 3:off 4:off 5:off 6:off
ypxfrd        0:off 1:off 2:off 3:off 4:off 5:off 6:off
cyrus-imapd   0:off 1:off 2:off 3:off 4:off 5:off 6:off
ypbind        0:off 1:off 2:off 3:off 4:off 5:off 6:off
xinetd based services:
  time:  off
  tftp:  on
  ..
  ..
  amanda: off
  rlogin: off
  chargen-udp:  off
```

Note the different run levels for which the services are being enabled. Keep in mind that, in Linux, unlike Solaris, the run level is not run from the lowest to the desired run level, but instead, it only runs the startup scripts specified at the desired run level.

Network services and inetd

Linux uses the **xinetd** service and the `/etc/xinetd.conf` and `/etc/xinetd.d` directory entries for configuring the Internet processes.

For more information about these services and configuration differences, refer to Chapter 7, “Network services” on page 107.

Task table

Table 9-2 illustrates system service management commands.

Table 9-2 Summary of system service commands

Task	Solaris	Red Hat	SUSE
Add a service that does not already exist in <code>chkconfig</code>	Copy start script into the appropriate run level directory, prefixed with <code>S</code>	<code>chkconfig --add service name</code>	<code>chkconfig -a or --add service name</code>
Show existing service links	<code>ls</code> run level directory	<code>chkconfig --list grep service name</code>	<code>chkconfig service name</code>
Enable service to start at boot time	Copy start script into the appropriate run level directory, prefixed with <code>S</code>	<code>chkconfig service name on</code>	<code>chkconfig -s or --set service name on</code>
Disable service from running at boot time	Rename the service start script to a name that does not start with <code>S</code> (typically lowercase <code>s</code>)	<code>chkconfig service name off</code>	<code>chkconfig -s or --set service name off</code>

9.4 Scheduling and cron services

This section discusses the command scheduling services and their differences. Scheduling involves being able to run commands at predefined times.

The **at** command enables a command to run at a predefined time. The **batch** command works the same way, except that it waits until system load levels permit.

The **cron** command allows for a command to be run at a predefined time, but offers much more flexibility than the **at** command.

Both Solaris and Linux handle the two scheduling methods the same way.

The **anacron** command is unique to Linux. This command is used just like **cron**, except that it allows for those scheduled commands that were not run due to the system being powered off or in suspended state to be run after the system is up and running normally again, sort of like a catch-up mode. This can be very useful in the case of desktops and notebooks that are sometimes powered off.

The **cron** and **anacron** services use tables to schedule the jobs to run. For more information about configuring commands to run, refer to the man pages.

Task table

Table 9-3 illustrates scheduling and **cron** management commands.

Table 9-3 Scheduling and cron service commands

Task	Solaris	Linux
Run a command at a predefined time	at	at
Run a command or set of commands periodically	cron and crontab	cron and crontab
Run a command or set of commands periodically, running when possible if system is off at the scheduled time	N/A	anacron and anacrontab

9.5 Quotas

This section describes the differences in the **quota** commands. The **quota** commands are for managing disk usage and limits.

The **quota** commands on Linux are very similar to those on Solaris; however, there are subtle differences in the command line parameters available. Check the man pages for further details about the differences.

Task table

Table 9-4 on page 146 illustrates **quota** commands.

Table 9-4 Quota commands

Quota task	Solaris	Linux
Display disk usage and limits	<code>quota</code>	<code>quota</code>
Edit users quotas	<code>edquota</code>	<code>edquota</code>
Check consistency between a file system and the disk quota file	<code>quotacheck</code>	<code>quotacheck</code>

9.6 Process accounting

This section describes the differences in the process accounting services. Process accounting is the method of keeping records on what commands are run and the user executing the command, as well as system resources used.

9.6.1 Solaris

The commands on Solaris are very similar to those available on Linux with the exception of the `ac` command. There is not an equivalent command with similar functionality.

9.6.2 Linux

On Linux, the commands differ somewhat from the commands on Solaris. The `accton` and `lastcomm` commands are very similar to those found on Solaris. Refer to the man pages for any differences in these commands.

Task table

Table 9-5 illustrates package commands.

Table 9-5 Process accounting commands

Process accounting task	Solaris	Linux
Turn on process accounting	<code>accton</code>	<code>accton</code>
List user's connection time	<code>none exists</code>	<code>ac</code>
List commands run previously	<code>lastcomm</code>	<code>lastcomm</code>
List user's connection time and idle time	<code>who</code>	<code>who</code>

9.7 Remote system management services

This section describes the differences in the common remote system management services between Solaris and Linux. You might or might not have used these services on Solaris, but you might want to consider their use when you migrate to Linux. These system management services enable a server or set of servers to be configured and monitored remotely using several different optionally available management frameworks or applications.

Web-Based Enterprise Management (WBEM) and the Common Information Model (CIM) are the object-based architecture standards and data model promoted by the industry consortia Distributed Management Task Force (DMTF). The WBEM services described here are implementations of these standards. For more information about these standards, refer to the DMTF Web site:

<http://www.dmtf.org>

Another commonly used remote system management service is the Simple Network Management Protocol (SNMP). SNMP is an Internet Engineering Task Force (IETF) standard and is perhaps the most widely implemented standard for system management. The data model used by an SNMP service is called a Management Information Base (MIB). Different sections of this MIB are supported by different devices and servers depending on the services available on the system. For more information about the SNMP service, refer to the IETF Web site:

<http://www.ietf.org>

9.7.1 Web-Based Enterprise Management (WBEM)

WBEM services are made available for use by a system management application through a CIM object manager or CIM server. Providers are written that gather and manage system resources. These providers register their classes with the CIM server to enable these objects to be used by a management application or even other providers. There are increasing numbers of applications adopting this new management standard. For more information about the components in the WBEM architecture, refer to the DMTF Web site:

<http://www.dmtf.org/standards/wbem>

Solaris

Some of the Solaris commands for the WBEM services include:

init.wbem	Used to start, stop, or retrieve status from the CIM server.
wbemadmin	Sun WBEM user manager.
wbemlogviewer	WBEM log viewer GUI.

The CIM server on the Sun Solaris platform is managed with the `inet.wbem` command. The `inet.wbem` command is in the `/etc/initd` directory. The `init.wbem` command accepts the `start`, `stop`, and `status` parameters.

Red Hat

Starting in RHEL4 Update 2, the OpenPegasus CIM server is available. For more information about the OpenPegasus CIM Server, refer to the OpenPegasus Web site:

<http://www.openpegasus.org>

Additionally, you can read more about how to set up and use the service by reviewing the documents in the `/usr/share/doc/tog-pegasus-*` directory.

Some of the commands available for the OpenPegasus WBEM services include:

cimconfig	Provides command line interface to manage CIM server configuration properties.
cimserver	Starts and stops the CIM server.
cimauth	Adds, modifies, removes, or lists CIM user authorizations.
wbemexec	Provides command line interface for querying CIM objects.
cimprovider	Disables, enables, removes, and lists registered CIM providers or CIM provider modules and module status.

OpenPegasus is started and stopped with the `cimserver` command in `/usr/sbin`, or with the `tog-pegasus` command in `/etc/inet.d`. These commands can only be executed as a privileged user. The `cimserver` command starts the service, and using the `-s` parameter, stops the service. The `osinfo` command is useful for checking the status of the `cimserver` process and it also reports the system information. For more information about these and other commands, refer to the man pages.

To enable the service to start at run time, run the `chkconfig` commands:

```
chkconfig --add tog-pegasus
chkconfig tog-pegasus on
```

Then to start the service, use the `tog-pegasus` command in `/etc/init.d`:

```
/etc/init.d/tog-pegasus start
```

Or run the `cimserver` command.

There are a very basic set of providers that are included with the OpenPegasus package. To make the service more useful, a more extensive set of

instrumentation for Linux is available through the Standards Based Linux Instrumentation for Manageability (SBLIM) project. For more information about the SBLIM project, refer to the project Web site:

<http://sblim.sourceforge.net>

Novell SUSE

On SUSE, the OpenPegasus CIM server is also available, but it is on the SDK CD and must be installed separately. The default CIM server for SUSE is the OpenWBEM CIM server. For more information about the OpenWBEM CIM server, refer to the OpenWBEM Web site:

<http://www.openwbem.org>

In addition, there is documentation about how to configure and use the service in the `/usr/share/doc/packages/openwbem` directory.

Some of the commands available for the OpenWBEM service include:

owcimomd	OpenWBEM CIM server.
rcowcimomd	Starts, stops, and queries status of CIM server.
owenumclasses	Lists CIM classes.

The OpenWBEM service can be started by using the **owcimomd** command in `/usr/sbin`. More information about these and other commands can be learned from the man pages.

To start the service and to enable it to start at run time, run the **chkconfig** command:

```
chkconfig owcimomd on
```

The more extensive set of instrumentation for Linux referenced earlier, the SBLIM providers, are included in the Novell SUSE release. You can install these providers to greatly expand the amount of management information available.

9.7.2 Simple Network Management Protocol (SNMP)

SNMP services are implemented through a background server or daemon. This daemon listens for requests, and if the object is supported in the set of MIB extensions maintained by the server, the action is performed and the response returned. The SNMP service also has the ability to send and receive system events called traps. And, with the deployment and wide spread adoption of secure SNMPv3, it is now even safer to use.

Solaris

The Solaris SNMP service is started at boot time through the startup script `/etc/rc3.d/S76snmpdx`. To start and stop the service manually, run the script with the **start** and **stop** command line parameters:

```
/etc/rc3.d/S76snmpdx start  
/etc/rc3.d/S76snmpdx stop
```

The configuration file, `snmpd.conf`, is in `/etc/snmp` subdirectory for Red Hat, and in `/etc` for Novell SUSE.

Linux

Both Red Hat and Novell SUSE include the Net-SNMP package as their default SNMP service. For more information about the Net-SNMP package, refer to the Net-SNMP project page:

<http://www.net-snmp.org/>

You can also read more about Net-SNMP in the documents in the `/usr/share/doc/packages/net-snmp` directory. There are some excellent SNMP tutorials available at:

<http://www.simpleweb.org/>

Starting the Linux SNMP daemon is done either by running the **snmpd** command or enabling the service to start at boot time. There are many command line options, but usually, if started from the command line, the service is just simply started with the default command line options by running the **snmpd** command. For more information about the **snmpd** command, review the man page.

To enable the service to start at run time, run the **chkconfig** command:

```
chkconfig snmpd on
```

To start the SNMP service:

```
/etc/inet.d/snmpd start
```

To configure the SNMP service, you need to edit the `/etc/snmpd.conf` file. The file has basic instructions for configuration, but you might want to refer to the man page for **snmpd.conf** as well.

To make the SNMP service secure, you need to configure SNMPv3. A full tutorial about how to set this up is beyond the scope of this book. For more information, refer to the SNMPv3 *readme* in the `/usr/share/doc/packages/net-snmp` directory.



Printing services

This chapter discusses commonalities and differences in printer management between Sun Solaris and Linux.

In this chapter, we discuss the following topics in detail:

- ▶ Linux CUPS
- ▶ Client and server setup
- ▶ Printer management using the lp commands
- ▶ Printer types and CUPS support

10.1 Overview

The printing subsystem on Linux is, on the surface, similar to the one on Solaris, supporting all the familiar BSD and System V UNIX-based “lp” command line utilities. Underneath, however, is the Common UNIX Printing System (CUPS), which is based on the new Internet Printing Protocol (IPP) standard. IPP has been adopted by many printer and print server makers, and was added by Microsoft into its Microsoft Windows 2000 product. While CUPS also supports the widespread UNIX-based LPD protocol, it is worthwhile to understand the Linux standard mechanism.

10.2 Linux CUPS

The Common UNIX Printing System (CUPS) is the standard for the Linux print service. It supports the Internet Printing Protocol (IPP) as its primary transport, but supports the line printer daemon (LPD) (client and server) as well. CUPS offers the following advantages over the earlier UNIX LPD/LPR mechanism:

- ▶ Individual setting of print parameters at user level
- ▶ Better support for PostScript printers
- ▶ Network “browsing”
- ▶ Web-based user and administrator front ends

IPP defines a standard protocol for printing, as well as managing print jobs and printer options such as media size, resolution, and so forth. IPP also supports access control, authentication, and encryption. IPP is layered on top of HTTP. This enables users to view documentation, check status information about a printer or server, and manage their printers, classes, and jobs using their Web browser.

CUPS supports the following protocols for network printing: IPP (preferred default), LPD (RFC 1179), SMB (Windows share), and socket.

While we recommend using the CUPS guides provided by your Linux distribution first, for complete and detailed CUPS documentation and tutorials for both system administrators and users, refer to:

<http://www.cups.org/>
<http://www.linuxprinting.org/>

10.3 Client and server setup

CUPS provides two methods for adding printers: `lpadmin` and a Web interface. The `lpadmin` command enables you to perform most printer administration tasks from the command line and is located in `/usr/sbin`. The Web interface is located at `http://localhost:631/admin` and steps you through the printer configuration. The IPP protocol is based on HTTP, and thus CUPS Web management can be done without the requirement of an HTTP server. The CUPS Web interface allows for full networked and remote printer management, after a CUPS administrator (who does not have to be root) is assigned using the `lppasswd` command.

Additionally, your Linux distribution will usually provide a third way, a printer administration GUI (a `yast2` module in SUSE, `system-config-printer-gui` in Red Hat). Both SUSE and Red Hat recommend that you attach and configure printers using their GUIs; you should follow your distribution's recommendation until you have experience in using CUPS.

Locally attached printers

Linux CUPS supports printers attached through serial (including USB) or parallel ports. They can be added through all three interfaces. Follow the manufacturer's instructions for installation, including installing any necessary device drivers. Use of the parallel port might require special configuration; consult the CUPS guide that came with your distribution.

Restriction: For zSeries (S/390®): Printers provided by zVM that you can connect locally with the zSeries hardware platform are not supported by CUPS. On these platforms, network printing is required.

Network printing

Linux CUPS supports networked printing through IPP, LPD (RFC 1179), SMB (Windows share), or direct socket. Configuration is possible for all through the command line, Web, or GUI. IPP is the preferred method between two CUPS servers or between the CUPS server and client.

CUPS also supports printers attached through Novell NetWare technology and HP JetDirect system.

10.4 Printer management using the lp commands

The basic BSD and System V UNIX-based command line utilities are available on Linux as well as Solaris OS:

lp	Print files
lpadmin	Configure the printer services and classes
lpc	Line printer control
lpr	Submit print requests
lpstat	Display print service status information
lpq	Display print queue status
lprm	Cancel print jobs
lpmove	Move print requests
enable	Start the named printers
disable	Stop the named printers
accept	Accept print jobs for the specified destination
reject	Reject print jobs for the specified destination
cancel	Cancel print request

While the basic option/flag sets of these commands are the same, there are differences in details based on how CUPS manages printers.

Solaris specific

Solaris provides some additional utilities to manage the print services, which are not available on Linux. They are as follows:

lpsched	Start the LP print service daemon
lpset	Set printing configuration in /etc/printers.conf
lpshut	Stop the lp print service
lpssystem	Register remote systems with print service
lpusers	Set printing queue priorities
lpforms	Administer forms used with the lp service
lpget	Get printing configuration
lpfilter	Administer filters used with the lp service

Linux specific

Linux also provides some additional functions and capability to manage the print services. They are as follows:

cupsd	Start the common UNIX printing system daemon
lpinfo	Show available devices or drivers
lpoptions	Display or set printer options and defaults
lppasswd	Add, change, or delete digest password in CUPS

Important: Most Linux distribution vendors advise against configuring the printing system through the direct editing of the configuration files. The CUPS service dynamically builds the configuration files each time it is started, and changes will be lost. The `/etc/printcap` file (for example) is created only for the purpose of compatibility with **printcap**-based applications.

Task table

Table 10-1 organizes the command comparison by administrator task.

Table 10-1 Print service management task commands

Task	Solaris	Red Hat	SUSE
Run multiple tasks in a GUI environment	admintool	system-config-printer-gui	yast2
Add a printer	lpadmin	lpadmin	lpadmin
Start a print queue	accept, enable, or lpc	accept, enable, or lpc	accept, enable, or lpc
Stop a print queue	reject, disable, or lpc	reject, disable, or lpc	reject, disable, or lpc
Display print queue status	lpstat or lpc	lpc or lpq	lpc or lpq
Cancel a print job	cancel	cancel or lprm	cancel or lprm
Add a print queue	lpadmin	lpadmin	lpadmin
Change a print queue	lpadmin	lpadmin	lpadmin
Remove a print queue	lpadmin	lpadmin	lpadmin
Display settings of a print queue	lpadmin	lpadmin or lpoptions	lpadmin or lpoptions
Set print queue settings	lpadmin	lpoptions	lpoptions
Print files	lp or lpr	lp or lpr	lp or lpr

Task	Solaris	Red Hat	SUSE
Display print service status information	lpstat	lpstat	lpstat
Move the print request to another print queue	lpmove	lpmove	lpmove
Print services daemon	lpsched	cupsd	cupsd
Start and stop script for the print services daemon	/etc/init.d/lp or lpshut	/etc/rc.d/cups	/etc/rc.d/init.d/cups
Put printing config information in the system config database	lpset	N/A	N/A
Display printing config information from the system config database	lpget	N/A	N/A
Register remote print queues in local system	lpssystem	N/A	N/A
Set printing queue priorities for users	lpusers	N/A	N/A
Configure forms for the print queue	lpforms	N/A	N/A
Apply filters to the print queue	lpfilter	N/A	N/A
Display available devices or drivers	N/A	lpinfo	lpinfo
Manage digest password in CUPS	N/A	lppasswd	lppasswd

10.5 Printer types and CUPS support

There are three classes of printers to consider, when matching a printer to a Linux/CUPS system, and the division is by language.

PostScript

CUPS supports these printers by default, using either supplied PostScript Printer Description (PPD) files for your brand and model of printer, or those available from your printer manufacturer, or from places such as:

<http://www.linuxprinting.org/>

Standard printer languages (PCL, ESC/P, or similar)

There are many printers, laser as well inkjet, that use these languages for processing. The PCL language is mostly used in certain lines of HP printers (and their clones). ESC/P is used by some Epson models. Linux can usually support these printers well through a filtering process known as *interpreting* (usually by using Ghostscript tools, which come with Linux). There are very few native Linux drivers for these printers, with the exception of HP products.

CUPS comes with special filters, known as **foomatic** (in the cups-drivers RPM package) and **rastertoprinter** (the cups-drivers-stp RPM package) to handle these printers if they do not have their own drivers. Consult the CUPS section of the *Help Guide* that comes with your Linux distribution.

Proprietary

A few of the least expensive printers use proprietary communication models and come with Windows-only drivers. Lexmark inkjet printers are a common example of this. These are usually poorly-supported by Linux, if they are supported at all. You might find help for some models at:

<http://www.linuxprinting.org/>



Users and groups

This chapter discusses basic user and group administration tasks and utilities, including:

- ▶ Commands
- ▶ Directory services
- ▶ NIS and NIS+
- ▶ User ID and group ID differences

11.1 Basic user administration

In general, user and group administration is the same as on most UNIX-based systems, including Solaris. Configuration files, such as `/etc/group`, are in the same locations and are used in the same manner. Most user administration tasks can be performed either from the command line or from your Linux distribution's GUI (for example `yast2` on SUSE and `system-config-users` on Red Hat).

Red Hat starts regular user IDs at 500; SUSE Linux starts them at 1000.

11.2 Commands

The following command line-based user ID administration tools have very similar basic functionality between Solaris and Linux:

- ▶ `useradd`
- ▶ `usermod`
- ▶ `userdel`

For these command line-based tools, functions such as adding comments, setting home directory path, base group ID, and other group IDs, creating a home directory, using skeleton profiles, and the default shell settings are the same between Solaris and Linux.

11.2.1 Adding user accounts

The key differences for the `useradd` command are as follows:

- ▶ Solaris:
 - P and -R options** Add RBAC profile and role in the `useradd` command.
 - p option** Assign a project name to the user.
 - A option** Assign an authorization attribute for executing restricted functions.
- ▶ Red Hat:
 - p option** Assign a user passwd when adding the user account.
 - r option** Assign a UID that is lower than the `UID_MIN` set in `/etc/login.defs`, which can make the user ID a system account.
 - M option** The `useradd` command creates a user home directory as per system-wide the configuration parameter set in

- /etc/login.defs. By giving it this parameter, it overwrites that option to not create it.
- n option** The **useradd** command creates a group name to the same user name by default. By giving this parameter, it will not create the group name with the same name as the user name.
 - l option** Not to add the user to the last login log.
 - ▶ **SUSE:**
 - p option** Assign a user passwd when adding the user account.
 - r option** Assign a UID that is lower than the UID_MIN set in /etc/login.defs, which can make the user ID a system account.
 - D or --binddn <binddn> option** Add the user using the Distinguished Name **binddn** to bind to the LDAP directory.
 - P or --path <path> option** Add the user ID in a different path for the passwd and shadow files.
 - service option** Add the user account using the special service, which can be a local file, LDAP, NIS, or NIS+.
 - usage option** Display a short list of valid options.
 - show-defaults option** Display the default settings for the user parameters: homedir, expiry, inactivity, gid, other groups, and default shell.
 - save-defaults options** Change the default settings for the user parameters.

Note: Both Solaris and Red Hat use the **-D** option to set and display the user parameter defaults. However, SUSE Linux uses the **-D** option for a different purpose.

11.2.2 Changing user account information

The differences between Solaris, Red Hat and SUSE for the **usermod** command are the same as the previous list, although some of the previous options are specific to the **useradd** command. The **usermod** utility can change most of the

option settings defined in the previous list. The command differences are as follows:

► Solaris:

The **usermod** command enables you to change most of the parameters that are set by the **useradd** command.

► Red Hat:

This also enables you to change most of the parameters that are set by the **useradd** command, with the following available options:

-L option Lock the user account.

-U option Unlock the user account.

► SUSE:

The **usermod** does not allow the changes to any of the basic user account parameter except for the following changes:

-D <binddn> option Change the Distinguished Name **binddn**.

-P <path> option Change the path for the passwd and shadow files.

-p <new password> option
Change the user account's password.

-r <service> option Change the account's special service procedure.
Either local files, LDAP, NIS, or NIS+.

11.2.3 Removing user accounts

There are no differences between Solaris and Red Hat for the **userdel** command. The **-r** option works the same way on both. However, SUSE does have differences, including:

-D <binddn> or --binddn <binddn> option
Add or remove the user account using the Distinguished Name **binddn** to the LDAP directory.

-P <path> or --path <path> option
Remove the user account from the specified directory to passwd and shadow file.

--service <service> option
Remove the user account using the special service. Either local files, LDAP, NIS, or NIS+.

-f option
Force the removal of files in the user's home directory even if the files are not owned by the account.

11.2.4 Home directory

Linux creates user home directories by default into /home. It has a location for creating a “default user profile,” a set of preconfigured files (such as .profile or X11 desktop files) used to populate a newly created user’s home directory called /etc/skel/.

11.3 Directory services

The Solaris version of LDAP uses the iPlanet™ Directory Server. Solaris 8 uses the 4.11 version and Solaris 9 uses the 5.1 version. There is also the more recent Sun One Directory Server (5.2). Both Red Hat and SUSE Linux use OpenLDAP (<http://www.OpenLDAP.org/>), a software package founded with support from IBM. This package provides both a stand-alone and update replication daemon, as well as libraries, utilities, Java classes, and sample clients.

OpenLDAP supports LDAPv3 (RFC 3377) with the addition of the following optional features:

- ▶ Schema updates over LDAP through cn=config (2.3 only)
- ▶ Component Matching (RFC 3687) (2.3 only)
- ▶ Language Tag and Range options (RFC 3866)
- ▶ DNS-based service location (RFC 2247 & RFC 3088)
- ▶ Password Modify operation (RFC 3062)
- ▶ Named Referrals/ManageDSAit control (RFC 3296)
- ▶ Simple Paged Result Control (RFC 2696)
- ▶ All Operational Attributes plus attribute list feature (RFC 3673)
- ▶ supportedFeatures discover mechanism (RFC 3674)
- ▶ Content Synchronization operation
- ▶ WhoAmI? operation
- ▶ Proxy Authorization control
- ▶ Matched Values control (RFC 3876)
- ▶ Assertion control
- ▶ Pre/Post Read controls
- ▶ No-Op control
- ▶ Modify/Increment extension
- ▶ Absolute True (&) and False (!) Filter extension

OpenLDAP is also available for Solaris. In addition to the documentation provided by SUSE and Red Hat, for a good “how-to” for LDAP administration, refer to the Linux Documentation Project (<http://www.tldp.org/HOWTO/LDAP-HOWTO/index.html>). There are modules (pam_ldap and nss_ldap) available to allow for workstation authentication against LDAP.

The `/etc/ldap.conf` and `/etc/nsswitch.conf` files have the same content on both operating systems. If you plan to connect Linux LDAP clients to Sun’s LDAP offerings, the connection should be straightforward. If you are moving from Sun’s offerings to a Linux LDAP server solution, keep in mind that authentication methods and schemas will probably differ—the LDAP technical community have not completely standardized in these areas.

The basic configuration command on RHEL4 is **system-config-authentication**, and on SUSE the YaST2 GUI contains modules for configuring either an LDAP client or server.

The OpenLDAP administration commands themselves are:

ldapadd	Add entry
ldapcompare	Perform compare based on parameters
ldapdelete	Delete entry
ldapmodify	modify entry
ldapmodrdn	Rename (modifies RDN™) entry
ldappasswd	Set password of entry
ldapsearch	Connect to server and searches based on parameters
ldapwhoami	Perform LDAP “whoami” operation

11.4 NIS and NIS+

Linux contains full support for name switch service (NSS) and has support for the following NIS (and NIS+) maps: aliases, ethers, group, hosts, netgroups, networks, protocols, publickey, passwd, rpc, services, and shadow. It fully supports shadow passwords over NIS. Although NIS+ is supported, its use is not recommended on Linux, because LDAP is the favored solution in this space, and some NIS+ problems are still present.

Linux configures in a similar manner as Solaris. For a complete NIS/NIS+ (and YP) how-to, refer to:

<http://www.tldp.org/HOWTO/NIS-HOWTO/>

11.5 User ID and group ID differences

There are differences in the UIDs and GIDs used for the various users defined on the system by default. Table 11-1 lists UID differences. Table 11-2 on page 168 lists the GID differences.

Table 11-1 User ID differences

User	Solaris UID:GID	Red Hat UID:GID	SUSE UID:GID	Comment
root	0:1	0:0	0:0	Superuser
daemon	1:1	2:2	2:2	N/A
bin	2:2	1:1	1:1	N/A
sys	3:3	N/A	N/A	N/A
adm	4:4	3:4	N/A	Admin
uucp	5:5	10:14	10:14	uucp admin
nuucp	9:9	N/A	N/A	uucp admin
smmsp	25:25	51:51	N/A	SendMail Message Submission Program
listen	37:4	N/A	N/A	Network admin
lp	71:8	4:7	4:7	Line printer admin
nobody	60001:60001	99:99	65534:65533	Nobody
noaccess	60002:60002	N/A	N/A	No access user
nobody4	65534:65534	N/A	N/A	SunOS™ 4.x Nobody
sync	N/A	5:0	N/A	
shutdown	N/A	6:0	N/A	
halt	N/A	7:0	N/A	
mail	N/A	8:12	8:12	
news	N/A	9:13	9:13	
operator	N/A	11:0	N/A	
games	N/A	12:100	12:100	

User	Solaris UID:GID	Red Hat UID:GID	SUSE UID:GID	Comment
gopher	N/A	13:30	N/A	
man	N/A	N/A	13:62	Man pages viewer
ftp	N/A	14:50	40:49	FTP user
squid	N/A	23:23	31:65534	Squid proxy server
pvm	N/A	24:24	N/A	Parallel processing pkg
named	N/A	25:25	44:44	
at	N/A	N/A	25:25	Batch daemon
postgres	N/A	26:26	26:26	PostgreSQL server
mysql	N/A	27:27	60:2	mySQL server
ncsd	N/A	28:28	N/A	ncsd daemon
mdom	N/A	N/A	28:28	Mailing list agent
rpcuser	N/A	29:29	N/A	RPC service user
wwwrun	N/A	N/A	30:8	WWW daemon Apache
rpc	N/A	32:32	N/A	Portmapper™ RPC user
amanda	N/A	33:6	37:6	Amanada backup suite
netdump	N/A	34:34	104:104	netdump
rpm	N/A	37:37	N/A	Package manager
ntp	N/A	38:38	74:65534	
canna	N/A	39:39	N/A	Canna service users
irc	N/A	N/A	39:65534	IRC daemon
mailman	N/A	41:41	72:67	GNU mailing list mgr
gdm	N/A	42:42	50:15	GNOME desktop

User	Solaris UID:GID	Red Hat UID:GID	SUSE UID:GID	Comment
xfs	N/A	43:43	N/A	X11 Font Server
mailnull	N/A	47:47	N/A	
apache	N/A	48:48	N/A	Apache
wnn	N/A	49:49	N/A	Wnn input server
ldap	N/A	55:55	76:70	LDAP user
vscan	N/A	N/A	65:103	Virus scanner
webalizer	N/A	67:67	N/A	Webalizer
pop	N/A	N/A	67:100	POP server
haldaemon	N/A	68:68	N/A	HAL daemon
vcsa	N/A	69:69	N/A	Virtual console memory owner
snort	N/A	N/A	73:68	Snort network monitor
sshd	N/A	74:74	71:65	Privilege- separated SSH
radvd	N/A	75:75	N/A	Router advertisement daemon
cyrus	N/A	76:12	96:12	Cyrus IMAP server
pcap	N/A	77:77	N/A	Network monitor user
fax	N/A	78:78	N/A	mgetty fax spool
dbus	N/A	81:81	N/A	System message bus
postfix	N/A	89:89	51:51	Mail server
quagga	N/A	92:92	101:101	Quagga routing suite
exim	N/A	93:93	N/A	Spam/virus pkg
radiusd	N/A	95:95	102:102	Radius user

User	Solaris UID:GID	Red Hat UID:GID	SUSE UID:GID	Comment
dovecot	N/A	97:97	N/A	IMAP/POP3 server
ident	N/A	98:98	N/A	
htt	N/A	100:101	N/A	IIIMF Htt
stunnel	N/A	N/A	100:65534	SSL tunnel daemon
dhcpcd	N/A	N/A	103:65534	DHCP daemon
nfsnobody	N/A	65534:65534	N/A	

Table 11-2 Group ID differences

Group	Solaris	Red Hat	SUSE
root	0	0	0
other	1	N/A	N/A
bin	2	1	1
sys	3	3	3
adm	4	4	N/A
uucp	5	14	14
mail	6	12	12
tty	7	5	5
lp	8	8	7
nuucp	9	N/A	N/A
staff	10	N/A	N/A
daemon	12	2	2
sysadmin	14	N/A	N/A
smmsp	25	N/A	N/A
nobody	60001	99	65533
noaccess	60002	N/A	N/A
nogroup	65534	N/A	65534
disk	N/A	6	6

Group	Solaris	Red Hat	SUSE
mem	N/A	8	N/A
www	N/A	N/A	8
kmem	N/A	9	9
wheel	N/A	10	10
news	N/A	13	13
man	N/A	15	N/A
shadow	N/A	N/A	15
dialout	N/A	N/A	16
audio	N/A	N/A	17
floppy	N/A	19	19
games	N/A	20	40
cdrom	N/A	N/A	20
slocate	N/A	21	N/A
console	N/A	N/A	21
utmp	N/A	22	22
squid	N/A	23	N/A
pvm	N/A	24	N/A
named	N/A	25	44
at	N/A	N/A	25
postgres	N/A	26	26
mysql	N/A	27	N/A
nscd	N/A	28	N/A
mdom	N/A	N/A	28
rpcuser	N/A	29	N/A
gopher	N/A	30	N/A
rpc	N/A	32	N/A
public	N/A	N/A	32

Group	Solaris	Red Hat	SUSE
video	N/A	N/A	33
netdump	N/A	34	N/A
rpm	N/A	37	N/A
ntp	N/A	38	N/A
canna	N/A	39	N/A
dip	N/A	40	N/A
mailman	N/A	41	67
xok	N/A	N/A	41
gdm	N/A	42	N/A
trusted	N/A	N/A	42
xfst	N/A	43	N/A
modem	N/A	N/A	43
mailnull	N/A	47	N/A
apache	N/A	48	N/A
wnn	N/A	49	N/A
ftp	N/A	50	49
smmsp	N/A	51	N/A
lock	N/A	54	N/A
ldap	N/A	55	70
maildrop	N/A	N/A	59
man	N/A	N/A	62
pkcs11	N/A	N/A	64
sshd	N/A	N/A	65
webalizer	N/A	67	N/A
haldaemon	N/A	68	N/A
snort	N/A	N/A	68
vcsa	N/A	69	N/A

Group	Solaris	Red Hat	SUSE
ntadmin	N/A	N/A	71
sshd	N/A	74	N/A
radvd	N/A	75	N/A
pcap	N/A	77	N/A
fax	N/A	78	N/A
dbus	N/A	81	N/A
postfix	N/A	89	51
postdrop	N/A	90	N/A
quagga	N/A	92	101
exim	N/A	93	N/A
radiusd	N/A	95	102
dovecot	N/A	97	N/A
ident	N/A	98	N/A
users	N/A	100	100
htt	N/A	101	N/A
quaggavt	N/A	102	N/A
vscan	N/A	N/A	103
dump	N/A	N/A	104
nfsnobody	N/A	65534	N/A



Monitoring and performance

This chapter provides information about the commands available to monitor the utilization of, and display the statistics for, the physical and logical resources of your system. The parameters for the commands are generally the same on Solaris and Linux. Many of the commands provide information about several resources. The output from the commands might differ significantly.

The physical components of a Linux server are its:

- ▶ Processors (CPUs)
- ▶ Real memory (RAM)
- ▶ Physical media, such as disk drives and floppy disks
- ▶ Network devices

The logical components of a Linux server are its:

- ▶ Virtual memory (SWAP)
- ▶ Software RAID drives
- ▶ Logical volume groups and logical volumes
- ▶ File systems
- ▶ System and user processes, such as daemons and applications

Certain tools are installed by default. You should review these packages and install them if your installation requires the tools provided by the packages:

- ▶ For RHEL:
 - sysstat package: **sysstat**, **sysstat**
 - sysreport package: **sysreport** (system information collector)
- ▶ For SLES:
 - sysstat package: **sysstat**, **sysstat**
 - siga package: **siga** (System Information GATHERing)

Many of the commands discussed in this chapter have options that allow them to be run iteratively. The output from these commands will eventually scroll off the screen. You can save the output several different ways. For example:

- ▶ Execute the command in a “script” subshell.
- ▶ Redirect the output to a file with the redirect operator “>” or “>>”.
- ▶ Redirect the output to a file with the **tee** command and view it at the same.
- ▶ Execute the command with the **watch** command.

Here are three more points to consider:

- ▶ Log files should be reviewed on a regular basis.
- ▶ You can use the **dmesg** command to display the current messages in the kernel's ring buffer.
- ▶ The `/proc` file system contains system-wide and process-specific configuration and state information.

There are configuration files associated with some of the commands in this chapter. Refer to the documentation for each command to find the location and format of its configuration file. Linux kernel parameters can be displayed with the **sysctl -a** command.

Setting kernel parameters is outside the scope of this chapter.

12.1 Processors

Processor configuration information is in The `/proc/cpuinfo` file. Table 12-1 describes the differences in CPU monitoring commands.

Table 12-1 CPU monitoring commands

Solaris	Linux	Comments
<code>iostat</code>	<code>iostat</code>	Display I/O statistics
<code>mpstat</code>	<code>mpstat</code>	Display individual processor statistics in a multiprocessor system
<code>psrinfo, sar -r</code>	<code>procinfo</code>	Display system status for CPU and memory usage
<code>ps</code>	<code>ps</code>	Per process CPU utilization and time
<code>sag</code>	<code>isag</code>	Display sar data graphically
<code>top</code>	<code>top</code>	Display top CPU user for running processes and CPU and memory statistics
<code>uptime</code>	<code>uptime</code>	Load average for the past 1, 5 and 15 minutes
<code>vmstat</code>	<code>vmstat</code>	Display virtual memory statistics
<code>w</code>	<code>w</code>	Display who is currently logged in

In addition to these tools, a profiling suite known as OProfile is available. OProfile is a system-wide profiler for Linux systems, capable of profiling all running code without too much extra impact on system load. It consists of a kernel driver and a daemon for collecting sample data and several post-profiling tools for turning data into information. This tool ships with RHEL4. In SUSE, the kernel is enabled for it (the driver portion) and the daemon is available from:

<http://oprofile.sourceforge.net/>

12.2 Real and virtual memory

Physical and virtual memory are monitored by the same commands. The `/proc/meminfo` file contains detailed information concerning memory usage. Refer to Table 12-2 on page 176 for a description of memory monitoring commands.

In addition, Linux provides the GNOME System Monitor, a GUI for monitoring resources.

Table 12-2 Real and virtual memory monitoring commands

Solaris	Linux	Comments
<code>sar -r</code>	<code>free</code>	Display amount of free and used memory
<code>psrinfo</code>	<code>procinfo</code>	Display system status for CPU and memory usage
<code>sag</code>	<code>isag</code>	Display sar data graphically
<code>sar</code>	<code>sar</code>	Display, collect, or store system activity status
<code>top</code>	<code>top</code>	Display top CPU user for running processes and CPU and memory statistics
<code>vmstat</code>	<code>vmstat</code>	Display virtual memory statistics

12.3 Physical media, software RAID, LVM, and file systems

Physical media must be partitioned and formatted with a file system unless it will be used as a raw device. For this reason, we differentiate between physical media and file systems in the section. Commands to monitor disk partitions, which are analogous to Solaris' slices, will be listed as physical media monitoring commands.

Software RAID and LVM provide complementary functions. LVM does not replace RAID, nor does software RAID replace LVM.

Task table: Physical media monitoring

Table 12-3 illustrates the physical media monitoring commands.

Table 12-3 Physical media monitoring commands

Solaris	Linux	Comments
<code>iostat</code>	<code>iostat</code>	Display I/O statistics
<code>sag</code>	<code>isag</code>	Display sar data graphically
<code>sar</code>	<code>sar</code>	Display, collect, or store system activity status
<code>vmstat</code>	<code>vmstat</code>	-a and -d parameters valid on Linux kernels 2.5.x and later

Task table: Software RAID

Linux provides a different set of utilities for software RAID devices and logical volume management. This section contains commands specific to monitoring

software RAID devices. Table 12-4 illustrates software RAID monitoring commands.

Table 12-4 Software RAID monitoring commands

Solaris	Linux	Comments
<code>metastat</code>	<code>cat /proc/mdstat</code>	Display status information for software RAID devices
<code>metadb</code>	<code>lsraid</code> (SUSE only; deprecated in Red Hat, use <code>mdadm</code> instead)	List and query MD devices
<code>mdmonitord</code>	<code>mdadm</code>	Manage and monitor software RAID devices
N/A	<code>sgraidmon</code> (SUSE only)	Monitor SCSI devices for hot-removal and hot-insertion events

12.3.1 Logical volume groups and logical volumes

Linux provides a different set of utilities for software RAID devices and logical volume management. This section contains commands specific to monitoring logical volume groups and logical volumes.

The commands to manage logical volumes, groups, and devices are in these directories:

- ▶ SUSE: `/sbin`
- ▶ Re Hat: `/usr/sbin`

Both Red Hat and SUSE provide LVM2. SUSE also provides the Enterprise Volume Management System (EVMS). The Linux 2.6 kernel prevents multiple volume managers from managing the same device. EVMS will display information about LVM2-managed devices by default. EVMS can be configured to exclude information about these devices.

Task table: Logical volume monitoring

Table 12-5 illustrates logical volume monitoring commands.

Table 12-5 Logical volume monitoring commands

Solaris	Linux	Comments
<code>metastat</code>	<code>lvdisplay</code>	Display logical volume attributes
<code>metastat</code>	<code>lvs</code>	Report logical volume information
<code>metadb</code>	<code>pvdisplay</code>	Display physical volume attributes

Solaris	Linux	Comments
metadb	pvs	Report physical volume information
metastat	vgdisplay	Display volume group attributes
metastat	vgs	Report volume group information

12.3.2 File systems

We differentiate file systems from physical devices in this chapter, because a file system can extend beyond a single device. Examples of file systems that extend beyond a single device are hardware and software RAID drives and logical volumes.

Table 12-6 File system monitoring commands

Solaris	Linux	Comments
df	df	Display file system disk space usage information
du	du	Display individual file disk space usage
repquota	repquota	
N/A	stat	

12.4 Network

Table 12-7 illustrates some network activity monitoring commands. Additional network performance information is available in `/proc/slabinfo`.

Table 12-7 Network monitoring commands

Solaris	Linux	Comments
N/A	ethereal	Graphical network protocol analyzer. Available for Solaris, but not installed by default.
ifconfig	ifconfig	Configure network device.
N/A	mrtg	Monitor traffic load. Available for Solaris, but not installed by default.
netstat	netstat	Display network statistics, routing information, connections, and so on.
nfsstat	nfsstat	Display NFS statistics.

Solaris	Linux	Comments
ping	ping	Send ICMP echo request packets to network host.
sag	isag	Display sar data graphically.
sar	sar	The -n parameter on Linux can be used to report different types of network activity.
snoop	N/A	Display network packets.
spray	N/A	Test network connectivity using a UDP protocol.
tcpdump	tcpdump, tethereal	Dump network traffic.

12.5 System and user processes

Each process has a directory containing its state data. The directory is */proc/nnnn*, where *nnnn* is the process ID (PID) number. Table 12-8 lists some process monitoring commands.

Table 12-8 Process monitoring commands

Solaris	Linux	Comments
ipcs	ipcs	Display interprocess communication facilities status
ldd	ldd	Display shared library dependencies
proc tools	lsdf	Display open files
ldd	ltrace	Display dynamic library calls
ps	ps	Display process status
pstree	pstree	Display process inter-dependencies in a tree format
sag	isag	Display sar data graphically
sar	sar	Display, collect, or store system activity status
truss	strace	Trace system calls and signals
top	top	Display top CPU user for running processes and CPU and memory statistics
vmstat	vmstat	Display virtual memory statistics



Backup and restore

This chapter describes some basic UNIX backup and compression tools and the differences in usage between Solaris and Linux.

We discuss the following topics:

- ▶ Common backup tools
- ▶ Compression tools
- ▶ File system backup and restore
- ▶ File system snapshots
- ▶ AMANDA

13.1 Common backup tools

Performing backups is a critical component of system maintenance. UNIX has several options available for the system administrator to use. Across UNIX, the most common tools available for backup are:

- ▶ **tar**
- ▶ **cpio**
- ▶ **dd**
- ▶ **pax**

These commands operate the same way on Linux as they do on Solaris, although there might be some differences in functionality. Refer to 13.3, “File system backup and restore” on page 183 for a highlight of the additional GNU **tar** functions on Linux.

13.2 Compression tools

System administrators use compression tools as a quick way to recover file system space when it is at a premium. They also allow the system administrator or user to keep copies, possibly multiple copies, of earlier versions of files without taking up as much space. In addition, administrators can use these tools to compress files before writing them to an archive device. Whatever the usage might be, these are convenient tools for use in the system administrator's day-to-day life cycle.

Common tools

The following tools are some of the tools available for all three operating systems discussed in this book:

- ▶ **bzip2, bunzip2, bzip2recover**
- ▶ **compress, uncompress**
- ▶ **gzip, gunzip, gzc**
- ▶ **gzip** (recompress .Z files to .gz files)
- ▶ **gzexe** (compress executable files in place)
- ▶ **pack, pcat, unpack**
- ▶ **zip, zipcloak, zipnote, zipsplit**

There are other programs that come with some of these tools that do specific functions. For example, **gzgrep** provides the ability to **grep** within a compressed

file and **bzless**, which enables a user to page a compressed file. For Solaris, some of these other programs are only available in Solaris 9, or as part of the Companion CD installation.

For more details about how to use each tool, refer to their operating system man pages.

13.3 File system backup and restore

This section discusses the unique backup and restore tools used in Solaris and their functional equivalents in Linux.

Task table

Table 13-1 illustrates the basic file system backup and restore commands.

Table 13-1 File system backup commands

File system backup task	Solaris	Linux
Full or incremental backup	ufsdump	dump (for ext2/3 file system only) tar
Restore	ufsrestore	restore (for ext2/3 file system only) tar

13.3.1 Solaris overview

Solaris comes with a useful tool for doing file system backups called **ufsdump**. A limitation to this tool is that it only works on the UFS file system. This tool enables you to do full and incremental backups based on the parameter options given to the tool. The corresponding tool for recovery is **ufsrestore**. The disadvantage of any backup tool is that the file, or file system, being stored in the backup needs to be quiescent in order to avoid having partially updated files or file systems stored in the backup.

ufsdump is an all-inclusive tool as far as being able to do incremental backups. However, you can also use **pax** if you want the archive to be portable across different versions of UNIX or platforms, or both.

13.3.2 Linux overview

Linux comes with the **dump** command that is specific to the ext2 and ext3 file systems. The program works at the inode-level and does not change the

last-access date stamps. This tool is very useful for doing a full backup. The corresponding tool to this is the **restore** command.

The GNU **tar** command that comes in Linux has the following useful features:

- ▶ **--atime-preserve**: Option to not change access times on dumped files.
- ▶ **--checkpoint**: Print directory names while read the archive.
- ▶ **-f, --file=[HOSTNAME:]<device>**: Use archive file or device <device>.
- ▶ **-F, --info-script=<file>, --new-volume-script=<file>**: Run the script <file> at the end of each archive or tape.
- ▶ **-g, --listed-incremental=<file>**: Create, list, or extract new GNU-format incremental backup file.
- ▶ **-h, --dereference**: Do not dump symlinks; dump the files to which they point.
- ▶ **-j, --bzip2, --bunzip2**: Filter the archive through **bzip2**.
- ▶ **-l, --one-file-system**: Stay in local file system when creating an archive.
- ▶ **-N, --after-date=<DATE>, --newer=<DATE>**: Store only files newer than <DATE>.
- ▶ **-V, --label=<NAME>**: Create archive with volume name <NAME>.
- ▶ **-Z, --compress, --uncompress**: Filter the archive through **compress**.
- ▶ **-z, --gzip, --ungzip**: Filter the archive through **gzip**.

For more information about the GNU **tar** command, refer to the man pages or run **tar --help** for a more updated list of available options.

13.4 File system snapshots

Using file system snapshots is another way to archive files while still allowing read/write activity on the file system. Performance degradation is only affected during the snapshot creation phase and when users want to update files after the snapshot is done. Care should be taken when the file system is large, because the snapshot size is correspondingly large.

Overview of Solaris task

Solaris comes with the **fssnap** command to create and manage a file system snapshot. After taking a snapshot, you can store the file system in a backup using the backup tool of our choice.

Linux task

Linux does not come with a “file system snapshot” as Solaris does. The closest thing it has is the **dump** command, as referred to in the previous section.

There is a script with about the same functionality as **fssnap**. It was originally written by Mike Rubel and is available at:

<http://www.mikerubel.org>

Look for the “Rsync snapshot” article.

13.5 AMANDA

AMANDA, the Advanced Maryland Automated Network Disk Archiver, is a public domain utility developed at the University of Maryland. It is a backup system that enables the administrator of a LAN to set up a single master backup server to back up multiple hosts to a single large capacity tape drive. AMANDA uses native dump or GNU tar facilities and can back up a large number of workstations running multiple versions of UNIX. Recent versions can also use SMB to back up Microsoft Windows hosts.

For more information about this tool, see:

<http://www.amanda.org/>



Security and hardening

This chapter discusses the differences in security and hardening configuration between Solaris and Linux. This chapter does not instruct the you about how to lock down a system, because those settings typically differ depending on how the system is used and what governance it must follow.

We discuss the following topics:

- ▶ Patching and security updates
- ▶ Security hardening
- ▶ Securing and removing services
- ▶ Kernel tuning for security
- ▶ Logging
- ▶ Access control lists
- ▶ PAM
- ▶ umask
- ▶ SSH
- ▶ IPSec and IKE
- ▶ Kerberos
- ▶ Warning banners
- ▶ Firewalls

14.1 Patching and security updates

The security of a system is only as good as its weakest link. While it is important for systems to be properly secured when they are initially installed and put on a network, it is imperative that each system be regularly patched against new security vulnerabilities. Solaris, Red Hat, and Novell SUSE are all supported through vendor services that release security patches, alerts, and software upgrades at the following locations:

- ▶ Solaris:

<http://sunsolve.sun.com/pub-cgi/show.pl?target=security/sec>

- ▶ Red Hat:

<https://www.redhat.com/apps/support/>

- ▶ SUSE:

<http://www.novell.com/linux/security/securitysupport.html>

14.2 Security hardening

Security hardening can be as important with Linux as it is with Solaris or any other type of UNIX-like operating system. Many “out-of-the-box” installations of Linux have numerous features that are disabled, but to really harden a system before putting it online, consider using a hardening tool.

14.2.1 Hardening tools

Most Solaris administrators are familiar with Yet Another Solaris Security Package, YASSP (<http://www.yassp.org/>) and the Solaris Security Toolkit. These tools are used to more efficiently harden a Solaris system. In Linux, one of the most popular tools used to implement system hardening is called Bastille.

Bastille Linux is an open source project aimed at hardening a Linux system by changing some of the system’s settings based on security best practices. For more information about Bastille, see:

<http://www.bastille-linux.org>

Bastille is a useful tool for system administrators to discover and tighten Linux security settings. A graphical user interface enables administrators to choose security settings appropriate for their environment. Bastille automatically implements changes based on selections provided by the user.

14.2.2 Auditing: ASET/BSM

As we discussed in the last section, Linux can be hardened using a tool called Bastille. Many Solaris system administrators are familiar with using the Automated System Enhancement Tool (ASET) and Basic Security Module (BSM) as ways to set basic security parameters, keep those settings configured appropriately, and monitor security levels.

On the Linux platform, Bastille can certainly help in creating a secure Linux configuration, but for more enhanced auditing functionality with features similar to ASET and BSM, look at Security Enhanced Linux (SE-Linux). SE-Linux is a special Red Hat package that was created by the National Security Agency (NSA). It provides role-based access control and additional auditing features. For more information about SE-Linux, see:

<http://www.nsa.gov/selinux/>

14.3 Securing and removing services

Both Solaris and Linux have many capabilities that are necessary in one environment but not necessarily in another. Additionally, some services are safe to run in one environment, but that same service will open the system to attack in a different environment. This being the case, system administrators are always trying to allow only the necessary services to run on their systems to prevent unnecessary security exposure. This section focuses on a few key differences between the two environments that enable the system administrator to identify and disable unnecessary services.

14.3.1 inetd/xinetd

In Solaris, the more traditional Internet services are enabled or disabled in the file `/etc/inetd.conf`. As we discuss in 7.8.1, “inetd-based versus xinetd-based network services” on page 117, Linux uses a slightly different method than Solaris when it comes to **inetd** services. From a security perspective, it is important to ensure that all unnecessary services are disabled and TCP wrappers are set to monitor all enabled services that can be wrapped.

Table 14-1 on page 190 shows the different locations of the **inetd** configuration files.

Table 14-1 *inetd configuration files*

Services	Solaris	Red Hat	SUSE
Configuration locations	/etc/inetd.conf	/etc/xinetd.d/ /etc/xinetd.conf	/etc/xinetd.d/ /etc/xinetd.conf

To disable an **xinetd** service, edit the appropriate file in the `/etc/xinetd.d/` directory by ensuring that the line `disable = yes` is present in the file.

Example 14-1 provides an example of how to disable the Telnet service.

Example 14-1 *Sample xinetd entry for a service*

```
# default: on
# description: The telnet server serves telnet sessions; it uses \
# unencrypted username/password pairs for authentication.
service telnet
{
    flags          = REUSE
    socket_type    = stream
    wait          = no
    user           = root
    server         = /usr/sbin/in.telnetd
    log_on_failure += USERID
    disable        = yes
}
```

To enable a service, change the entry to `disable = no`.

14.3.2 TCP wrappers

When enabling any **inetd** services in Linux, TCP wrappers add an extra layer of access control the same way it does with Solaris. In most Linux distributions, TCP wrappers are installed and enabled by default.

As in Solaris, TCP wrappers access control is managed through two configuration files:

/etc/hosts.allow This file is used to set the systems and protocols that are allowed to connect to this host.

/etc/hosts.deny This file is used to set the systems and protocols that are not allowed to connect to this host.

In Solaris, TCP wrappers are enabled by adding the following line to the `/etc/default/inetd` file:

```
ENABLE_TCPWRAPPERS=YES
```

This is not necessary in Linux because TCP wrappers are typically enabled during installation.

In Solaris, each service to be wrapped must be specified as such in the `/etc/inetd.conf` file. Example 14-2 shows how to “wrap” the Telnet service in the `/etc/inetd.conf` file.

Example 14-2 Wrapping the Telnet service in `/etc/inetd.conf`

```
telnet stream tcp nowait root /usr/sbin/tcpd in.telnetd
```

Linux TCP wrappers are automatically applied to all **xinetd** services as long as TCP wrappers are compiled in at installation time.

For detailed configuration information, check the following resources.

- ▶ Chapter 15, “TCP Wrappers and xinetd,” in *Red Hat Linux 9: Red Hat Linux Reference Guide*, available at:
<http://www.redhat.com/docs/manuals/linux/RHL-9-Manual/ref-guide/ch-tcpwrappers.html>
- ▶ See also 7.9, “TCP wrappers” on page 123

14.3.3 FTP

FTP historically is a protocol has had a lot of documented security issues associated with its use, for example, the broadcasting of unencrypted user names and passwords during authentication. Nevertheless, FTP is a very popular and useful internetworking protocol. This section discusses methods for running FTP on Linux in a reasonably secure manner.

Implementations of the FTP protocol are generally similar across all platforms. However, there are different versions of the server and several differences between the Solaris and Linux versions of the FTP server. Solaris currently installs `wu-ftp` as its default FTP server. Red Hat and SUSE both use Very Secure FTP (**vsftpd**).

vsftpd is a GPL-licensed FTP server for UNIX systems and is included in Linux. It is considered secure, extremely fast, and stable.

Although it is small in size, **vsftpd** has a lot of functionality. Some of the more popular features of **vsftpd** include:

- ▶ Virtual IP configurations
- ▶ Virtual users
- ▶ Ability to **chroot** users into their home directories

- ▶ Run either stand-alone or from **inetd**
- ▶ Written to reduce the risk of buffer overflow attacks
- ▶ Powerful per-user configurability
- ▶ Reduce the risk of DoS attacks by using bandwidth throttling
- ▶ IPv6 support
- ▶ Support for SSL encryption

One of the most important security-related differences between Solaris and Linux FTP is the location of the `ftputers` file. This file is used to control access to the FTP server.

- ▶ Solaris: `/etc/ftpd/ftputers`
- ▶ Linux: `/etc/ftputers`

With all of these advanced features, the following references should be helpful in learning and configuring **vsftp**:

http://vsftpd.beasts.org/vsftpd_conf.html
<http://vsftpd.beasts.org/>
<http://www.vsftpdrocks.org/>

14.4 Kernel tuning for security

Solaris kernel tuning to modify behavior of TCP for security purposes can be manually implemented using the **ndd** command. The problem with manual configuration at the command prompt is that the changes will not survive a reboot. Because of this, the **ndd** commands can be set more permanently by placing them in a boot-time shell script, such as `/etc/init.d/inetinit`.

Similarly, Linux allows for manual kernel tuning at the command prompt using the **sysctl** command. But for persistent configuration changes, you need to modify the `/etc/sysctl.conf` file. If changes are made to the `/etc/sysctl.conf` file, the changes will not become active until the system is rebooted or the command is executed:

```
# sysctl -wp /etc/sysctl.conf
```

Here we provide some examples of kernel tuning in both Solaris and Linux.

For socket queue defense against SYN attacks:

- ▶ Solaris: `/usr/sbin/ndd -set /dev/tcp tcp_conn_req_max_q 1024`
- ▶ Linux: `sysctl -w net.ipv4.tcp_max_syn_backlog=1280`

Redirects (IP redirects can be used to modify the routing table on a remote host):

► Solaris:

```
/usr/sbin/ndd -set /dev/ip ip_ignore_redirect 1  
/usr/sbin/ndd -set /dev/ip ip_send_redirects 0
```

► Linux:

```
sysctl -w net.ipv4.conf.all.send_redirects=0  
sysctl -w net.ipv4.conf.all.accept_redirects=0
```

As you can see, Linux uses the **sysctl** command to modify kernel parameters at run time. These kernel parameters are controlled by the files in the `/proc/sys/` directory. For more detailed information, check the **sysctl** man page online at:

http://linuxcommand.org/man_pages/sysctl18.html

Another good resource about this topic is the *UNIX IP Stack Tuning Guide* at:

<http://www.cymru.com/Documents/ip-stack-tuning.html>

14.5 Logging

Logging in Linux is similar to Solaris with a few exceptions. For the most part, Solaris logs syslog messages to the `/var/adm/messages` file unless otherwise specified in the `/etc/syslog.conf` file. Linux by default logs many of its syslog messages in numerous files in `/var/log/` by default. As with Solaris, these are specified in the `/etc/syslog.conf` file. Table 14-2 shows many of the default Linux log files found in `/var/log/`.

Table 14-2 Security logging facilities

	Solaris	Red Hat	SUSE
Main log file	<code>/var/adm/messages</code>	<code>/var/log/messages</code>	<code>/var/log/messages</code>
authpriv		<code>/var/log/secure</code>	
cron logs	<code>/var/cron/log</code>	<code>/var/log/cron</code>	
Mail messages		<code>/var/log/maillog</code>	<code>/var/log/mail</code> <code>/var/log/mail.info</code> <code>/var/log/mail.warn</code> <code>/var/log/mail.err</code>
Boot messages	<code>/var/log/boot.msg</code>	<code>/var/log/boot.log</code>	

For more information about logging, see 17.4, “Logs” on page 220.

14.6 Access control lists

Access control lists (ACLs) are managed in much the same manner in Linux as they are in Solaris. The commands `chac1`, `getfac1`, and `setfac1` should be familiar to Solaris and Linux administrators alike. Because much of the Linux ACL functionality is the same as Solaris, you should be able to manage the transition with little direction. For further review of the more technical details of Linux ACLs, refer to:

- ▶ POSIX ACLs on Linux:

<http://www.suse.de/~agruen/ac1/linux-ac1s/online/>

- ▶ Red Hat ACLs:

<http://www.redhat.com/docs/manuals/enterprise/RHEL-4-Manual/sysadmin-guide/ch-ac1s.html>

14.7 PAM

The default mechanism used for user authentication in Linux uses Pluggable Authentication Modules (PAM). PAM provides centralized authentication for common services such as login, FTP, Telnet, and SSH. PAM is implemented using dynamic load libraries (DLLs). Authentication can be customized for PAM-aware services using PAM configurations files. Figure 14-1 illustrates PAM operations.

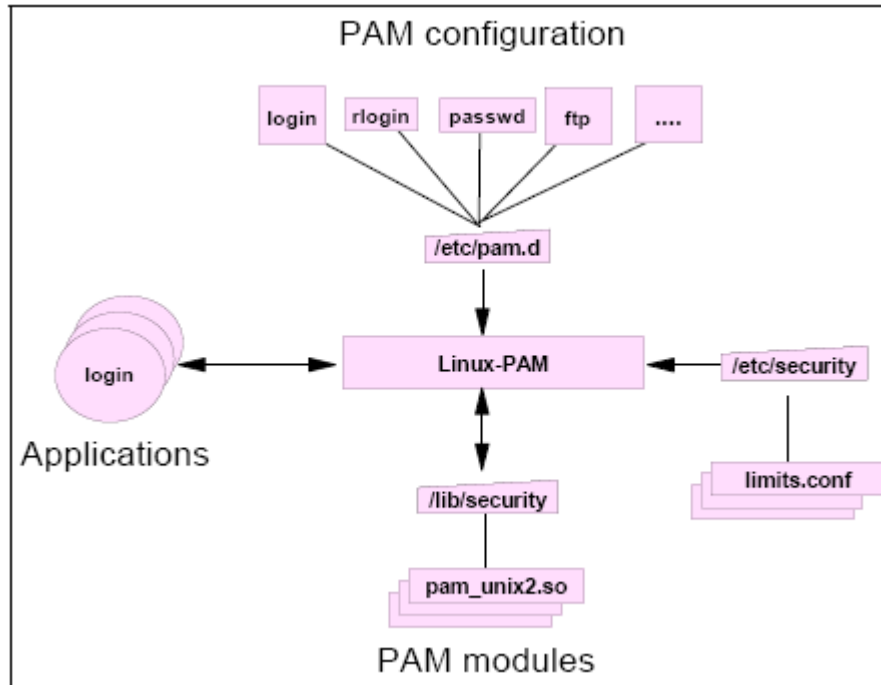


Figure 14-1 PAM operations

Authentication for each PAM-aware application is configured by a configuration file in the `/etc/pam.d` directory (the file name indicates the service it configures). PAM modules implement user authentication; by default, PAM modules are located in the `/lib/security` directory. Some PAM modules read global configuration files located in the `/etc/security` directory.

Note: Solaris PAM configuration occurs in the `/etc/pam.conf` file, and Linux uses the `/etc/pam.d` directory.

A PAM configuration file consists of lines in the form:

```
module-type control-flag module-path arguments
```

We describe each field in the following sections.

14.7.1 PAM module types

Table 14-3 on page 196 shows four types of authentication mechanisms (referred to as module types).

Table 14-3 Authentication mechanisms

Type	Description
auth	Used to authenticate a user based on user name and password
account	Used to grant or deny access based on account policies (such as time of day)
password	Used to authorize user password updates
session	Used to apply settings before a session is established

Note: More than one line can be specified for any PAM module type. This is referred to as *module stacking*. In this case, each module is executed in first-to-last order. Final authentication depends on the PAM control flags specified.

14.7.2 Limiting superuser login to secure terminals

Superuser access should always be restricted to a secure terminal. Network login by root should be denied. This can be enforced using the `pam_securetty.so` module. If enabled when a superuser logs in, the `pam_securetty.so` modules deny access unless the login originates from a terminal device listed in the `/etc/securetty` file. Example 14-3 shows the default secure device defined for SLES8 on a zSeries platform.

Example 14-3 The SLES8 default /etc/securetty file

```
# This file contains the device names of tty lines (one per line,
# without leading /dev/) on which root is allowed to login.
ttyS0
```

If superuser authority is required for a network session, log in as a non-privileged user and then issue the `su` command to switch to superuser. Alternatively, consider using `sudo` to delegate authority.

The `pam_securetty.so` module is not enabled for SSH in the default SLES8 installation. It might be desirable to deny superuser SSH network access even though the data is encrypted. This can help prevent an attacker from guessing the root password. See Table 14-4 on page 197.

Table 14-4 Limiting superuser login

	Solaris	Red Hat	SUSE
Allow/deny root login	/etc/default/login	/etc/securetty	/etc/securetty

14.7.3 Restricting user login

At times, it might be useful to restrict non-superuser logins (for example, when performing system maintenance). The `pam_nologin.so` module can be used to deny access to the system. When enabled, `pam_nologin.so` checks for the existence of the `/etc/nologin` file. If found, `pam_nologin.so` denies access and displays the contents of the file to the user. An alternate nologin file can be specified by supplying the `file=pathname` PAM argument.

Note: The nologin mechanism does not affect the superuser login. Superusers can log in even if the `/etc/nologin` file exists.

14.8 umask

The function and usage of `umask` is the same between Solaris and Linux. The configuration locations are a little different, as shown in Table 14-5.

Table 14-5 Controlling default umask

	Solaris	Red Hat	SUSE
<code>umask</code>	/etc/profile	/etc/bashrc	/etc/profile

14.9 SSH

The Secure Shell (also known as SSH) is a secure network access protocol. Traditional network access protocols (Telnet, FTP, and the `r-command` suite, such as `rlogin`, `rnp`, and `rsh`) must be considered inherently insecure. Unencrypted data (including user IDs and passwords) is transmitted over the network and can be captured by malicious third parties. SSH provides a secure alternative for remote access, file transfer, and remote command execution.

SSH was originally developed by Tatu Ylönen (a researcher at the Helsinki University of Technology) as a method to prevent password-sniffing attacks. His initial implementation (known as SSH1) was released as free software. The SSH-1 protocol was formalized as an Internet Engineering Task Force (IETF) Draft. Subsequent improvements resulted in a new SSH-2 protocol originally

implemented as the SSH2 commercial product available from SSH Communications Security, Ltd. (founded by Ylönen in 1995). OpenSSH is an open source implementation of Secure Shell based on the last free version of SSH1. It supports both the SSH-1 and SSH-2 protocols. OpenSSH is provided as part of most distributions of Linux. For details about OpenSSH, refer to:

<http://www.openssh.com>

Note: When we use the term SSH in the remainder of this chapter, we are referring specifically to the OpenSSH implementation.

14.10 IPsec and IKE

In this section, we discuss the differences in IPsec/IKE implementation between Red Hat and SUSE.

Red Hat IPsec

The Red Hat Enterprise Linux implementation of IPsec uses IKE for sharing keys between hosts across the Internet. The racoon keying daemon handles the IKE key distribution and exchange.

Implementing IPsec requires that the ipsec-tools RPM package be installed on all IPsec hosts (if using a host-to-host configuration) or routers (if using a network-to-network configuration). The RPM package contains essential libraries, daemons, and configuration files to aid in setup of the IPsec connection, including:

/lib/libipsec.so	A library that contains the PF_KEY trusted key management socket interface between the Linux kernel and the IPsec implementation used in Red Hat Enterprise Linux.
/sbin/setkey	setkey manipulates the key management and security attributes of IPsec in the kernel. This executable is controlled by the racoon key management daemon. For more information about setkey , refer to the setkey(8) man page.
/sbin/racoon	The IKE key management daemon, used to manage and control security associations and key sharing between IPsec-connected systems. This daemon can be configured by editing the <code>/etc/racoon/racoon.conf</code> file. For more information about racoon , refer to the racoon(8) man page.

/etc/racoon/racoon.conf The racoon daemon configuration file used to configure various aspects of the IPsec connection, including authentication methods and encryption algorithms used in the connection. For a complete listing of directives available, refer to the **racoon.conf(5)** man page.

Configuring IPsec on Red Hat Enterprise Linux can be done through the Network Administration Tool or by manually editing networking and IPsec configuration files. For more information about using the Network Administration Tool, refer to the *Red Hat Enterprise Linux System Administration Guide*, available at:

<http://www.redhat.com/docs/manuals/enterprise/RHEL-4-Manual/security-guide/ch-vpn.html>

SUSE IPsec

SLES uses the Road Warrior server as its VPN tool. *Road Warrior server* is a VPN server that accepts connections from any client with a valid and signed CA certificate. Perform the following three steps to set up a Road Warrior server:

1. Create a server certificate on the CA management computer.
2. Import a certificate on the server computer.
3. Set up a connection on the server.

For details about how to configure both the IPsec server and client on SLES, refer to Chapter 26 of the document available at:

http://www.novell.com/documentation/sles9/pdfdoc/sles_9_admin_guide/sles_9_admin_guide.pdf

14.11 Kerberos

We discuss the differences in Kerberos implementations between Red Hat and SUSE in this section.

Red Hat Kerberos

To configure a basic Red Hat Kerberos server, follow these steps:

1. Ensure that clock synchronization and DNS are running.
2. Install the `krb5-libs`, `krb5-server`, and `krb5-workstation` packages on the KDC system.
3. Edit the `/etc/krb5.conf` and `/var/kerberos/krb5kdc/kdc.conf` configuration files to reflect your realm name and domain-to-realm mappings.

4. Create the kerberos database using the following command:

```
/usr/kerberos/sbin/kdb5_util create -s
```

5. Edit the `/var/kerberos/krb5kdc/kadm5.acl` file.
6. Start Kerberos using the following commands:

```
/sbin/service krb5kdc start  
/sbin/service kadmin start  
/sbin/service krb524 start
```

7. Add principals for your users using the `addprinc` command with `kadmin`.
8. Verify that the server will issue tickets by running `kinit` to obtain a ticket and store it in a credential cache file.

For more information about how to configure a Kerberos server, see:

<http://www.redhat.com/docs/manuals/linux/RHL-9-Manual/ref-guide/s1-kerberos-server.html>

The Red Hat kerberos distribution has the ability to run through the configuration of services within `xinetd`. Example 14-4 shows the contents of a typical `xinetd` configuration for some “kerberized” services (from `/etc/xinetd.d/`).

Example 14-4 xinted service entries

```
krb5-telnet  
  
service telnet  
{  
    flags          = REUSE  
    socket_type    = stream  
    wait          = no  
    user          = root  
    server         = /usr/local/krb5/sbin/telnetd  
    server_args    = -X KERBEROS_V4 -a valid  
    log_on_failure += USERID  
    disable       = no  
}  
  
kshell  
  
service kshell  
{  
    flags          = REUSE  
    socket_type    = stream  
    wait          = no  
    user          = root  
    server         = /usr/local/krb5/sbin/kshd
```

```

        server_args = -5 -c -A
        disable     = no
    }

klogin

service klogin
{
    flags          = REUSE
    socket_type    = stream
    wait           = no
    user           = root
    server         = /usr/local/krb5/sbin/klogind
    server_args    = -5 -c
    disable        = no
}

eklogin

service eklogin
{
    flags          = REUSE
    socket_type    = stream
    wait           = no
    user           = root
    server         = /usr/local/krb5/sbin/klogind
    server_args    = -5 -c -e
    disable        = no
}

gss-ftp

service ftp
{
    flags          = REUSE
    socket_type    = stream
    wait           = no
    user           = root
    server         = /usr/local/krb5/sbin/ftpd
    server_args    = -a
    disable        = no
}

```

SUSE Kerberos

SUSE uses the Heimdal Kerberos implementation. This functions similarly to other Kerberos implementations. For configuration and administration details of both the Key Distribution Center (KDC) and the Kerberos client, see Chapter 26

of the *SUSE Linux Enterprise Server 9 Administration and Installation Guide*, available at:

<http://www.novell.com/documentation/sles9/>

14.12 Warning banners

Warning banners are used both prior to, and after successful login. Many organizations have specific legal verbiage that must be placed in these warnings, but from a security perspective, it is common practice to remove all references to the operating system name and version from the banners to prevent an attacker from easily identifying a system's potential weaknesses. Table 14-6 shows the paths to the common locations for the control files remaining the same between Solaris and Linux.

Table 14-6 Locations of motd and issue files

	Solaris	Red Hat	SUSE
Warning at login prompt	/etc/issue	/etc/issue	/etc/issue
Warning after successful login	/etc/motd	/etc/motd	/etc/motd

14.13 Firewalls

Both Linux and Solaris include firewall services. Linux uses the open source GPL Netfilter, and Solaris 9 includes SunScreen 3.2, a Sun-branded, non-open source firewall.

Both firewalls offer standard filtering such as blocking traffic by port or protocol, stateful inspection, and logging, and both offer NAT and IP masquerading capabilities.

Note: Both firewalls have the ability to run as a host-based firewall to protect only the host on which it is installed.

Consider that there are two general types of firewalls:

- ▶ Those that protect networks and hosts on networks
- ▶ Those that protect individual hosts

The Linux firewall can be configured to act as both a network-based firewall with stateful inspection and packet filtering, and as an individual firewall protecting only the host on which it is installed.

The Linux firewall is made up of three basic components:

- ▶ **Tables:** Tables can be thought of as a set of firewall rules.
- ▶ **Chains:** Chains are groups of rules found within a table. The default chains are FORWARD, INPUT, and OUTPUT.
- ▶ **Policies:** Policies are the default actions taken on a packet when it does not match any of the chains. The policies for packets are DROP, REJECT, and ACCEPT.

The primary command facility used to add and remove rules to or from a Linux firewall is **iptables**.

Table 14-7 shows some of the more useful **iptables** command options. For a more exhaustive list, see the **iptables** man page.

Table 14-7 *iptables* command options

Command	Description
iptables -A	Append a rule to the end of the selected chain
iptables -D	Delete one or more rules from the selected chain
iptables -I (capital i)	Insert a rule into the selected chain as a specific rule number
iptables -R	Replace a rule in the selected chain
iptables -L	List all of the rules in the selected chain
iptables -F	Flush the selected chain (delete all rules)
iptables -N	Create a new user-defined chain
iptables -h	Help

For more detailed information about how to configure firewall services for Linux, see the following links:

- ▶ Red Hat:

<http://www.redhat.com/docs/manuals/enterprise/RHEL-4-Manual/security-guide/>
<http://www.redhat.com/docs/manuals/linux/RHL-9-Manual/install-guide/s1-firewallconfig.html>

► Novell SUSE:

<http://www.novell.com/documentation/sles9/>

<http://www.novell.com/documentation/suse91/suselinux-adminguide/html/ch19.html>



Linux high availability overview

This chapter describes high availability features for Linux to better enable migration from a Solaris high availability system, such as SunCluster. We discuss the services available and considerations for the migration environment and provide pointers to more information.

This chapter introduces and discusses the following topics:

- ▶ Introduction to Linux-HA
- ▶ Linux-HA services
- ▶ Linux migration
- ▶ Cluster environment considerations

Significant portions of this chapter are from the High Availability Linux project Web site, available at:

<http://linux-ha.org>

15.1 Introduction to Linux-HA

Linux high availability (Linux-HA) provides basic high-availability (fail-over) capabilities for Linux. It provides monitoring of cluster nodes and applications, and provides a sophisticated dependency model with a rule-based resource placement scheme. When faults occur, or a rules change occurs, the user-supplied rules are then followed to provide the desired resource placement in the cluster.¹ For much more detailed information about Linux-HA, including configuration details, tutorials, and technical papers, refer to the High Availability Linux project at:

<http://linux-ha.org>

15.1.1 Migration scenarios

Linux-HA can be deployed in several different migration environments:

- ▶ Single node HA system: Provides stand-alone application monitoring and restart
- ▶ Multiple virtual machines (single physical machine): Adds OS crash protection and also provides for rolling upgrades of OS and applications
- ▶ Multiple physical machines: Adds protection against hardware failures
- ▶ Split-site (stretch) clusters: Adds protection against site failures

15.1.2 Some features of Linux-HA

Some of the features and functionality offered by Linux-HA include¹:

- ▶ Works on all known Linux variants and platforms, supplied with many distributions including Novell SUSE starting with SLES9 (not yet included in Red Hat).
- ▶ Sub-second server failure detection.
- ▶ Sophisticated knowledge of service interdependencies, always starts and stops resources quickly and correctly.
- ▶ Policy-based resource placement puts services exactly where you want them based on dependencies, user-defined rules, and node attributes.
- ▶ Fine-grained resource management can be based on user-defined criteria.
- ▶ Time-based rules allow for different policies depending on time.
- ▶ Easy-to-understand and deploy init-based application management; most services require no application scripts for basic management.

¹ From <http://linux-ha.org>

- ▶ Resource groups provide simple, easy-to-use service management for straightforward environments.
- ▶ Active fencing mechanism (STONITH) provides strong data integrity guarantees, even in unusual failure cases. Provides stronger integrity guarantees than simple quorum tie-breakers.
- ▶ Open source implementation avoids vendor lock-in and provides great flexibility, stability, responsiveness, and testing.
- ▶ Works on stand-alone machines to provide easy-to-use, init-script-based service monitoring and restart.

15.2 Linux-HA services

In this section, we describe the different system services that work together to provide the Linux-HA functions.

15.2.1 Heartbeat

The Heartbeat program is one of the core components of Linux-HA. It performs death-of-node detection, communications, and cluster management in one process.

Heartbeat monitors node death and includes built-in resource monitoring for all types of resources. Resources are automatically restarted on failure.¹ Heartbeat also provides strongly authenticated, locally ordered, multicast messaging.

15.2.2 Cluster Resource Manager (CRM)

The Cluster Resource Manager (CRM) is the brain of Linux-HA. It maintains the resource configuration, decides what resources should run where, decides how to move from the current state into the desired state. In addition, it supervises the Local Resource Manager (LRM) in accomplishing all these things.¹

CRM functionality:

- ▶ Uses heartbeat for communication
- ▶ Receives membership updates from the Consensus Cluster Membership
- ▶ Directs the work of and receives notifications from the Local Resource Manager
- ▶ Tells the STONITH daemon when and what to reset
- ▶ Logs using the logging daemon

15.2.3 Consensus Cluster Membership (CCM)

Consensus Cluster Membership (CCM) provides strongly connected consensus cluster membership services. Ensures that every node in a computed membership can talk to every other node in this same membership.

15.2.4 Local Resource Manager (LRM)

The Local Resource Manager is basically a resource agent abstraction. It starts, stops, and monitors resources as directed by the CRM.

It has a plug-in architecture and can support many classes of resource agents. The types currently supported include:¹

- ▶ Open Cluster Framework (OCF). For more information, refer to:
<http://opencf.org>
- ▶ Heartbeat (release 1) style resource agents.
- ▶ Linux Standards Base (LSB) (normal init script) resource agents.
- ▶ STONITH. Resource agent interface for instantiating STONITH objects to be used by the STONITH daemon.

15.2.5 STONITH daemon

The STONITH daemon provides cluster-wide node reset facilities using the improved release 2 STONITH API. The STONITH library includes support for about a dozen types of C STONITH plug-ins and native support for script-based plug-ins, allowing scripts in any scripting language to be treated as full-fledged STONITH plug-ins. The STONITH daemon locks itself into memory, and the C based plug-ins are preloaded by the stonith daemon so that no disk I/O is required for a STONITH operation using C methods. The STONITH daemon provides full support for asymmetric STONITH configurations, allowing for the possibility that certain STONITH devices can be accessible from only a subset of the nodes in the cluster.¹

15.3 Linux migration

There are several sources of additional information to study after deciding to migrate from the Solaris environment to one using Linux-HA. There is a tutorial explaining the basics of how to set up the various components of Linux-HA available on the Linux-HA Web site:

<http://linux-ha.org/GettingStarted>

For more information about how to get started configuring Heartbeat, refer to the documentation available at the Linux-HA Web site:

<http://linux-ha.org/GettingStartedWithHeartbeat>

This page is particularly good because it shows various example configurations for many popular cluster environments.

Kernel

Linux-HA runs well on any distribution using Linux kernel version 2.6 or later. This includes all current Red Hat and Novell SUSE kernels. There are also no kernel dependencies or hooks.

15.4 Cluster environment considerations

Linux-HA clustering is designed to recover from single-system faults. For more complete and up-to-date information about how to implement HA in a clustered environment, refer to the sources mentioned in 15.3, “Linux migration” on page 208.

15.4.1 Data sharing arrangements

Linux-HA has no special shared disk requirements. It supports (at least) the following data sharing configurations:¹

- ▶ No shared data
- ▶ Replication (DRBD, or application-specific)
- ▶ SCSI RAID controllers supporting clustering (IBM ServeRAID™, ICP Vortex)
- ▶ External RAID units: SCSI, Fibre Channel, any kind

The only requirements on shared disk is that it support mount and umount. More specifically, it does not rely on SCSI reservations (or their equivalent).

15.4.2 IBM ServeRAID

Certain of the IBM ServeRAID SCSI RAID controllers provide support for access to a given RAID volume by either one of two machines at a time. In this environment, two ServeRAID controllers can be connected to the same SCSI array, and then either (but not both) of the two machines can access the data.

For more specifics about the supported IBM ServeRAID controllers and their configuration, refer to the Linux-HA Web site:

<http://linux-ha.org/ServeRAID>



Shell scripting

This chapter discusses the common shells that system administrators use for their day-to-day work.

This chapter includes the following topics:

- ▶ Overview of the shell environment
- ▶ Public Domain Korn shell
- ▶ Moving from ksh to bash

16.1 Overview of the shell environment

Shells are standard command interpreters for the UNIX system. Commands are entered through the terminal or from a file. This is not a complete description of what a shell is. We discuss only the most commonly used shells in this section.

16.1.1 Solaris shell environments

Solaris 8, when installed, comes with the following shells:

sh	Bourne shell
ksh	Korn shell
csch	C shell
rsh	Restricted Bourne shell
rksh	Restricted Korn shell

There are other shells that can be installed from the Companion CD.

Solaris 9, in addition to the previous shells, also comes with:

bash	Bourne-Again shell
tcsh	Tenex C shell
zsh	Another shell option that includes a combination of features found in bash, ksh, and tsch

The default shell provided when a new user is created, and the shell parameter is not used, is sh (Bourne shell).

16.1.2 Linux shell environments

The Bourne-Again Shell (bash) is the default shell for Linux. The bash shell consists of the features of sh and ksh, plus a bit more. For more information about bash, see the man pages.

Linux also comes with other shells: csch, tcsh, zsh, and so on.

Most of these shells are dynamic executables that rely on shared library files. In a disaster scenario when these libraries are not available, the shell cannot be started. In Linux, a file called `/bin/ash.static` is a statically linked binary that does not rely on any external shared libraries. This shell can be used as the shell when the system is in a recovery mode and when the normal shells do not work as intended.

For a description about how to use this file, refer to Chapter 17, “Troubleshooting” on page 215.

16.2 Public Domain Korn shell

The Linux equivalent of the Korn shell (ksh) is the Public Domain Korn shell (pdksh). Although it is intended to be a POSIX-compliant Korn shell, it is not fully conforming at this time.

The bash shell has most of the features and functionality of ksh, and is the recommended migration path in the Linux environment.

16.3 Moving from ksh to bash

Migrating your sh and ksh scripts to bash is typically a straightforward process. However, there is a minor difference to be aware of. If you are using the **read** statement in your script, as shown in Example 16-1, the second line **echo** results in a blank line. This is a known issue in bash and applies to both single and multiple variable reads.

Example 16-1 Sample read statement

```
> echo "A B C" | read x y z
> echo $x $y $z

> _
```

Here are some suggested workarounds for the read command:

- ▶ Work around 1: Using **echo**, as shown in Example 16-2

Example 16-2 Using a redirected echo to a read

```
> read x y z <<(echo "A B C")
> echo $x $y $z
A B C
> _
```

- ▶ Work around 2: Using `awk`, as shown in Example 16-3

Example 16-3 Using awk

```
> x='echo "A B C" | awk '{ print $2 }''  
> echo $x  
B  
> _
```

For more information about using bash, refer to the following Web site:

<http://www.gnu.org/software/bash/>



Troubleshooting

This chapter describes some differences in the troubleshooting process of Linux and Solaris.

We discuss the following topics:

- ▶ Troubleshooting the booting process
- ▶ Core files
- ▶ Crash dumps
- ▶ Logs
- ▶ Permissions: File access problems
- ▶ Printing
- ▶ File systems
- ▶ Packages
- ▶ root password recovery
- ▶ Network
- ▶ System and user processes
- ▶ Diagnostic and debugging tools

17.1 Troubleshooting the booting process

Sometimes a system will not boot properly. It might hang at a point in the boot process or reboot continuously.

Solaris boot troubleshooting

In Solaris, you can perform interactive boots, device reconfiguration boots, single user boots, and verbose boots in order to correct a faulty system that does not boot anymore. If the system is severely damaged, and the file systems are corrupted, you can boot from an external device such as CD/DVD or the network in order to repair the system.

Linux boot troubleshooting

In Linux, you can also perform single user boots or verbose boots for maintenance purposes using the already installed kernels, or if the local kernels or **initrd** images are corrupted or file systems are broken, you can boot from a CD, DVD, USB flash, or network in rescue mode in order to repair the system. To boot in single user mode, you need to add **1** at the end of kernel parameters at GRUB prompt, and the **init** will use that level instead of the default found in the `/etc/inittab` configuration file. To boot in verbose mode, you have to remove **rhgb quiet** from the kernel parameters at the GRUB prompt in Red Hat, or just press Esc when booting in Novell SUSE. If your `/etc/inittab` file is corrupted and you cannot boot in any run level, you can use the **emergency** kernel parameter that will tell **init** to ignore the `/etc/inittab` file and launch the default shell after prompting for the root password. If there is a major problem with your installed shared libraries, you might find that the **init** program or default shell are not working because they are dynamically linked to the respective corrupted libraries. In this case, you can boot the system in a very minimal single user mode by using **init=/bin/ash.static** as the kernel parameter, and it will run this static compiled shell instead of the **init** program. Then, you can use several other static binaries to correct your system. For more information about possible kernel parameters, refer to the man page for **bootparam**, and check your vendor documentation.

For rescue booting, you just need to boot from an installation CD, DVD, USB flash, or network and follow the on-screen instructions. Some vendors also provide a rescue mode CD to be used only for rescue.

17.2 Core files

Core files are created when an application or process terminates abnormally. These core files are used by the developers to determine the cause of abnormal

termination and to troubleshoot the respective program that generated the core file.

Solaris management of core files

In Solaris, core files are managed with the **coreadm** command. You can have a per-process core file path, which is enabled by default, and it creates a core file in the current directory. In addition, you can create a global core file path, which is disabled by default, and it creates another core file in the global directory you specify. The per-process core is created with read/write rights only for the owner of the process; the global core is created with read/write rights for the root only. You can also expand the default core file name to include useful information such as the PID, EUID, and EGID. *Setuid* programs do not produce a core by default, but this behavior can be changed with the **coreadm** command. Then, you can use the **pstack**, **pmap**, **p1dd**, **p1lags**, and **pcrd** tools to show information about the core file. You can inspect core files with the **mdb** utility.

Linux management of core files

In Linux, we do not have a per-process core files and a global core file as in Solaris, but a single core file and its generation is disabled by default. To enable core file generation, you need to use the **ulimit** command to change the maximum accepted core file size from 0 to something more reasonable. This setting affects only the programs started from that shell. To set this option permanently, you can modify an initialization script such as `/etc/profile` or `.bash_profile`.

To customize the core file name and path, use the **sysctl -w** command to dynamically change the value of the **kernel.core_pattern** attribute, or if you need a permanent setup, you can add your settings to the `/etc/sysctl.conf` file and run **sysctl -p**. For more information about setting core file names and path, refer to the man pages for **proc**, and search for **core_pattern**.

There is also a **kernel.core_uses_pid** attribute that is obsolete and is maintained only for compatibility reasons. If you need PID in your core file names, use **kernel.core_pattern** instead with the `%p` pattern. You should set **kernel.core_uses_pid** to 0 using the same methods described earlier.

Example 17-1 shows how to enable core file generation and modification of the default core pattern to store the core files in `/core/core.program_name.pid`.

Example 17-1 Example of core file setup

```
ulimit -c 1000000
sysctl -w kernel.core_uses_pid=0
sysctl -w kernel.core_pattern=/core/core.%e.%p
```

Table 17-1 shows the differences between the Solaris and Linux core file patterns.

Table 17-1 Core file patterns

Core pattern definition	Solaris	Linux
Process ID	%p	%p
Effective user ID	%u	%u
Effective group ID	%g	%u
Executable file name	%f	%e
System node name (<code>uname -n</code>)	%n	%h
Machine name (<code>uname -m</code>)	%m	N/A
Time stamp	%t	%t
Number of signal causing dump	N/A	%s
Literal %	%%	%%

17.3 Crash dumps

When a system crashes, it can save an image of the physical memory at the time of crash on the default dump device for further analysis. System crashes can be due to from software problems or hardware problems.

Solaris management of crash dump files

In Solaris, when a system crashes, it writes a copy of the physical memory to the dump device. Then, after reboot, the `savecore` command is executed to save the crash dump files from the dump device to the configured `savecore` directory in the form of `unix.X` and `vmcore.X` files, where `X` is the dump sequence number. The crash dump files are used by Sun support to diagnose the problem. The default directory for saving crash dump files is `/var/crash/hostname`. You can inspect crash dump files with the `mdb` utility. The default dump device is the swap space of the system, but it can be changed to a separate partition. You can generate a crash dump of a running system with the `savecore -L` command if you configured a separate dump partition. Managing the crash dump configuration and content can be modified with the `dumpadm` command.

Linux management of crash dump files

The management of crash dump files is handled differently in Red Hat and SUSE.

Crash dump files in Red Hat

Red Hat has two options for managing crash dump files: **diskdump** for disk-based dump files and **netdump** for network-based dump files.

The **diskdump** is a standard service in the `/etc/init.d/diskdump` directory and has the `/etc/sysconfig/diskdump` configuration file. This service is disabled by default, and you need to manually configure and activate it. When a crash occurs, a dump is stored in the partition specified in the `/etc/sysconfig/diskdump` configuration file. Then, use the **savecore** utility to read the data from the dump partition. This creates a **vmcore** file in a directory named `/var/crash/127.0.0.1-timestamp`. The **vmcore** crash dump file can be analyzed with the **crash** utility that is provided in a separate package.

For network-based dump, Red Hat uses the **netdump** package on the client machine that will generate the dump, and the **netdump-server** package on a reliable server that will receive and host the dump files. The **netdump** is a standard service in `/etc/init.d/netdump` and has the `/etc/sysconfig/netdump` configuration file. This service is disabled by default, and you need to manually configure and activate it. In order to use this service, you need to configure the `/etc/netdump.conf` file and start the **netdump-server** service on a reliable server in your network that will receive the oops data and memory dumps from the client and store them in the default `/var/crash` directory. You can instruct **netdump-server** to execute certain scripts upon receiving a dump data.

The dump data can be inspected with the **crash** utility that is provided in a separate package.

For more information, check the **diskdumpctl**, **diskdumpfmt**, **savecore**, **crash**, **netdump**, and **netdump-server** man pages and the vendor documentation.

Crash dump files in Novell SUSE

SUSE manages crash dump files using the Linux Kernel Crash Dump (LKCD) utility that provides the ability to collect, save, and analyze information about the kernel when the system crashes. After LKCD is configured in the `/etc/sysconfig/dump` file, a crash dump is created whenever a kernel panic or kernel oops occurs, or on demand by using a specific key combination. The default settings for the dump is to pick up the dump headers and kernel pages only. In addition, the default dump device is the network. When a kernel panic or oops occurs, the following steps are performed:

1. Saves system memory to the dump device.
2. Reboots the system.
3. Runs the LKCD configuration, which prepares the system for the next crash.

- Creates dump files on the permanent dump storage area from the dump device.

The resulting dump can be analyzed with the **1crash** tool that is part of this package. And last but not least, LCKD provides for both local and network storage of dumps.

For more information about the Linux Kernel Crash Dump utility, refer to the following Web site and the and SUSE documentation:

<http://1kcd.sourceforge.net>

Table 17-2 shows the main differences between Solaris and Linux crash dump management.

Table 17-2 Crash dump management

Topics	Solaris	RHEL	SLES
Disk crash dump utilities	dumpadm savecore	<code>/etc/init.d/diskdump</code> savecore	1kcd
Disk crash config files	<code>/etc/dumpadm.conf</code>	<code>/etc/sysconfig/diskdump</code>	<code>/etc/sysconfig/dump</code>
Network crash dump utilities	N/A	<code>/etc/init.d/netdump</code> <code>/etc/init.d/netdump-server</code>	1kcd
Network crash config files	N/A	<code>/etc/sysconfig/netdump</code> <code>/etc/netdump.conf</code>	<code>/etc/sysconfig/dump</code>
Crash analyzing tools	mdb	crash	1crash
Crash dump default store directory	<code>/var/crash</code>	<code>/var/crash</code>	<code>/var/log/dump</code>

17.4 Logs

System logs are very useful for troubleshooting. The logs are the first thing to check if something is going wrong. Even if the system is stable and running just fine, it is good to inspect the system logs periodically. There might be an indication of symptoms that will potentially affect the availability of the system.

Logs in Solaris

Solaris uses the **syslogd** daemon to record various system warnings and errors. The **syslogd** daemon is configured by default to send the most critical messages to the console in addition to logging them to files. Most system messages are

stored by default in the `/var/adm/` directory, and the file to check is `/var/adm/messages`.

The system logs are rotated by the `logadm` (in Solaris 9) and `newsyslog` (in Solaris 8) command, which has an entry in the root crontab file.

The `syslogd` configuration file is `/etc/syslog.conf` and it can contain macros and logic blocks. It also has a `/etc/default/syslogd` configuration file for changing the daemon behavior globally.

To see the system diagnostic messages from the boot time, use the `dmesg` command.

Logs in Linux

Linux also uses the `syslogd` daemon to record various system warnings and errors. The `syslogd` daemon is configured by default to send the most critical messages to the console in addition to logging them to files. Most system messages are stored by default in the `/var/log/` directory, and the file to check is `/var/log/messages`.

The system logs are rotated by the `logrotate` program, which has an entry in the `/etc/cron.daily/` directory and the configuration is stored in the `/etc/logrotate.conf` file.

The `syslogd` configuration file is `/etc/syslog.conf` and it has a syntax similar to Solaris with the exception that it cannot contain macros and logic blocks. It also has a `/etc/sysconfig/syslog` configuration file for changing the daemon behavior globally.

In Linux, there is also a `klogd` kernel log daemon, which has no equivalent in Solaris. It intercepts and logs Linux kernel messages from `/proc/kmsg` and logs it directly or through the `syslogd` daemon. For more information, refer to the man page of `klogd`.

To see the system diagnostic messages from the boot time, use the `dmesg` command or look in `/var/log/boot.log` (in Red Hat) or `/var/log/boot.msg` (in SUSE).

The default behavior for SLES is to split the messages for all facilities into files based on the priority of the messages. This is set in the `/etc/syslog.conf` file and described in the `syslog.conf` man page. So, unless `auth` and `cron` have specific entries in the `/etc/syslog.conf` file, their messages will be directed to files based on the priority of the messages.

17.5 Permissions: File access problems

In this section, we discuss problems accessing files and file systems. There are many reasons why a file cannot be accessed. File access problems can be as simple as a bad PATH environment variable or as complex as a non-existent UID. File systems might not be mounted or might be mounted read-only.

17.5.1 Problem: Command not found

The system-wide PATH environment variable is set in /etc/profile. Each user ID can use the system-wide PATH as it is defined or can modify it in their login profile script. Login profile script names are different for each shell. A user's login shell is defined in /etc/passwd. Refer to the man page for the shell to determine the login profile name. Table 17-3 shows file location commands.

You can display the current PATH with this command:

```
echo $PATH
```

The **locate** command is different from the **find** command. First, the **locate** command can be used to find any file or directory, not just executable files. Second, **locate** is much faster than the **find** command because it uses a file name database. The **updatedb** command is used to initialize and maintain the file name database. You can run the **updatedb** command at any time to update the file name database. If you want to schedule the **updatedb** command to run one or more times daily, pick times when there is less activity on the server. The **updatedb** command can run for a long time as it searches the file systems on the server. Refer to the man page for **updatedb** for configuration options.

Table 17-3 File location commands

Task	Solaris	Linux
Display \$PATH settings	echo \$PATH	echo \$PATH
Search \$PATH for first occurrence of executable <i>filename</i>	which	which
Search file system or directory tree for <i>filename</i>	find	find
Update the file name database	N/A	updatedb
Search the file name database for <i>filename</i>	N/A	locate
Display mounted file systems and mount options	mount	mount

17.5.2 Problem: File access

File permissions are based on owner, group, and guest access permissions. These are defined in a file's access control list (ACL). The simplest way to display file permissions is with the `ls -l` command.

The `chown`, `chgrp`, and `chmod` are the same for Solaris and Linux. Refer to the corresponding man pages for each command for more information.

The `stat` command can give you more detailed information about a file. The `stat` command reports such items as the access permissions, the last time the file was accessed or modified, and the size of the file.

There might be times when a file has been opened by another application and you can only open it in read-only mode. Linux provides the `lsof` and `fuser` commands to list the open files and the PID associated with the file, as shown in Table 17-4.

Table 17-4 File access commands

Task	Solaris	Linux
Display file name, permissions, and owner	<code>ls -l</code>	<code>ls -l</code>
Display detailed information for a file or file system	N/A	<code>stat</code>
Display open files	<code>proc tools</code>	<code>lsof</code>
Display processes using files or sockets	<code>fuser</code>	<code>fuser</code>

17.6 Printing

This section covers basic troubleshooting of print subsystems. For more detailed information about managing print queues and print jobs, refer to your Linux distribution's administration guides.

Table 17-5 lists basic troubleshooting commands.

Table 17-5 Print service troubleshooting commands

Print service troubleshooting task	Solaris	Linux
Display print queue status	<code>lpstat</code> <code>lpc</code>	<code>lpc</code> <code>lpq</code>
Display settings of a print queue	<code>lpadmin</code>	<code>lpadmin</code> <code>lpoptions</code>
Display print service status information	<code>lpstat</code>	<code>lpstat</code>

Print service troubleshooting task	Solaris	Linux
Display printing config information from the system config database	lpget	N/A
Display available devices or drivers	N/A	lpinfo

17.6.1 Troubleshooting remote printer connectivity

The technique used to debug network connectivity to a remote printer is to try to make a connection to the printer port. There are several commands that can be used to verify network connections. We demonstrate the **telnet** and **netcat** commands in this section.

The default port for the **lpd** service is 515. The default port for the **cupsd** service is 631.

Use either of the following commands to test network connections to remote printers:

```
telnet hostname portnumber
netcat -v -z hostname portnumber
```

Both commands will return “connection refused” if no services are registered with the supplied port number. Any other output will mean that a service is registered with the supplied port number.

After you have determined the problem is not with the network, troubleshoot the printer locally at the server.

17.6.2 Troubleshooting local printers

This section covers troubleshooting locally attached printers.

The basic procedure to troubleshoot a print service is to review the error log and fix any problems noted in the log. Perform the following steps:

1. Stop the print service.
2. Back up, rename, or delete the print service’s error logs.
3. For CUPS, set `LogLevel` to debug in the configuration file `/etc/cups/cupsd.conf`.
4. Start the print service.
5. Retry the action or operation that causes the problem.
6. Correct any problems noted in the error log.

LPD

Access permissions are located in `/etc/lpd.perms`.

CUPS

Access permissions and other configuration parameters are in `/etc/cups/cupsd.conf`.

Log files are in the `/var/log/cups` directory.

CUPS provides online manuals at the following location. The *Software Administrator's Manual* contains a troubleshooting section along with installation and configuration sections.

[http://localhost:6\(2\)31/documentation.html](http://localhost:6(2)31/documentation.html)

17.7 File systems

If your server crashes, or has somehow powered off without actually flushing the buffer and unmounting the file systems, you will have some *dirty* file systems and might end up with inconsistencies in those file systems. That is why on each boot, the initialization scripts run **fsck** on selected file systems from `/etc/vfstab` in Solaris or `/etc/fstab` in Linux. If they were not unmounted properly, the **fsck** program will scan for inconsistencies. Usually, the **fsck** program runs in a non-interactive mode, automatically fixing minor inconsistencies. However, if there are some major problems, it switches to manual mode and lets the operator make the decisions.

If you are in a situation where you need to manually fix the file system with **fsck**, it is a good idea to back up the partition before any attempt to fix the file system, especially if you have valuable data on it. In both operating systems, you can run **fsck -y** to automatically fix any type of problem, be it minor or major.

In Solaris 9 and all current Linux distributions, the default file system is journalized unless the administrator chooses something else. This means that even if the system crashes, or is abruptly powered off, the mounted file systems will not be *dirty* because of this transactional log that is the journal. In this case, it is still a good practice to do a forced **fsck -f** at scheduled intervals, just to be sure that everything is fine. This is because the journalization will not prevent a kernel or other software bug from causing a file system inconsistency.

One other problem with disks is that they can develop bad blocks. If this is the case, you can scan for bad blocks in Linux by using the **badblocks** command, and then you can instruct **fsck** to update the bad block list of the existing file system.

Important: In Linux, it is a good practice to use the specific `fsck.filesystemtype` program instead of the generic `fsck`, or if you use the generic `fsck`, always use the `-t fstype` switch.

17.7.1 Remote file systems

You might notice that a remote file system has not mounted, or that it has not mounted properly. Consider the following troubleshooting methods.

At the server:

- ▶ Check `/etc/exports` for access permission.
- ▶ Verify that the `rpcbind/portmap` service is running. NFSv2 or NFSv3 require the `rpcbind` service in Solaris and the `portmap` service in Linux.
- ▶ Verify that the `nfsd` or `nfsserver` process is running.
- ▶ Display registered `portmap` processes.
- ▶ Use the `showmount` command to display exported subdirectories.

At the client:

- ▶ Check `/etc/fstab` for default mount options.
- ▶ Check `/etc/mstab` for current mount options.
- ▶ Ping the target server.

Table 17-6 shows some of the commands used in this process.

Table 17-6 Remote file system troubleshooting commands

Task	Solaris	Linux
Display mounted file systems	<code>df</code>	<code>df</code>
Display detailed information about mounted file systems	<code>mount</code>	<code>mount</code>
Display NFS server's export list	<code>showmount -e hostname</code>	<code>showmount -e hostname</code>
Display rpcbind/portmap server PID	<code>pgrep rpcbind</code>	<code>pgrep portmap</code>
Display registered rpcbind/portmap processes	<code>rpcinfo -p hostname</code>	<code>rpcinfo -p hostname</code>

17.7.2 Software RAID

In Linux, the status of software RAID partitions is maintained in `/proc/mdstat`. If you find that a software RAID array is damaged:

1. Partition a new drive identically to the damaged drive.
2. Shut down your Linux system and power off if necessary.
3. Replace the defective drive with the new drive.
4. Boot the server.
5. Issue the `raidhotadd` command.

17.7.3 Logical volumes

LVM2 is the default Linux logical volume management utility. Basic operations on volume groups (`vg*`) and logical volumes (`lv*`) are `vgcreate/lvcreate`, `vgextend/lvextend`, and `vgreduce/lvreduce`. Table 17-7 lists troubleshooting tasks and the associated command.

Table 17-7 Logical volume troubleshooting tasks

Logical volume troubleshooting task	Solaris	Linux
Check volume group metadata	<code>metadb -i</code> <code>metastat</code>	<code>vgck</code>
Scan disk drives and rebuild caches	N/A	<code>vgscan</code>
Activate/deactivate volume groups	N/A	<code>vgchange</code>
Display volume group status	<code>metastat</code>	<code>vgdisplay</code>
Scan disk drives for logical volumes	N/A	<code>lvscan</code>
Activate/deactivate logical volumes	N/A	<code>lvchange</code>
Display logical volume status	<code>metastat</code>	<code>lvdisplay</code>

17.8 Packages

This section covers troubleshooting software packages. We describe the following tasks in this section: removing packages, upgrading packages, and verifying the installation of packages. Table 17-8 on page 228 illustrates commands used to accomplish these tasks.

Table 17-8 Package troubleshooting tasks

Package management tasks	Solaris	Linux
Remove software package	<code>pkgrm</code>	<code>rpm -e</code>
Upgrade/install a package	<code>pkgadd</code>	<code>rpm -U</code> <code>rpm -F</code>
Verify correct installation	<code>pkgchk</code>	<code>rpm -V</code>
Verify digest and digital signature	N/A	<code>rpm -K</code>

Solaris package troubleshooting

The Solaris package format is based on the System V Interface Definition for Application Binary Interface (ABI) and has tools for managing these packages. These management tools include the ability to add and remove packages, check for consistency, and display package information. If system administrators use the `pkgadd` and `pkgrm` tools to manage packages, the tools update the software product database accordingly. You can use `pkgchk` command to perform a consistency check on an installed package and `pkgparam` to see the parameters of an installed package. Other useful information is found in the `/var/sadm/install/contents` file and the `/var/sadm/pkg/` directory.

Linux package troubleshooting

Linux provides the `rpm` command for all aspects of package management.

Testing and verifying installed packages

The `-K` parameter is used to verify digests and digital signatures. The `-V` parameter is used to test or verify an installed package. Up to eight tests are run against the package. The results for the tests, an attribute marker, and the file name from the package are all returned by the command. For example, if you check the `cron` package, you might see output similar to that shown in Example 17-2.

Example 17-2 rpm command -V output

```
myhost:~ #rpm -V cron-3.0.1-920-1
.M..... c /etc/crontab
.M..... /usr/bin/crontab
```

In this case, the file mode test failed because the file mode is different on the file system when compared to the file mode in the package. This particular test compares permissions and the file type. All other tests passed because they returned a single “.”. The letter “c” is the attribute marker. It tells us that `/etc/crontab` is a configuration file.

Refer to the man page for the `rpm` command for a detailed list of options and parameters.

Upgrading installed packages

The `-U` and `-F` parameters are used to upgrade packages. The `-F` parameter upgrades a package only if it is already installed. The `-U` parameter installs the package if it is not installed; otherwise, it upgrades the installed version and then removes all other versions after the upgrade completes.

SUSE provides a debug option with the command `yast2 online_update`. For the full syntax of the command, see *SUSE Linux Enterprise Server Installation And Administration*.

Removing installed packages

The `-e` parameter is used to erase or remove installed packages. The `--nodeps` parameter is useful if you do not want to check for dependencies when removing the package.

Database consistency

You might encounter a situation where the RPM database becomes unusable or corrupt. It can be difficult, if not impossible, to upgrade packages if their corresponding entries in the database cannot be accessed. Two parameters are available for RPM database management:

- ▶ `--initdb` is used to create a new RPM database.
- ▶ `--rebuilddb` is used to rebuild RPM database indexes from existing, installed RPM package headers.

17.9 root password recovery

There are cases when the root password is lost. This section explains how to change the root password in such cases.

Solaris root password recovery

In Solaris, perform the following steps to change a lost root password:

1. Stop the system, or do a power off/on cycle.
2. Boot in single user mode from a CD-ROM/DVD or from a network boot/install server.
3. Mount the root (`/`) file system in read/write mode.
4. Remove the root password from the `/etc/shadow` file and save the file.
5. Unmount the root file system.

6. Reboot the system.
7. The root password now is not set. Press Enter at the password prompt when you login as root.

Linux root password recovery

In Linux, you have two options for recovering a lost root password.

The first option is to add `init=/bin/bash` as kernel parameters at GRUB prompt, and the kernel will launch the bash shell instead of the `init` process. This works only if you can modify the boot parameters, that is, GRUB has no password, or has a known password for configuring boot parameters.

The second option is to boot in rescue mode from a CD, DVD, USB flash media, or from the network. To boot in rescue mode, follow the instructions presented on screen when you boot from CD/DVD media. This works even if local GRUB is password protected and you do not know the GRUB password. You can also change the GRUB password if needed by editing its configuration file.

The rest of steps are the same in both password recovery options:

1. Mount the root (/) file system in read/write mode if not already mounted by the rescue program.
2. `chroot` in the new mounted root (/) file system.
3. Use the `passwd` command to change the root password.
4. `exit` from the chroot shell.
5. Unmount the root file system.
6. Reboot the system.

Tip: In Red Hat, it is enough to boot in single user mode, change to root password with the `passwd` command, and reboot the server.

17.10 Network

Most of the networking tools used for network troubleshooting are the same in both Solaris and Linux, with some minor differences as illustrated in Table 17-9 on page 231.

Table 17-9 Troubleshooting network problems

Task	Solaris	Linux
Display interface settings	<code>ifconfig</code>	<code>ip</code> <code>ifconfig</code>
Display interface status and statistics	<code>netstat -i</code>	<code>netstat -i</code> <code>ifconfig</code>
Configure interface	<code>ifconfig</code>	<code>ip</code> <code>ifconfig</code>
Check various network statistics	<code>netstat</code>	<code>netstat</code>
Check DNS resolver	<code>more /etc/resolv.conf</code>	<code>more /etc/resolv.conf</code>
Check name services config	<code>more /etc/nsswitch.conf</code>	<code>more /etc/nsswitch.conf</code>
Display kernel network parameters	<code>ndd /dev/ip (\?)parameter</code> <code>ndd /dev/tcp (\?)parameter</code>	<code>sysctl -a grep ^net</code>
Configure kernel network promontories	<code>ndd -set driver parameter</code>	<code>sysctl -w var=value</code>
Check for network link	<code>ndd driver link_status</code>	<code>ethtool</code> <code>mii-tool</code>
Query DNS	<code>nslookup</code> <code>dig</code>	<code>nslookup</code> <code>dig</code>
Check routing table	<code>netstat -r</code>	<code>netstat -r</code> <code>route</code>
Check ARP entries	<code>arp -a</code>	<code>arp</code>
Testing for connectivity	<code>ping</code>	<code>ping</code>
Check IP path	<code>traceroute</code>	<code>traceroute</code>
Capturing network packets	<code>snoop</code>	<code>tcpdump</code> <code>tethereal</code> <code>ethereal</code>

Note: In Linux, a network interface is considered up by the kernel even if the network link is not present, so this is the first thing to check if you have network problems.

17.11 System and user processes

Both Solaris and Linux provide the `ps` and `top` commands (`top` is optionally installable on Solaris from the Companion CD), used to display the state of

running processes. The **ps** command gives you a point in time snapshot of the system processes. The **top** command gives you an iterative, interactive display of system processes. Whether you use the **ps** or **top** command, important columns in the output of these commands to monitor are %CPU, STAT, START or STIME, and TIME. Table 17-10 provides an overview of the command sets used to troubleshoot system and user processes.

Table 17-10 Troubleshooting system and user processes

Task	Solaris	Linux
Trace system calls and signals	truss	strace
Library trace	N/A	ltrace
Print shared library dependencies	ldd	ldd
Report IPC status	ipcs	ipcs
Display and manage system resources available to a user or process	ulimit	ulimit
List open files and sockets and their owning user IDs and PIDs	proc tools: pfiles, pmap, ptree, and so on	lsdf
Identify which user or process is using a file or socket	fuser	fuser
Send signals to processes	kill pkill	kill pkill
List current processes	ps	ps
List current processes based on name and other attributes	pgrep	pgrep
List current processes in an interactive, iterative format	top (on companion CD)	top
Report system activity	sar	sar
Display virtual memory statistics	vmstat	vmstat
Display I/O statistics	iostat	iostat
Display system error log	dmesg	dmesg

17.12 Diagnostic and debugging tools

Certain tools are installed by default. Review these packages and install them if your installation requires the tools provided by the packages. The base package

name for some packages will be the same on Red Hat Enterprise Linux and Novell SUSE Linux Enterprise Server. The version might be different. The format of the following lists is:

base package name* : command

For Red Hat Enterprise Linux:

- ▶ `sysreport*` : **sysreport**
- ▶ `hwbrowser*` : **hwbrowser** (graphical tool)
- ▶ `pciutils*` : **lspci**
- ▶ `oprofile*` : **opreport**
- ▶ `gdb*` : **gdb**
- ▶ `strace*` : **strace**
- ▶ `statserial*` : **statserial**

For Novell SUSE Linux Enterprise Server:

- ▶ `sig*` : **sig**
- ▶ `pciutils*` : **lspci**
- ▶ `gdb*` : **gdb**
- ▶ `strace*` : **strace**
- ▶ `ksymoops*` : **ksymoops**
- ▶ `statserial*` : **statserial**

Refer to the man pages and RPM package descriptions for information regarding these commands.

Memtest86

SUSE provides a memory test application available only on x86 systems called **memtest86**. It is available from the boot screen when booting from the CD-ROM. This RAM test runs an endless read and write loop. You should let this test run for several hours. Reboot the system to terminate the test. For more information about this application, refer to the following Web site:

<http://www.memtest86.com>



Part 3

IBM eServer platforms

In the following chapters, we provide an overview of Linux system administrator tasks and issues specific to the IBM `@server` platforms.



IBM eServer xSeries hardware platform specifics

This chapter covers the differences in the migration process as they relate to IBM *eServer* xSeries and *eServer* BladeCenter® server hardware and describes some differences in setting up and managing these systems.

This chapter describes the distinguishing differences in the hardware for the xSeries server platforms and BladeCenter platforms in the following areas:

- ▶ Installation considerations
- ▶ Hardware and device differences

18.1 Installation considerations

Both Red Hat and Novell SUSE Linux distributions are architected to exploit the capabilities of the @server xSeries and BladeCenter servers. The xSeries and BladeCenter server platforms are both Red Hat and Novell SUSE certified.

Before you install Linux, we recommend that you check the BIOS and firmware levels on the following items and update them to the most current revision as part of your installation procedure for the xSeries server:

- ▶ System BIOS
- ▶ Remote Supervisor Adapter firmware and BIOS
- ▶ Onboard diagnostics
- ▶ Additional devices if installed, such as ServeRAID adapters and FASTT Fibre Channel host adapters

For the latest BIOS and firmware code, refer to:

<http://www.ibm.com/pc/support/site.wss/document.do?lnocid=MIGR-4JTS2T>

Follow the installation instructions provided with each package or those included with your server platform regarding updating firmware levels.

For a complete driver listing of the most common server components, refer to:

<http://www.ibm.com/pc/support/site.wss/DRV-R-MATRIX.html>

For more information about the booting and installation process, refer to Chapter 8, “Boot and system initialization” on page 125.

18.1.1 xSeries

Before starting an installation, make sure that any disk subsystems, such as RAID arrays, have already been configured. For more information about how to configure these systems, refer to the documentation included with the disk subsystem hardware.

In general, the xSeries server platform is installed in much the same manner as other IA32-based architecture platforms. The xSeries specifics for installation starts with the selection of the boot device. If the system does not boot from the installation device that contains the Linux distribution, you need to enter the boot setup device screen at startup. Follow the instructions after initially powering on the system for what keys to press to enter the boot device configuration or refer to the system documentation.

18.1.2 BladeCenter

All the blades share the CD-ROM, floppy drive, keyboard, video, and mouse. There are two I/O selection buttons on the front of each blade that allow for assignment of the CD-ROM and floppy drive and also for the keyboard, video, and mouse (KVM).

There is also a power button on each blade that is protected by a hinged plastic flap. After a blade is powered up, pressing the CD-ROM or the KVM button on that blade selects the corresponding resource for that blade. On the blade that is currently connected to the CD-ROM or the KVM, the corresponding I/O selection button lights up solid green.

Sharing the CD-ROM for all the blades is a limitation of installing on multiple blades. Using the CD-ROM, you can install individual blades one at a time; however, that process is time-consuming if you install more than one or two blades. As an alternative, install one blade and configure it to be a network-installed server. Subsequent operating system installations are then performed from that server. To read more about how to do this installation process, refer to the IBM Redbook *IBM eServer BladeCenter, Linux, and Open Source: Blueprint for e-business on demand*, SG24-7034, available at:

<http://www.redbooks.ibm.com/abstracts/sg247034.html>

18.2 Hardware and device differences

Accessing the xSeries server and BladeCenter hardware for initial configuration and monitoring is done through special management hardware included with the system. This special management hardware depends on the purchased server options, but can include the integrated the Baseboard Management Controller (BMC), the Remote Supervisor Adapter II (RSA II), or the BladeCenter Management Module.

Access the management hardware using the available user interfaces such as the Web interface, Management Processor Command Line Interface (MPCLI), Advanced Settings Utility (ASU), and the OSA System Management Bridge (SMBridge) Utility, as well as the ability to **telnet** or **ssh** directly to some models.

For much more in-depth information about the xSeries and BladeCenter management hardware and service processors, refer to the IBM Redbook *IBM eServer xSeries and BladeCenter Server Management*, SG24-6495, available at:

<http://www.redbooks.ibm.com/abstracts/sg246495.html>

IBM Director

For complete server management of xSeries and BladeCenter systems, the recommended tool is IBM Director. With IBM Director, you have complete access to the management hardware in addition to other management tasks such as event management, inventory, and deployment.

IBM Director is available to xSeries platform customers from:

<http://www.ibm.com/pc/support/site.wss/MIGR-57057.html>

18.2.1 xSeries

This section provides functional details for IBM @server xSeries Baseboard Management Controller and Remote Supervisor Adapter II.

Baseboard Management Controller (BMC)

Many xSeries servers have service processors integrated onto the system board. These provide different levels of monitoring and alerting depending on the type and functionality of service processor.

Functionality

The integrated BMC might have all or part of the following functionality:

- ▶ Monitoring of system voltages
- ▶ Battery voltage monitor
- ▶ System temperature monitors
- ▶ Fan speed control
- ▶ Fan tachometer monitor
- ▶ Power Good signal monitor
- ▶ System ID and planar version detection
- ▶ System power control
- ▶ System reset control
- ▶ NMI detection
- ▶ SMI detection and generation
- ▶ Serial port text redirection
- ▶ Remind button detection
- ▶ System LEDs control (power, HDD activity, alert, and so on)
- ▶ Control of Lightpath LED

For more information about the full functionality of the BMC, refer to the xSeries system documentation that came with the server.

External connection

The BMCs communicate through one of the integrated Ethernet adapters on the server. To communicate with the BMC, attach a standard Ethernet cable.

Configuration

There are two different methods for configuring the BMC. The BMC configuration can be done through the System Setup in BIOS (pressing F1 at boot time) and accessing BMC settings through the Advanced Options menu. Using System Setup in BIOS is the recommended method.

After the server network settings are configured, you can use the Management Processor Assistant (MPA) task in IBM Director to configure the other required settings such as user IDs, passwords, and alert destinations in the BMC.

The second configuration method uses the **bmc_cfg.exe** utility. This utility is located on the BMC firmware update diskette or CD. You can only run **bmc_cfg** by exiting to a MS-DOS-based program after booting the server from the bootable BMC management firmware update diskette or CD.

Remote Supervisor Adapter II (RSA II)

The Remote Supervisor Adapter II (RSA II) is an advanced management adapter for xSeries server platforms. It provides many options for alerting, monitoring, and remote management of xSeries servers. It replaces the earlier Remote Supervisor Adapter (RSA or RSA I) and the Advanced System Management PCI Adapter (ASMA).

Functionality

Some of the features and functionality of the RSA II include:

- ▶ Automatic notification and alerts
- ▶ Continuous health monitoring and control
- ▶ Event log
- ▶ LAN and Advanced System Management (ASM) interconnect remote access
- ▶ Operating system failure screen capture
- ▶ Remote media
- ▶ Remote power control
- ▶ Server console redirection

Connections

The RSA II supports various connections available on the card and on breakout connectors. Refer to Figure 18-1 for connector locations. There are variations on the location of the connectors available depending on the model of adapter and xSeries server. For specific connector locations on your adapter, refer to the system documentation that came with the xSeries server hardware.

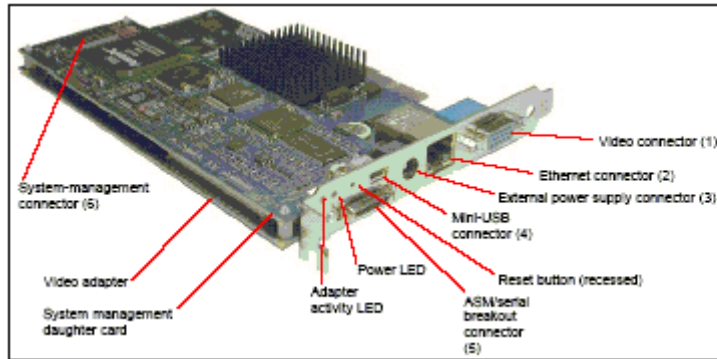


Figure 18-1 xSeries Remote Supervisor Adapter II (RSA II)

The RSA II has the following connectors (the numbers refer to Figure 18-1):

- ▶ Video connector (1 in Figure 18-1): The RSA II contains an additional video subsystem on the adapter. When the RSA II is installed in a server, it automatically disables the on-board video. The server's monitor should be connected to the RSA II video connector.
- ▶ 10/100 Ethernet connector (2): For connection to a 10 Mbps or 100 Mbps Ethernet-based client LAN or management LAN.
- ▶ Power connector (3): Access to RSA II is possible if the server is powered down when the external power supply is used (supplied when the adapter is purchased as an option). Connect the power supply to a different power source as the server (for example, a separate UPS).
- ▶ Mini-USB connector (4): This port provides the ability for a remote keyboard and mouse when using the remote control feature. Connect this to a USB port of the server, except for the following servers where the cable should not be connected (the USB signal is transmitted inside these servers):
 - xSeries 225
 - xSeries 226
 - xSeries 365
 - @server 326

- ▶ Breakout connector (5): To use RSA II as a focal point for an ASM network or for connecting a modem, a breakout cable is supplied, which has the ASM and serial connections. The breakout cable has two serial connectors (earlier RSA II adapters had only one serial port) and two RJ45 connectors for daisy chaining the ASM RS-485 interconnect network.
- ▶ 20-pin connector for the connection to the server's motherboard (6): Figure 18-1 on page 242 shows the connector on the planar where the supplied cable should be connected.

Accessing the RSA II

RSA II provides access to the server for the system administrator, even when the server is down. Monitoring alerts and management functions are performed directly with the RSA II service processor through one of the server's Gigabit Ethernet ports.

The Advanced System Management (ASM) network is for interconnectivity of older service processors with the RSA II. The RSA II can work as a gateway for the ASM network. It can provide access to the connected management processors (that have no Ethernet connection) from the LAN and is the gateway for the management processors to the LAN.

For more information about how to configure the RSA to act as a gateway to the ASM, refer to the IBM Redbook *IBM @server BladeCenter, Linux, and Open Source: Blueprint for e-business on demand*, SG24-7034, available at:

<http://www.redbooks.ibm.com/abstracts/sg247034.html>

Management Processor Command line Interface (MPCLI)

The Management Processor Command Line Interface (MPCLI) is a management tool for xSeries servers running Linux. The system management functions are provided from a command line interface (CLI) that connects to some models of the RSA II service processor in the server. Refer to the server documentation for specific information about this support.

Using this CLI, you can access and set a wide range of information about the health, configuration, communication, and state of your system. These functions are available after you install the CLI and make a connection to the service processor. You can use the MPCLI on a remote server provided you know the IP address of the remote service processor and have a valid user ID and password.

For more information about how to configure user IDs and passwords for this service, refer to the IBM Redbook *IBM @server BladeCenter, Linux, and Open Source: Blueprint for e-business on demand*, SG24-7034, available at:

<http://www.redbooks.ibm.com/abstracts/sg247034.html>

MPCLI supports three methods of communicating with a service processor:

- ▶ In-band communication using a device driver
- ▶ Out-of-band communication using an IP connection
- ▶ Out-of-band communication using an RS-485 interconnect

For more information and to download the MPCLI, refer to following link:

<http://www.ibm.com/pc/support/site.wss/MIGR-54216.html>

Web interface

The Remote Supervisor Adapter II and BladeCenter Management Module have a built-in Web server that enables users to access these service processors using a Web browser.

The browser must be Java enabled, support JavaScript™ 1.2 or later, and have a Java 1.4.1 or later plug-in installed. For best results when using the Web browser, ensure that your monitor's resolution is at least 800x600 and at least 256 colors.

18.2.2 BladeCenter

The BladeCenter system provides shared resources to all the blades, such as power, cooling, system management, network connections, CD-ROM, floppy, keyboard, video, and mouse. The use of common resources allows the blades to be smaller and reduces the need for cabling.

BladeCenter consists of a rack-mounted chassis. The front of BladeCenter supports 14 blade server slots and has a CD-ROM drive, USB port, and floppy drive. The back of the chassis has slots for two blower modules, four power modules, four network modules, and a Management Module.

BladeCenter Management Module

The BladeCenter Management Module has similar capabilities to the RSA II. There are some additional BladeCenter-specific features such as the integrated KVM switch. The BladeCenter Management Module acts like a global RSA II for all the installed blade servers in the chassis. See “Remote Supervisor Adapter II (RSA II)” on page 241 for more information about the features and functionality of the BladeCenter Management Module.

The Management Module manages the BladeCenter chassis itself with all its networking modules (for example, Gigabit Ethernet or SAN) and all blade servers installed in the chassis. The Management Module has an integrated KVM switch and an integrated network switch for internal IP connections to all the modules, such as Ethernet switch modules (ESMs) and Fibre Channel switch modules, to

manage the blade servers. Additionally, it acts like a RSA II for every installed blade server.

For more information about how to connect and configure the Management Module, refer to “Management Module configuration” on page 264.

18.3 References

The following links provide more specific information about the @server xSeries and BladeCenter server hardware:

- ▶ From this chapter:
 - The latest BIOS and firmware code is available at:
<http://www.ibm.com/pc/support/site.wss/document.do?lnocid=MIGR-4JTS2T>
 - For a complete driver listing of the most common server components, refer to:
<http://www.ibm.com/pc/support/site.wss/DRV-R-MATRIX.html>
 - The IBM Redbook *IBM @server BladeCenter, Linux, and Open Source: Blueprint for e-business on demand*, SG24-7034, available at:
<http://www.redbooks.ibm.com/abstracts/sg247034.html>
 - The IBM Redbook *IBM @server xSeries and BladeCenter Server Management*, SG24-6495, available at:
<http://www.redbooks.ibm.com/Redbooks.nsf/RedbookAbstracts/sg246495.html>
 - IBM Director is available to xSeries platform customers from:
<http://www.ibm.com/pc/support/site.wss/MIGR-57057.html>
 - For more information and to download the MPCLI, refer to the following link:
<http://www.ibm.com/pc/support/site.wss/MIGR-54216.html>
- ▶ Additional resources:
 - More information about the IBM @server xSeries server platforms:
<http://www.ibm.com/servers/eserver/xseries>
 - For more information about the IBM BladeCenter server platform:
<http://www.ibm.com/servers/eserver/bladecenter>
 - For more information about the IBM Director management application:
http://www.ibm.com/servers/eserver/xseries/systems_management/ibm_director



IBM POWER technology hardware platform specifics

In this chapter, we provide an overview of Linux system administrator tasks and issues specific to the IBM POWER5™ technology for IBM @server and BladeCenter JS20 hardware platforms.

We discuss the following topics:

- ▶ Planning
- ▶ Booting and system initialization
- ▶ Linux installation
- ▶ eServer i5 and eServer p5 virtualization
- ▶ POWER technology platform service and productivity tools
- ▶ References and further readings

In this chapter, we provide guidance for basic system administration. Refer to the references listed in 19.6, “References and further readings” on page 304, for more POWER technology-based, platform-specific system administration information.

19.1 Planning

This section describes issues that system administrators should consider before beginning the installation of an @server i5, @server p5, or @server BladeCenter JS20.

19.1.1 IBM eServer i5 and eServer p5

There are two general paths available for installing Linux on IBM @server i5 @server p5 systems depending on your configuration choice of a stand-alone system or a partitioned system.

Monolithic (stand-alone)

In a monolithic installation, Linux owns the entire server and all of its resources. This installation is only common for @server p5 systems. A monolithic install does not require any POWER technology-specific preinstallation planning.

Note: The @server p5 system is initially preconfigured for monolithic operation.

Hosted (partitioned)

In a hosted installation, Linux runs in a partition, along with other instances of Linux. By using logical partitioning (LPAR), a single physical system can be divided into multiple logical partitions, each running their own operating system image. The @server i5 and @server p5 both support hosted installations.

Concepts for Linux logical partitions

The purpose of this information is to familiarize you with the hardware and software required for Linux logical partitions.

Logical partitioning is the ability to make a server run as though it were two or more independent servers. When you logically partition a server, you divide the resources on the server into subsets called logical partitions. Processors, memory, and input/output devices are examples of resources that you can assign to logical partitions. You can install software on the logical partition, and the logical partition runs as an independent logical server with the processor, memory, and I/O resources that you have allocated to it.

You must use tools to partition @server i5 and @server p5 servers. The tool that you use to partition each server depends on the server model and the operating systems and features that you want to use:

- ▶ **Hardware Management Console (HMC)**

The Hardware Management Console (HMC) is a hardware appliance that connects to the server firmware. You use the HMC to specify to the server firmware how you want resources to be allocated among the logical partitions on the managed system. You also use the HMC to start and stop the logical partitions, update the server firmware code, manage IBM @server Capacity on Demand, and transmit service information to service and support if there are any hardware problems with your managed system.

Figure 19-1 illustrates the logical partitions and the server firmware on the IBM @server hardware. The server firmware is code stored in system flash memory on the server. The server firmware directly controls the resource allocations on the server and the communications between logical partitions on the server. The HMC connects with the server firmware and specifies how the server firmware allocates resources on the server.

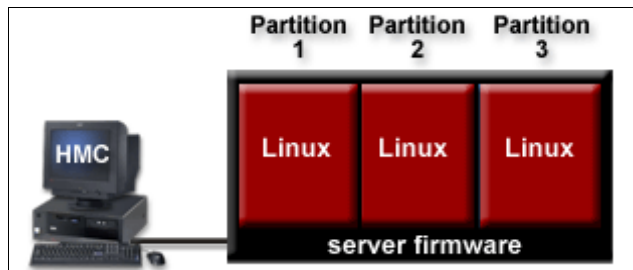


Figure 19-1 HMC illustration

If you use a single HMC to manage a server, and the HMC malfunctions or becomes disconnected from the server firmware, the server continues to run, but you will not be able to change the logical partition configuration of the server or manage the server. If desired, you can attach an additional HMC to act as a backup and to provide a redundant path between the server and IBM service and support.

Partitioning using the HMC is supported on all IBM @server i5, IBM @server p5, IBM System p5™, and IBM @server OpenPower server models, although some models require you to enter an Advanced POWER Virtualization activation code before you can partition the server.

- ▶ **Integrated Virtualization Manager**

The Integrated Virtualization Manager is a browser-based system management interface for Virtual I/O Server that enables you to create and

manage Linux logical partitions on a single IBM @server p5 or IBM System p5 server or Linux logical partitions on a single IBM @server OpenPower server. The Integrated Virtualization Manager is supported only on select IBM @server p5, IBM System p5, and OpenPower server models.

Virtual I/O Server is software that provides virtual storage and shared Ethernet resources to the other logical partitions on the managed system. Virtual I/O Server is not a general purpose operating system that can run applications. Virtual I/O Server is installed on a logical partition in the place of a general purpose operating system and is used solely to provide virtual I/O resources to other logical partitions with general purpose operating systems. You use the Integrated Virtualization Manager to specify to Virtual I/O Server how these resources are assigned to the other logical partitions.

To use the Integrated Virtualization Manager, you must first install Virtual I/O Server on an unpartitioned server. Virtual I/O Server automatically creates a logical partition for itself, which is called the management partition for the managed system. The management partition is the Virtual I/O Server logical partition that controls all of the physical I/O resources on the managed system. After you install Virtual I/O Server, you can configure a physical Ethernet adapter on the server so that you can connect to the Integrated Virtualization Manager from a computer with a Web browser.

Figure 19-2 on page 251 illustrates Virtual I/O Server in its own logical partition, and the Linux logical partitions that are managed by the Virtual I/O Server logical partition. The browser on the PC connects to the Integrated Virtualization Manager interface over a network, and you can use the Integrated Virtualization Manager to create and manage the logical partitions on the server.

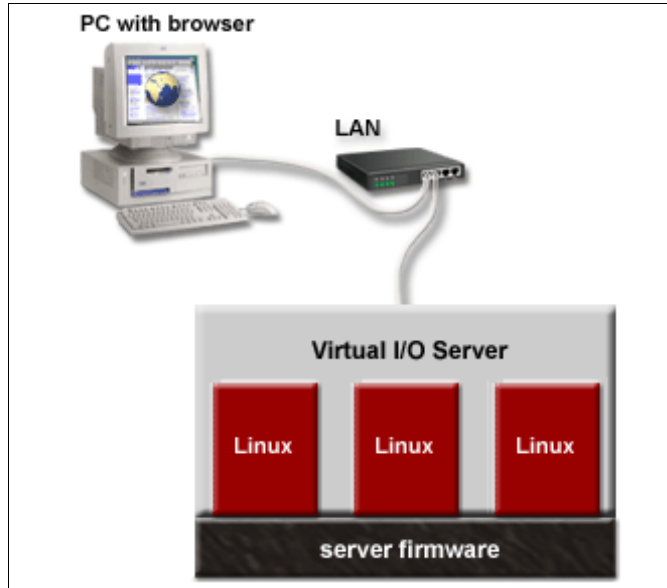


Figure 19-2 Virtual I/O server illustration

► Virtual Partition Manager

The Virtual Partition Manager is a feature of IBM i5/OS® V5R3 that enables you to create and manage one i5/OS logical partition and up to four Linux logical partitions on a single IBM @server i5 server.

To use the Virtual Partition Manager, you must first install i5/OS on an unpartitioned server. After you install i5/OS, you can initiate a console session on i5/OS and use Dedicated Service Tools (DST) or System Service Tools (SST) to create and configure Linux logical partitions. i5/OS controls the resource allocations of the logical partitions on the server.

Figure 19-3 on page 252 illustrates the i5/OS logical partition and the Linux logical partitions that are managed by the i5/OS logical partition. The twinaxial console connects to the i5/OS logical partition, and you can use DST or SST to create and configure the Linux logical partitions on the server.

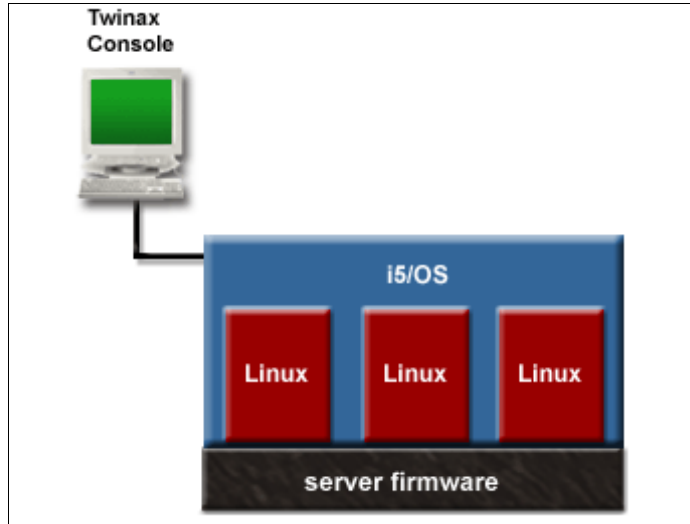


Figure 19-3 Virtual partition manager illustration

When you use the Virtual Partition Manager to partition an @server i5 server, DST and SST are the only tools that you can use to create and manage the logical partitions. You cannot use iSeries™ Navigator to create or manage logical partitions on an @server i5 server. However, the console session that you use to access DST or SST can be initiated using either iSeries Operations Console (LAN or direct attach) or a twinaxial console.

Although each logical partition acts as an independent server, the logical partitions on a server can share some kinds of resources with each other. The ability to share resources among many logical partitions enables you to increase resource utilization on the server and to shift the server resources to where they are needed. The following list illustrates some of the ways in which logical partitions can share resources. (For some server models, the features mentioned in this list are options for which you must obtain and enter an activation code.)

- ▶ Micro-Partitioning™ (or shared processing) enables logical partitions to share the processors in the shared processor pool. The shared processor pool includes all processors on the server that are not dedicated to specific logical partitions. Each logical partition that uses the shared processor pool is assigned a specific amount of processor power from the shared processor pool. If the logical partition needs more processor power than its assigned amount, the logical partition is set by default to use the unused processor power in the shared processor pool. The amount of processor power that the logical partition can use is limited only by the virtual processor settings of the logical partition and the amount of unused processor power available in the shared processor pool.

- ▶ Dynamic logical partitioning enables you to move resources to, from, and between running logical partitions manually without shutting down or restarting the logical partitions. This enables you to share devices that logical partitions use occasionally. For example, if the logical partitions on your server use an optical drive occasionally, you can assign a single optical drive to multiple logical partitions as a desired device. The optical drive would belong to only one logical partition at a time, but you can use dynamic logical partitioning to move the optical drive between logical partitions as needed. On servers that are managed using the Integrated Virtualization Manager, dynamic logical partitioning is supported only for the management partition. Dynamic logical partitioning is not supported on servers that are managed using the Virtual Partition Manager.
- ▶ Virtual I/O enables logical partitions to access and use I/O resources on other logical partitions. For example, virtual Ethernet enables you to create a virtual LAN that connects the logical partitions on your server to each other. If one of the logical partitions on the server has a physical Ethernet adapter that is connected to an external network, you can configure the operating system of that logical partition to connect the virtual LAN with the physical Ethernet adapter. This enables the logical partitions on the server to share a physical Ethernet connection to an external network.

Hardware requirements for Linux logical partitions

During the process of planning for logical partitions, you must decide how you want to configure hardware resources. Each Linux logical partition on an IBM @server hardware system requires the following minimum hardware resources:

- ▶ One dedicated processor or 0.1 processing unit
- ▶ 128 MB memory
- ▶ One storage adapter (physical or virtual)
- ▶ Approximately 1GB of space provided by one of the following storage options:
 - Internal storage using SCSI adapters and drives attached within the system
 - External storage (SAN) using SAN adapters and drives in an external storage unit
 - Virtual storage using a virtual SCSI adapter and storage in a different partition
- ▶ One network adapter (physical or virtual)

Logical partition planning tasks

Use the following planning tasks:

1. Identify the hardware requirements for each logical partition based on the hardware configuration of the server.
2. Identify whether the partitions will communicate with other partitions, servers, or workstations using physical or virtual Ethernet connections.
3. Design and validate your partition configuration using the IBM LPAR Validation Tool (LVT), available at:

<http://www.ibm.com/servers/eserver/iseries/lpar/>

The LVT provides you with a validation report that reflects your system requirements while not exceeding logical partition recommendations.

4. If your machine is an @server i5, decide if you want i5/OS to provide Linux logical partitions.
5. Plan for Linux software licensing in a partitioned environment by reading and understanding the license agreement for your Linux distribution.

19.1.2 IBM eServer BladeCenter JS20

This section discusses planning considerations associated with the implementation of IBM @server BladeCenter JS20. Specifically, it covers network planning.

Network planning

Successful installation of BladeCenter JS20 requires that you have a clear plan for how you will use the various networking capabilities of the BladeCenter infrastructure. This plan should address the following questions:

- ▶ What network connectivity is needed for the blade servers to support the applications installed on them?
- ▶ What network connectivity is needed to manage the BladeCenter, input/output (I/O) modules, and blade servers?
- ▶ What virtual local area networks (VLANs) are required for each LAN Switch I/O Module to provide the network connectivity established previously?
- ▶ What IP subnet will be used for each VLAN and how will IP addresses be allocated to each device on the VLAN?
- ▶ Will IP addresses be assigned statically or dynamically using DHCP?
- ▶ What host names will be used for each network interface?
- ▶ How will host names be resolved to IP addresses?

- ▶ Are there requirements for a high-performance, low-latency interconnection network?
- ▶ Where multiple BladeCenter chassis are installed, how will they be interconnected?

The following topics explore common network planning requirements that we expect to arise in the installation of BladeCenter JS20s.

Minimal network requirements

At a minimum, most BladeCenter JS20 environments have the following network requirements:

- ▶ A dedicated hardware management subnet: It is used to access both the Management Module and management interfaces of I/O modules that are installed in each BladeCenter chassis.
- ▶ A Serial over LAN (SoL) subnet internal to each BladeCenter chassis that supports the SoL remote text console function: This is always implemented using a LAN Switch I/O Module installed in I/O module bay 1.
- ▶ A subnet connected to each BladeCenter JS20: It is used to install and manage the operating system on the blade server.
- ▶ One or more subnets connected to each BladeCenter JS20: They are used by applications installed on the blade server to communicate to other systems.

Figure 19-4 on page 256 illustrates how these requirements can be provided in a logical network view.

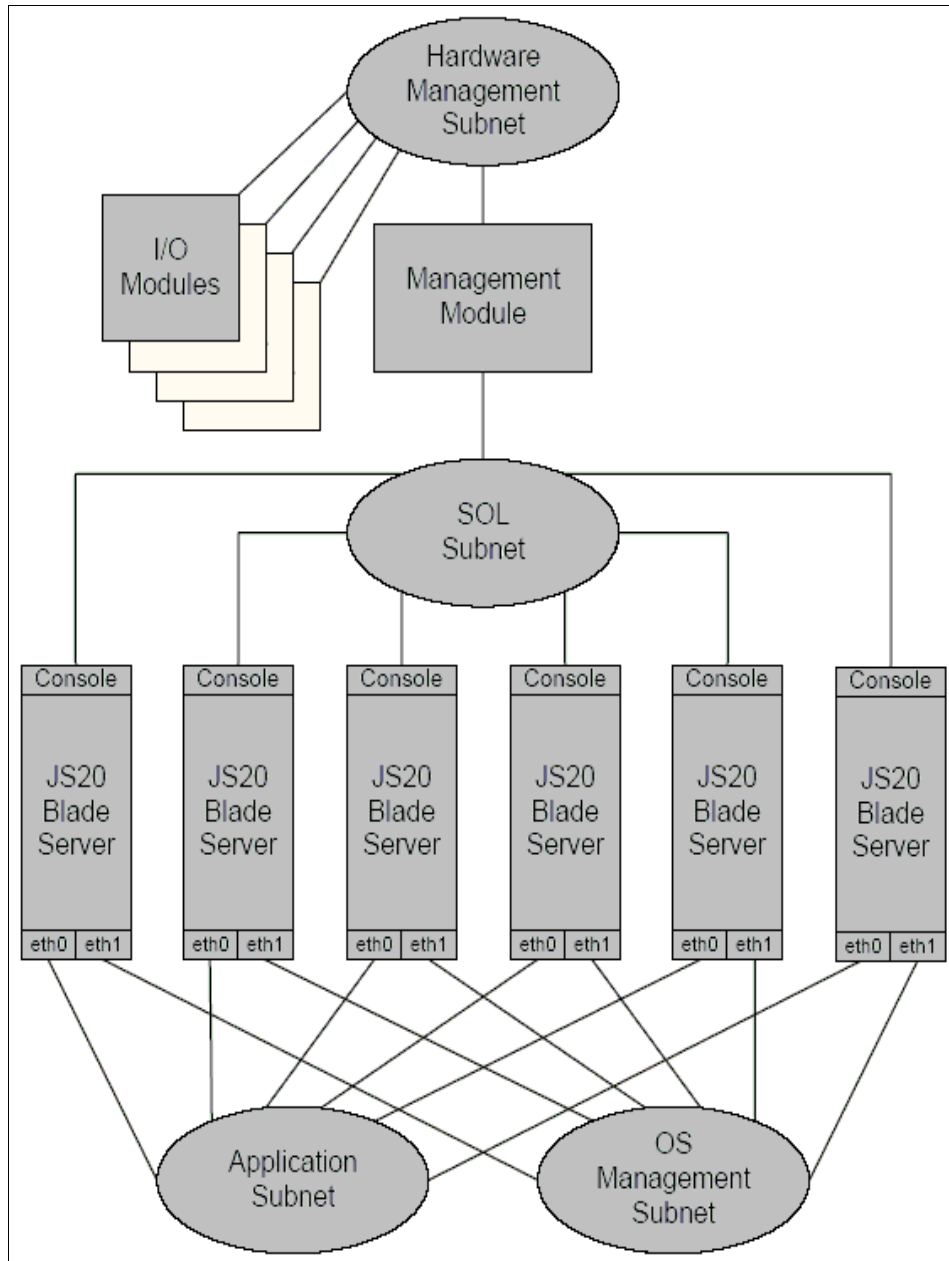


Figure 19-4 Network logical view

The following four sections describe each of the logical networks illustrated in Figure 19-4.

Hardware management subnet

We recommend that you establish a dedicated hardware management subnet. The 10/100BaseT Ethernet interface on the Management Modules installed in each BladeCenter chassis provides the gateway to this subnet for each BladeCenter chassis. You need to install external LAN switches or hubs to interconnect the 10/100BaseT Ethernet interface on the Management Modules of each BladeCenter chassis with external management systems.

You use this subnet to access the Management Module Web interface and command line interface (CLI). You also use this subnet to access the Web interface and CLI of I/O modules.

System management applications such as IBM Director or IBM Cluster Systems Management also use this subnet to communicate with the hardware management functions of the BladeCenter infrastructure.

Restrict access to this subnet to those management systems, system administrators, and operators who have responsibility for managing the BladeCenter infrastructure.

You need to allocate multiple IP addresses to each BladeCenter chassis on this subnet, including:

- ▶ One IP address for the external interface of the Management Module in each BladeCenter chassis
- ▶ One IP address for the internal interface of the Management Module in each BladeCenter chassis
- ▶ One IP address for the management interface of each I/O module in each BladeCenter chassis

Note: Although the logical network view illustrated in Figure 19-4 shows the I/O Management Module interfaces connecting directly to the hardware management subnet, they are physically connected through the Management Module, which acts as gateway to those interfaces. The Management Module performs a proxy Address Resolution Protocol (ARP) function to make it appear as though the I/O module management interfaces are attached to the hardware management subnet.

It is possible to use a different subnet for the Management Module internal network interface and I/O module management interfaces. However, we do not recommend this configuration.

Serial over LAN subnet

The SoL remote text console function requires a subnet and underlying VLAN that is implemented by a LAN Switch I/O Module installed in I/O module bay 1 of the BladeCenter chassis. This subnet and VLAN is entirely internal to each BladeCenter chassis and should not be externally accessible.

If you use the 4-Port Gigabit Ethernet Switch Module or the Nortel Networks Layer 2-7 Gigabit Ethernet Switch Module, the VLAN uses VLAN ID 4095. If you use the Cisco Systems Intelligent Gigabit Ethernet Switch Module, you can choose the VLAN ID.

You need to assign a unique range of IP addresses to this subnet for use by the SoL remote text console function.

Important: One IP address is required for each blade server.

You need to specify only the starting IP address within the range of IP addresses that you assign into the Management Module. The Management Module then automatically assigns consecutive IP addresses from the starting address that you provide to each blade server that you have installed.

Operating system management subnet

We expect most environments that use the BladeCenter JS20 to rely on the network installation procedure to install the operating systems.

The operating system management subnet is used to support both the initial installation and subsequent management of the operating systems installed on BladeCenter JS20s. This subnet is implemented using a VLAN provided by the LAN Switch I/O Modules installed in I/O module bay 2 of each BladeCenter chassis. Alternatively, you can implement it using a Pass-Thru I/O Module installed in I/O module bay 2 that is connected to an external LAN switch.

Application subnet

The primary reason you install BladeCenter JS20s is to support applications. Many applications have requirements to communicate with other systems. You use one or more application subnets for this purpose.

The primary application subnet is implemented using a VLAN provided by the LAN Switch I/O Modules installed in I/O module bay 1 of each BladeCenter chassis. The same LAN Switch I/O Module is used to support the SoL subnet and VLAN.

Where different BladeCenter JS20s participate in different applications and there is a requirement to separate application traffic, you might need to define multiple

application subnets and VLANs. Each BladeCenter JS20 is connected to the appropriate application subnet based on the applications that are installed on the blade server.

If application communication requirements with other systems are complex, you can install an additional pair of Gigabit Ethernet interfaces on each BladeCenter JS20. You do this using the Gigabit Ethernet Expansion Card in conjunction with compatible I/O modules installed in I/O module bays 3 and 4.

19.2 Booting and system initialization

This section describes the booting and system initialization process of @server i5, @server p5, and JS20 blade. For simplicity, we only cover booting and installing from the CD in this section. For additional methods, refer to your hardware and Linux distribution installation guides.

19.2.1 IBM eServer i5 and eServer p5

The initial step of powering on the server depends on whether the system is monolithic or hosted.

Monolithic (stand-alone)

There are three ways to power on a monolithic @server p5:

- ▶ Power switch
- ▶ The Advanced System Management Interface (ASMI) through a serial connection
- ▶ The ASMI Web interface

Only the first and second option are available when the machine is initially unpacked from the box and plugged in. The third option requires the configuration of the Hardware Management Console (HMC) network port using the ASMI serial connection. For more information about the ASMI, refer to *Managing the Advanced System Management Interface (ASMI)*, available at:

<http://publib.boulder.ibm.com/infocenter/eserver/v1r3s/index.jsp>

Tip: Use ASMI to set the system's time and date *before* beginning any Linux installations. This will help avoid potential problems during Linux installations.

After powering on, a monolithic system proceeds to progress through a series of hardware checks. The LED panel on the front of the machine will flash various codes to indicate this progression.

If the machine is booted with an ASCII terminal connected to its serial port, these same LED codes will display on the ASCII terminal. Nothing will show on a console connected to the graphics card during this process.

Hosted (partitioned)

The server's HMC network port must be configured using the ASMI serial connection. For more information, refer to *Managing the Advanced System Management Interface (ASMI)*, available at:

<http://publib.boulder.ibm.com/infocenter/eserver/v1r3s/index.jsp>

Tip: Use ASMI to set the system's time and date *before* beginning any Linux installations. This will help avoid potential problems during Linux installations.

The Hardware Management Console must be configured and the server attached before powering up. Refer to *Managing the Hardware Management Console (HMC)*, available at:

<http://publib.boulder.ibm.com/infocenter/eserver/v1r3s/index.jsp>

When connected to an HMC, there are three ways to power on @server i5 and @server p5 systems:

- ▶ The HMC
- ▶ The Advanced System Management Interface (ASMI) through a serial connection
- ▶ The ASMI Web interface

Restriction: When connected to an HMC, @server i5 and @server p5 will not power up through the power switch.

We recommend using the HMC to power on a hosted system into the partition standby mode. This mode powers the machine to a level that allows partition creation and manipulation. To power on a managed system to this mode, complete the following steps:

1. In the Navigation area, expand the **Server and Partition** folder.
2. Click the **Server Management** icon.
3. In the Contents area, select the managed system.
4. From the menu, click **Selected** → **Power On**.
5. Select **Standby** power-on mode and click **OK**.

After powering on, a hosted system proceeds to progress through a series of hardware checks. The LED panel on the front of the machine and the panel

display for the machine in the HMC window will flash various codes to indicate this progression. The system status panel displays “Standby” after the power on process has completed.

System management services (SMS) menu

This section describes the basic steps involved in booting a monolithic system or partition to the point where the system management services menu becomes available. The SMS menu enables the system administrator to configure, among other things, the boot options for the system or partition. There are two ways to boot into the SMS menu:

- ▶ Preboot: This involves using the ASMI power on options.
- ▶ During boot: This involves using the console or HMC terminal.

We only cover the second method in this section. For information about the first method, refer to *Managing the Advanced System Management Interface (ASMI)*, available at:

<http://publib.boulder.ibm.com/infocenter/eserver/v1r3s/index.jsp>

Monolithic (stand-alone)

During the power-on process, the console (ASCII or graphical) will eventually display information about the final stage of hardware checking. This stage is where the SMS menu becomes available. To access the SMS menu, the usual method involves pressing the F1 key after the keyboard icon appears for a graphical console (see Figure 19-5 on page 262), or pressing the 1 key after the word “keyboard” appears for a ASCII terminal (see Figure 19-7 on page 264). Consult your hardware guide to confirm the correct graphical key sequence. The menu opens after the hardware check process completes.

Note: If prompted for a password, the default password is “admin”.



Figure 19-5 Graphical SMS example

Hosted (partitioned)

When the system is in standby mode, the next step involves creating and booting a partition. For steps for creating partitions, as well as example partition configurations, refer to *Partitioning for Linux*, available in the IBM @server Information Center:

<http://publib.boulder.ibm.com/infocenter/eserver/v1r3s/index.jsp>

In hosted mode, the partition to install must have the CD/DVD drive in its partition profile. To assign the CD/DVD drive to a partition:

1. Go to **Server and Partition** → **Server Management**.
2. Open the server and partition to which you want to install.
3. Right-click the profile you want to use for installation and select **Properties**.
4. In the “Logical Partiton Profile Properties” window, go to the Physical I/O tab.
5. From the “Managed system I/O devices” window, select the **Other Mass Storage Controller** from the bus where it is attached.
6. Click **Add as required** to assign this CD/DVD drive to the partition.

The result should look similar to the example in Figure 19-6 on page 263.

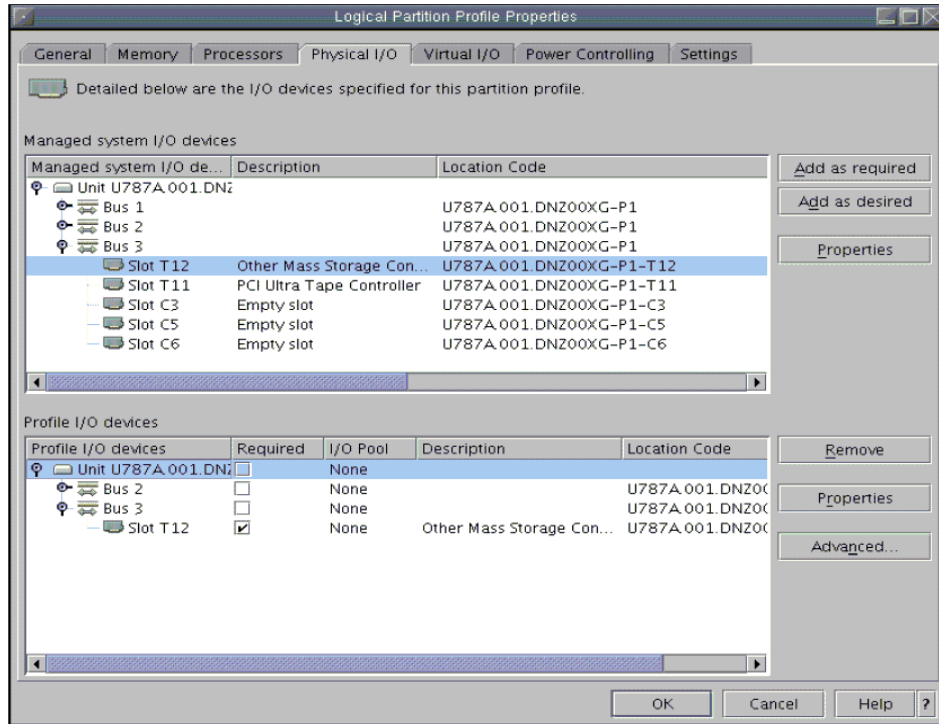


Figure 19-6 Adding a CD device example

To boot an @server p5 partition:

1. Right-click the partition.
2. Select **Activate**.
3. Select the **Open terminal** option.
4. Click **OK**.

To boot an @server i5 partition:

1. Right-click the partition.
2. Select **Open Terminal Window**.
3. Vary on the partition from i5/OS.

During the power on process, the terminal console eventually displays information about the final stage of hardware checking. This stage is where the SMS menu becomes available. To access the SMS menu, press the 1 key after the word “keyboard” appears (see Figure 19-7 on page 264). The menu appears after the hardware check process completes.

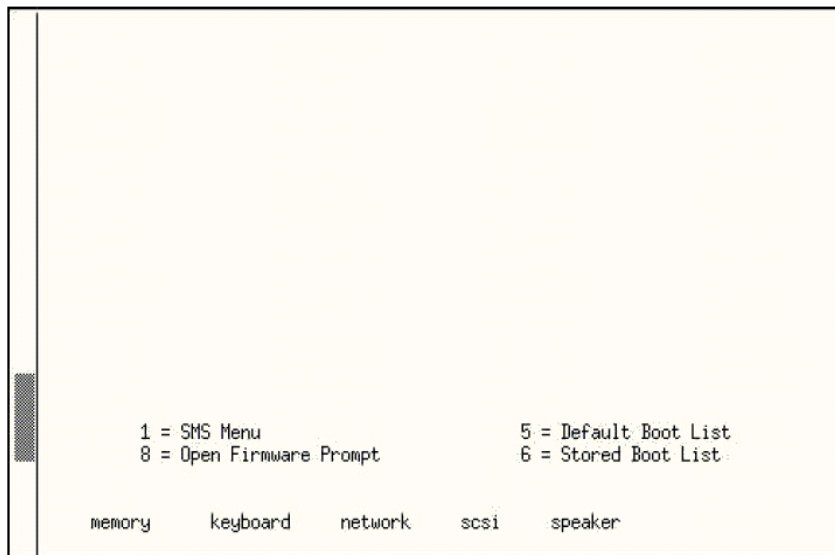


Figure 19-7 Text SMS example

Boot device selection

When in the SMS menu, select the **Select Boot Options** option. In that menu, specify **Select Install or Boot a Device**. There, select **CD/DVD**, and then the bus type (IDE or SCSI). If you are uncertain, you can select to view all devices. This scans all available buses for boot devices, including network adapters and hard drives (will not display empty or non-bootable drives). Finally, select the device containing the installation CD.

19.2.2 eServer BladeCenter JS20

We cover the following topics in this section:

- ▶ Management Module configuration
- ▶ LAN Switch I/O Module configuration
- ▶ Blade server configuration
- ▶ Providing a console for the JS20 blades

Management Module configuration

This section describes the steps to set up the BladeCenter Management Module so that you can work with BladeCenter JS20s. Advanced configuration of the Management Module is beyond the scope of this section.

The primary setup task for the Management Module is to assign IP addresses, which are necessary to communicate with the Management Module Web and command line interfaces.

The Management Module has two network interfaces, an external interface (eth0) and an internal interface (eth1). The external interface is accessible through the 10/100BaseT connector on the Management Module. The internal interface is connected to the management interfaces of all the installed I/O modules that support such interfaces. This includes all switch I/O modules.

The default behavior of a new Management Module is to request an IP address for the external interface through DHCP. If the Management Module does not receive a valid response from a DHCP server within two minutes of powering on, it uses a default static IP address of 192.168.70.125 with the default subnet mask of 255.255.255.0.

Note: The default host name is MMxxxxxxxxxxxx, where xxxxxxxxxxxx is the burned-in medium access control (MAC) address.

The default IP address for the internal interface is statically assigned to be 192.168.70.126 with a subnet mask of 255.255.255.0.

You can reset the IP addresses of a Management Module that was previously configured back to the factory defaults by using the IP reset button on the Management Module. For the procedure for doing this, see *IBM @server BladeCenter Management Module User's Guide*, available at:

<http://www.ibm.com/pc/support/site.wss/MIGR-45153.html>

Configuring the Management Module

Use the following procedure to set the IP addresses for the Management Module external and internal interfaces:

1. Connect the Management Module to an isolated private Ethernet network. Also connect a workstation that has a Web browser to the same isolated private Ethernet network.
2. Configure a static IP address for the workstation Ethernet interface that is in the same subnet as the Management Module default IP addresses. For example, we used a static address of 192.168.70.100 with a subnet mask of 255.255.255.0 for the workstation Ethernet interface.

Restriction: Do not use addresses in the range of 192.168.70.125 through 192.168.70.130. These addresses conflict with the default addresses assigned by the Management Module.

3. Connect to the Management Module Web interface by pointing your Web browser on the workstation to:
`http://192.168.70.125`
4. Enter a valid user ID and password. The factory default configuration of a Management Module defines a user ID named, USERID, with a password of PASSWORD. The 0 in the password is a zero. In production environments, consider changing these defaults.
5. From the Management Module Web interface, select **MM Control** → **Network Interfaces**.
6. The form window shown in Figure 19-8 on page 267 opens. Enter the desired external and internal IP addresses, subnet masks, and default gateway for the Management Module. Then, click **Save** to store the new IP addresses.

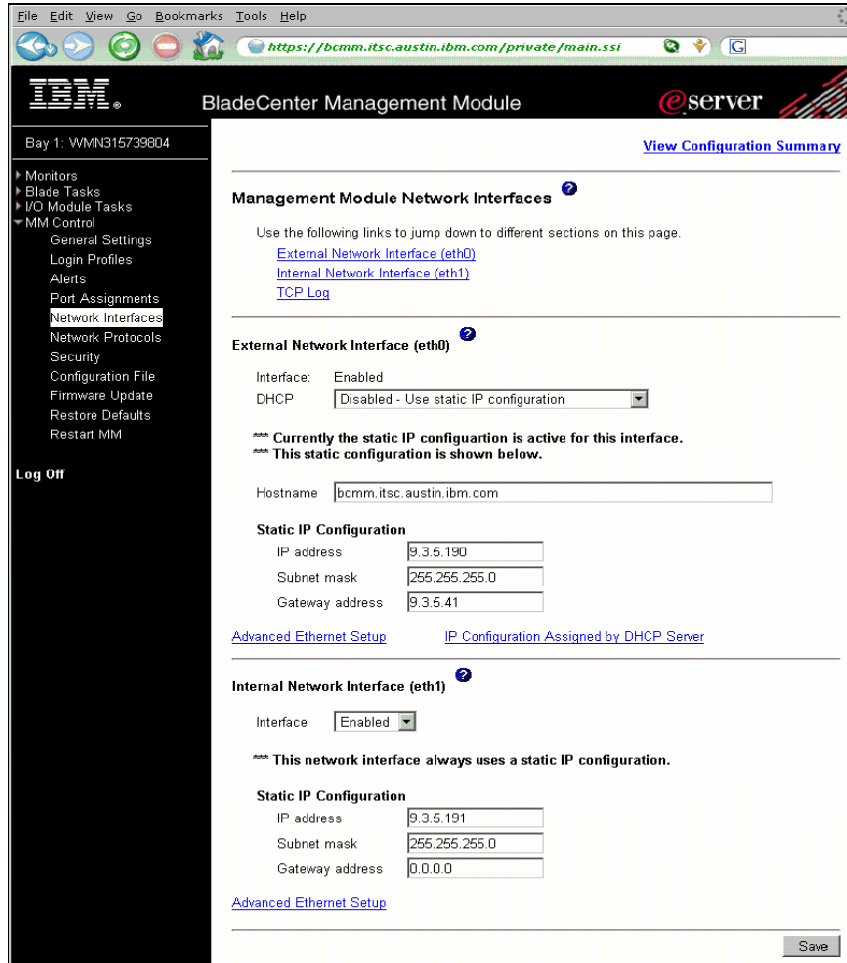


Figure 19-8 IP address configuration

7. Restart the Management Module. In the Management Module Web interface, select **MM** → **Control** → **Restart MM**. Then, click the **Restart** button on the displayed form. You are prompted to confirm the restart before it occurs.
8. Remove the Management Module from the isolated private Ethernet network and connect it to the network you will use to manage BladeCenter.

You can now connect to the Management Module Web and command line interfaces using the IP address that you assigned to the Management Module external network interface.

Now, consider performing other Management Module setup tasks, such as:

- ▶ Setting the Management Module date and time so that log entries have useful time stamps.
- ▶ Defining user IDs and passwords for the system administrators and operators who will manage the BladeCenter.
- ▶ Alternatively, you can configure the Management Module to use a Lightweight Directory Access Protocol (LDAP) directory for this purpose.
- ▶ Configuring the Management Module to send alerts to management systems through SNMP, or system administrators using e-mail through Simple Mail Transfer Protocol (SMTP).
- ▶ Enabling the use of Secure Sockets Layer (SSL) to securely access the Management Module Web interface.
- ▶ Enabling the use of Secure Shell (SSH) to securely access the Management Module command line interface (CLI).

For additional information about how to perform these tasks, refer to the *IBM @server BladeCenter Management Module User's Guide*, available at:

<http://www.ibm.com/pc/support/site.wss/MIGR-45153.html>

LAN Switch I/O Module configuration

This section explains the basic set up for LAN Switch I/O Modules so that you can work with BladeCenter JS20s. The advanced configuration of LAN Switch I/O Modules is beyond the scope of this section.

The primary setup tasks for LAN Switch I/O Modules are to:

1. Assign IP addresses to the LAN Switch I/O Module management interfaces.
2. Enable the LAN Switch I/O Module external Ethernet ports.

When a new LAN Switch I/O Module is first installed, the Management Module assigns a default IP address to the management interface of the LAN Switch I/O Module. The default IP address is chosen based on the I/O module bay where the LAN Switch I/O Module is installed. Those I/O modules installed in I/O module bays 1, 2, 3, and 4 are assigned IP addresses 192.168.70.127, 192.168.70.128, 192.168.70.129, and 192.168.70.130, respectively.

Set the IP address of each LAN Switch I/O Module management interface and enable the external ports:

1. Connect to the Management Module using the Web browser interface as explained in "Configuring the Management Module" on page 265.

2. From the Management Module Web interface, select **I/O Module Tasks** → **Management**.
3. In the form shown in Figure 19-9, scroll down to the entry for the LAN Switch I/O Module that you want to configure. Enter the IP address, subnet mask, and gateway that you want to assign to the LAN Switch I/O Module management interface. Click **Save** to activate the new IP address.

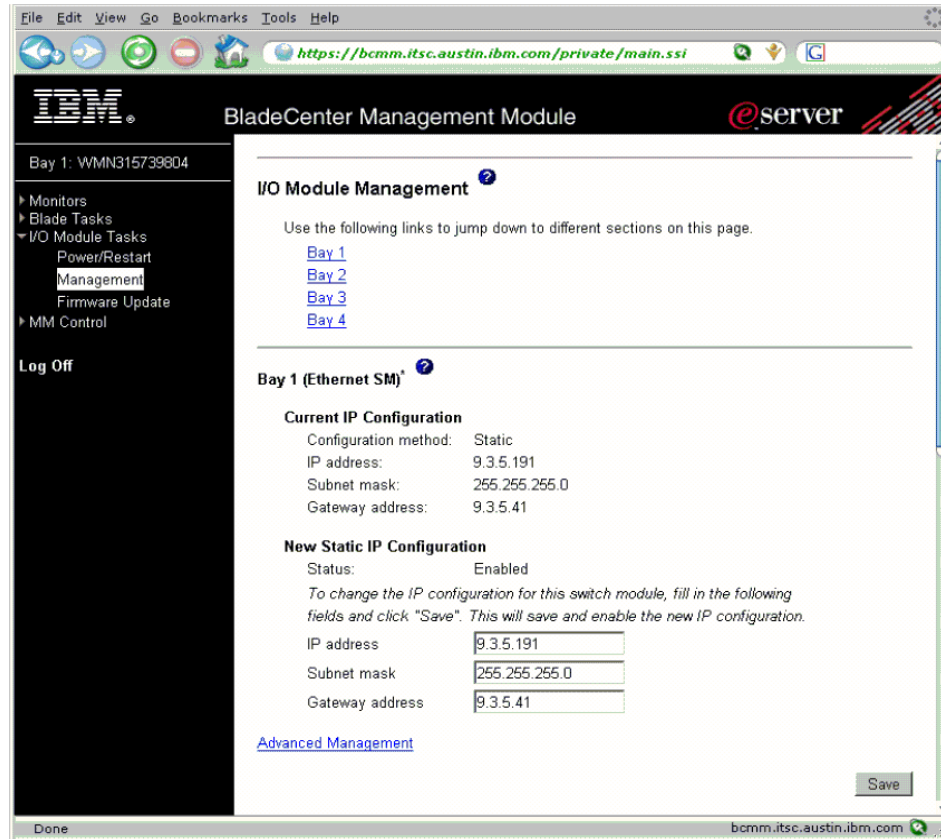


Figure 19-9 LAN Switch I/O Module IP address

4. You are prompted to confirm that you want to change the IP address.
5. Select the **Advanced Management** link for the LAN Switch I/O Module.
6. Scroll down to the Advanced Setup section of the displayed form, as illustrated in Figure 19-10 on page 270. For the External ports field, select **Enabled** from the drop-down list and then click **Save**.

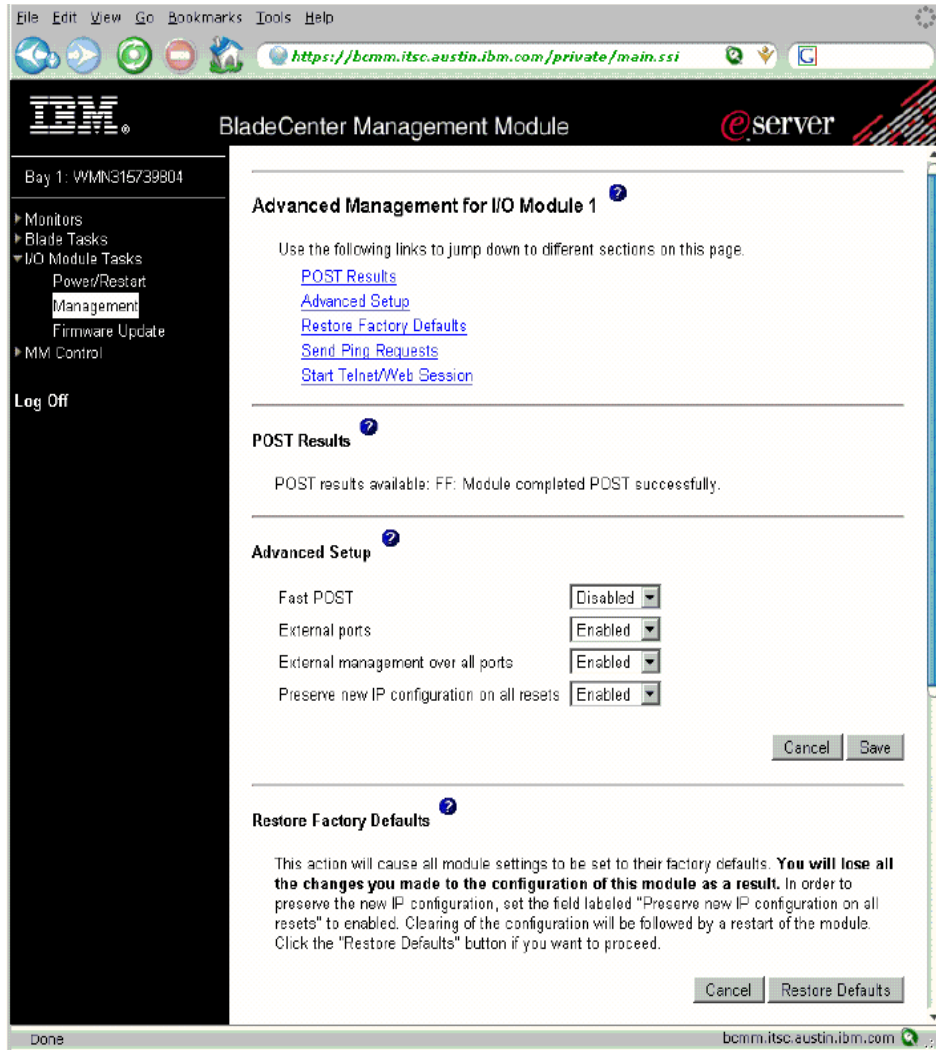


Figure 19-10 LAN Switch I/O Module setup

At this point, the LAN Switch I/O Module management interface has an IP address. In addition, the external ports on the LAN switch module are enabled so that they can be used to communicate with blade servers.

The SoL remote text console function of the Management Module depends on a VLAN provided by a LAN Switch I/O Module installed in I/O module bay 1. This VLAN is automatically provided by the 4-Port Gigabit Ethernet Switch Module and the Nortel Networks Layer 2-7 Gigabit Ethernet Switch Module using VLAN 4095.

If you are using a Cisco Systems Intelligent Gigabit Ethernet Switch Module in I/O module bay 1, you must manually configure this VLAN. A procedure is documented in the *Cisco Systems Intelligent Gigabit Ethernet Switch Module for the IBM @server BladeCenter Installation Guide*. This Cisco guide describes how to manually set up the VLAN that is needed to support the SoL remote text console function and is available at:

<http://www.ibm.com/pc/support/site.wss/document.do?lnocid=MIGR-57858>

Blade server configuration

Minimal configuration is required for each BladeCenter JS20 prior to installing an operating system. The two main configuration tasks are:

1. Assigning a name to the blade server
2. Setting the blade server boot sequence

The easiest way to perform these tasks is through the Management Module Web interface.

Assigning names to blade servers

Set the name of each blade server in a BladeCenter chassis by using these steps:

1. From the Management Module Web interface, select **Blade Tasks** → **Configuration**.
2. In the Blade Information section of the form, as shown in Figure 19-11 on page 272, type the name that you want to assign to each blade server.
3. Scroll down the form and select **Save** to save the names that you have assigned to each blade server.

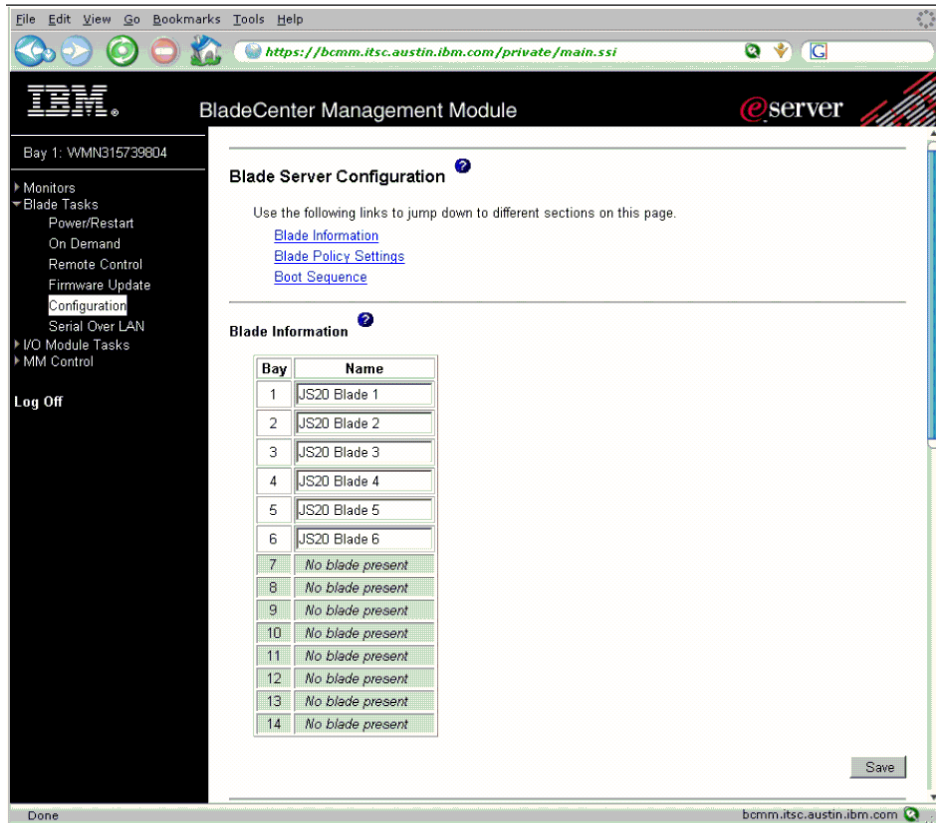


Figure 19-11 Blade Server naming

Setting the boot sequence

Set the boot sequence of each blade server by using this procedure:

1. From the Management Module Web interface, select **Blade Tasks** → **Configuration**.
2. Scroll down the form to the Boot Sequence section to see the current boot sequence for all the blade servers.
3. If a blade server does not have the correct boot sequence, you can change the boot sequence by selecting the blade server name, which opens the form shown in Figure 19-12 on page 273.
4. Set the desired boot sequence using the drop-down lists and click **Save** to set the new boot sequence.

The correct boot sequence depends on the method that you plan to use to install an operating system on the blade server. To install Linux from a CD, set the CD-ROM first.

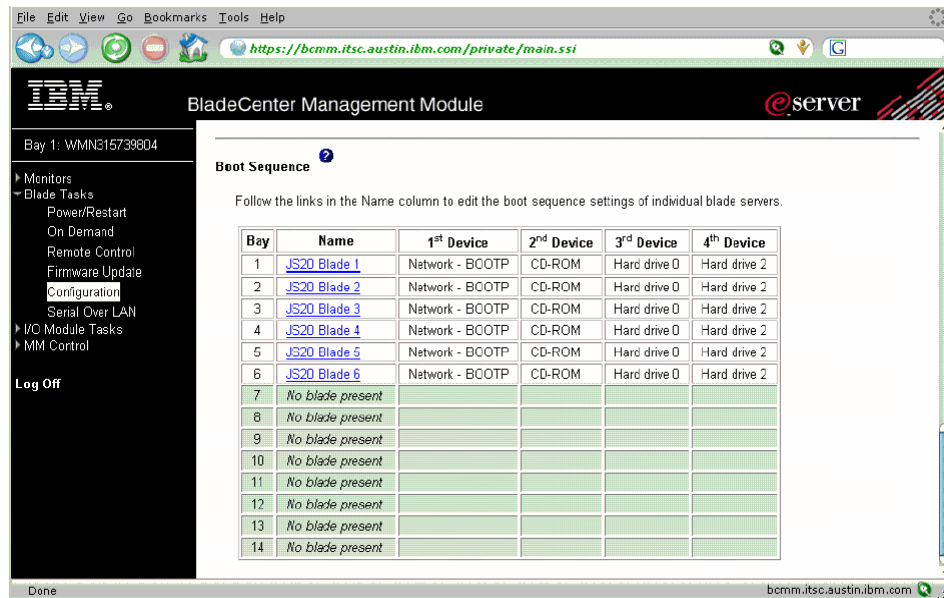


Figure 19-12 Boot Sequence of blades

Providing a console for the JS20 blades

The BladeCenter JS20 differs from other blade servers that are available for the BladeCenter, in that it does not provide an interface to a keyboard, video monitor, and mouse (KVM) console. Therefore, you must set up the Serial over LAN (SoL) remote text console function to provide a console for a BladeCenter JS20. Use of SoL is optional for other types of blade servers that support KVM consoles.

The SoL remote text console function involves several different components in the BladeCenter infrastructure and works as follows:

1. Using a Telnet or SSH client, connect to the BladeCenter Management Module CLI. This is usually through an external management network that is connected to the Management Module's 10/100BaseT Ethernet interface.
2. From the Management Module CLI, initiate a SoL remote console session to the desired blade server.
3. The Management Module uses a private VLAN provided by the LAN switch module in I/O module bay 1 to transport the SoL data stream to the Ethernet interface of the target blade server.

4. The Ethernet controller of the target blade server passes the SoL data stream received from the private VLAN to the blade system management processor (BSMP), which manages the text console for the blade server.

Configuring Serial over LAN

Before you attempt to configure the SoL remote text console function, verify that you have all the prerequisites in place:

- ▶ A supported LAN switch module installed in I/O module bay 1
This switch module is used to provide a VLAN that connects the Management Module to the first Ethernet interface on each blade server.
- ▶ The correct firmware levels
This is important if you install a BladeCenter JS20 in an existing BladeCenter chassis that might have back-level firmware in the Management Module or LAN switch modules.
- ▶ A reliable Telnet or SSH client
- ▶ A network connecting the Telnet or SSH client to the BladeCenter Management Module external 10/100BaseT Ethernet interface
- ▶ An identified range of IP addresses that will be used by the Management Module to communicate with the BSMP on each blade server through the private VLAN

The easiest way to configure the SoL remote text console function is through the Management Module Web interface as explained in the following steps.

To configure the SoL remote text console function, use the following procedure:

1. From the Management Module Web interface, select **Blade Tasks** → **Serial Over LAN**.
2. In the right pane, scroll down until you see the Serial Over LAN Configuration section (see Figure 19-13 on page 275). Complete the following tasks:
 - a. From the Serial over LAN list, select the **Enabled** option.
 - b. Leave the value for SoL VLAN ID at the default (4095) if you have either a 4-port Gigabit Ethernet Switch Module or a Nortel Networks Layer 2-7 Gigabit Ethernet Switch Module installed in I/O module bay 1.
If you have a Cisco Systems Intelligent Gigabit Ethernet Switch Module installed in I/O module bay 1, set the VLAN ID to the same value you used when you manually defined the VLAN that supports SoL.
 - c. Enter the start of the IP address range that will be used by the Management Module to communicate with the BSMP on each blade server.

- d. Leave the values for Accumulate timeout, Send threshold, Retry count, and Retry interval at their defaults (5, 250, 3, and 250).
- e. Leave the values for User Defined Keystroke Sequences at their defaults.
- f. Click **Save**.

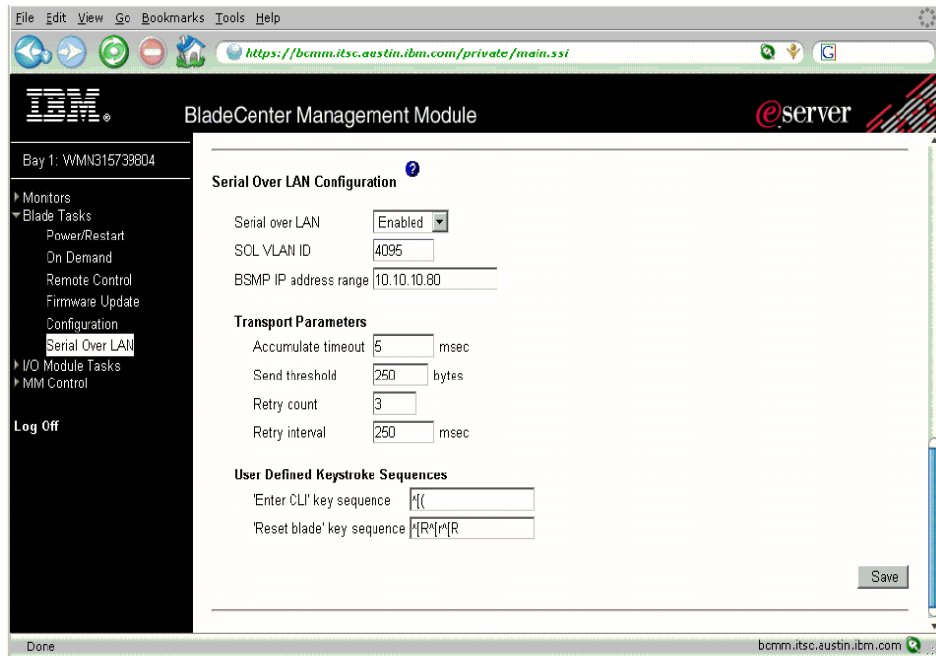


Figure 19-13 Serial Over LAN Configuration

3. Restart the Management Module:
 - a. In the Management Module Web interface, select **MM Control** → **Restart MM**.
 - b. Click the **Restart** button on the displayed form.
 - c. You are prompted to confirm the restart before it occurs.
4. After the Management Module has restarted and you have reconnected to the Management Module Web interface from your Web browser, enable the SoL remote text console for each blade server.
 - a. Select **Blade Tasks** → **Serial Over LAN** from the Management Module Web interface.
 - b. In the right pane, scroll down until you see the Serial Over LAN Status section (see Figure 19-14 on page 276).

- c. Select the blade servers you want to enable for SoL. You can choose all of them by selecting the check box at the top of the table.
- d. Click the **Enable Serial Over LAN** link under the table. You might need to scroll down to see this link.

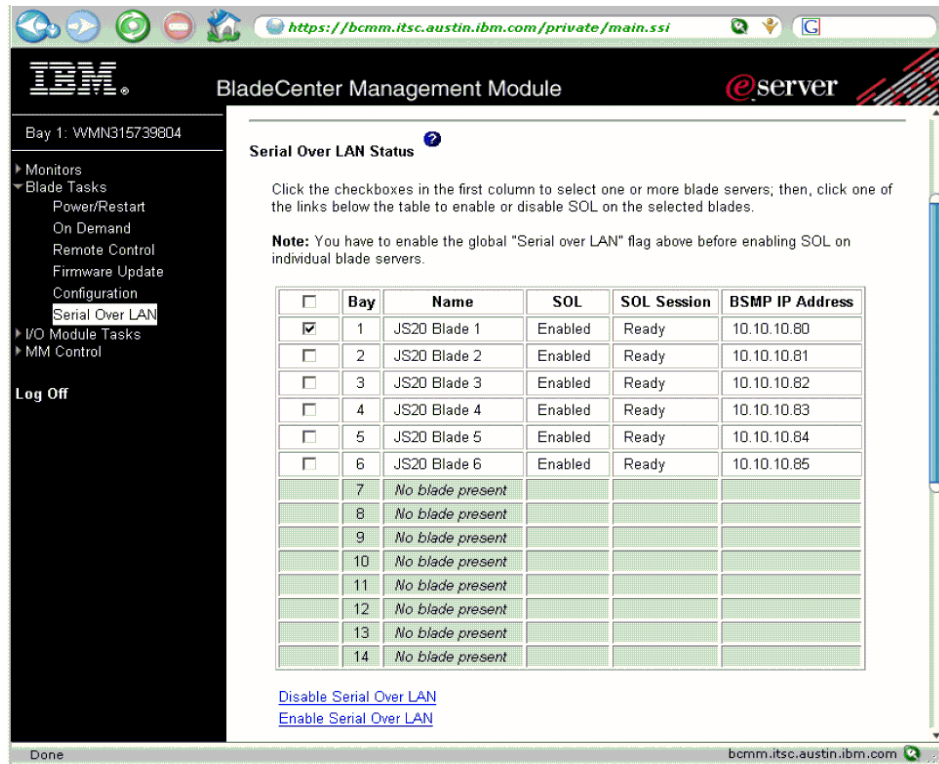


Figure 19-14 Serial Over LAN Status

- e. After a few seconds, the window refreshes. In the SOL column of the table, verify that each blade server has a status of Enabled.

The configuration of the SoL remote text console function is now complete.

Using Serial over LAN

Access the SoL remote text console function through the Management Module CLI. The CLI is documented in the *IBM @server BladeCenter Management Module Command-Line Interface Reference Guide*, available at:

<http://www.ibm.com/support/docview.wss?uid=psg1MIGR-54667>

You can connect to the Management Module CLI using either a Telnet or SSH client. The Management Module can support up to 20 concurrent CLI

connections. This is sufficient to support the concurrent use of a SoL remote text console to each blade server in a full BladeCenter chassis. At the same time, it supports six additional CLI connections for other administrative activities.

Restriction: You can only have one active SoL remote text console connection to each blade server.

The Management Module CLI is context sensitive. When you issue CLI commands, you can accept the current context or override the context using the `-T` option provided on most commands. In our examples, we use the `-T` option to make it clear on which entity the command is operating. You can change the context for the Management Module CLI by using the `env -T` command.

To use the SoL remote text console function, first connect to the Management Module from a Telnet or SSH client. You are prompted to enter a user ID and password. The default user ID is `USERID` and the default password is `PASSWORD`, where 0 in the default password is a zero. Consider changing the defaults in a production environment.

There are several different commands that you can use to access the SoL remote text console function. Table 19-1 lists the most useful commands.

Table 19-1 Management Module commands for SoL

Command	Description
<code>console</code>	Open a SoL remote text console for the blade server. This command fails if another SoL remote text console is already open for the blade server.
<code>console -o</code>	Terminate any existing SoL remote text console for the blade server and open a SoL remote text console for the blade server.
<code>boot -c</code>	Reset the blade server, and then open a SoL remote text console for the blade server.
<code>reset -c</code>	This is functionally equivalent to the <code>boot -c</code> command when used in a blade server context.
<code>power -on -c</code>	Power on the blade server and then open a SoL remote text console for the blade server.
<code>power -cycle -c</code>	Power on the blade server and then open a SoL remote text console for the blade server. If the blade server is already powered on, power it off first, and then power on.

Here are some examples of how to use these commands. To open a SoL remote text console to the blade server in bay 3, use the following command:

```
console -T system:blade[3]
```

To reset the blade server in bay 2 and then open a SoL remote text console to the blade server, use the following command:

```
boot -c -T system:blade[2]
```

To terminate an active SoL remote text console, press the Esc key followed by an open parenthesis (Shift+9 on U.S. keyboards). When the SoL remote text console ends, you return to the Management Module CLI prompt.

19.3 Linux installation

This chapter explains how to install Red Hat Enterprise Linux (RHEL) and Novell SUSE Linux Enterprise Server (SLES) on IBM @server i5, @server p5, and @server BladeCenter JS20 using the CDs and configure the YaBoot OpenFirmware boot loader. For information about performing a network install of either distribution, refer to Chapter 6, “Installing Linux,” in the IBM Redbook *The IBM @server BladeCenter JS20*, SG24-6342:

<http://www.redbooks.ibm.com/abstracts/sg246342.html>

This section is applicable to @server BladeCenter JS20 and the @server i5 and @server p5 platforms.

19.3.1 eServer i5 and eServer p5

After configuring the system to boot from the CD-ROM using the SMS menu, insert the first CD from your RHEL or SLES distribution. Select **Normal Mode Boot**, followed by selecting **1. Yes** to exit System Management Services.

Refer to 19.3.3, “Red Hat” on page 279, or 19.3.4, “Novell SUSE” on page 280, for additional information about installing Linux on @server i5 and @server p5 systems or a partition.

19.3.2 eServer BladeCenter JS20

After configuring the system to boot from the CD-ROM using the BladeCenter Management Module, insert the first CD from your RHEL or SLES distribution. Select **Blade Tasks** → **Power/Restart** and connect to the console using SoL.

Refer to 19.3.3, “Red Hat” on page 279, or 19.3.4, “Novell SUSE” on page 280, for additional information about installing Linux on a BladeCenter JS20 blade.

19.3.3 Red Hat

For information about installing Linux on @server i5 and @server p5 systems or a partition or a JS20 blade server, refer to *Installation Guide for the IBM POWER Architecture*, available at:

<http://www.redhat.com/docs/manuals/enterprise/>

These are the steps for performing a *basic* install of RHEL on @server i5 and @server p5 systems or a partition or a JS20 blade server from a CD:

1. At the yaboot prompt (boot:), press Enter to begin the installation.

Note: Refer to “Additional RHEL installation boot options” on page 280 for information about providing additional options, such as using VNC.

2. In the CD Found window, click **Skip** if you do not want to test the CD media, and go to step 3. Otherwise, click **OK** if you want to test the CD media before you proceed with the installation.
 - a. In the Media Check window, click **Test** to test the CD currently in the CD-ROM drive.
 - b. If the media check result is **PASS**, click **OK** and proceed with the installation from that CD.
 - c. If the CD is ejected after the media check, reinsert the CD into the CD-ROM drive.
 - d. Click **Continue**.
3. In the Red Hat Enterprise Linux AS welcome window, click **OK**.
4. Select your language; then click **OK**.
5. In the Disk Partitioning Setup window, select **Autopartition**. If you get a warning that says the drive will be initialized and all data will be lost, select **Yes**.
6. In the Partitioning window, you can see how the devices will be allocated by the Linux installation. Click **OK**. Be aware that you will need approximately 3 GB allocated for the root (/) partition.

Important: The @server i5 and @server p5 installations *require* /sda1 to be a PReP boot partition. Changing the size of or removing this partition will cause the system to fail during boot.

7. Configure the appropriate network settings for your environment.

8. When your network configuration is complete, the Firewall and SELinux windows open. Click **OK** if you want to enable the default configurations.
9. In the Language Support window, select any additional languages that you want to use on this Linux installation, and then click **OK**.
10. In the Time Zone Selection window, select **System clock uses UTC** and select the appropriate time zone for your system. Click **OK**.
11. Set the root password for your system. Click **OK**.
12. Select what packages to install in the Package Defaults window. Click **OK**.
13. In the Installation to begin window, click **OK**.
14. The installation process prompts you to change the CD several times. After the installation is complete, the system will inform you to remove the CD and reboot.

Note: The SMS menu should be invoked after reboot on @server i5 and @server p5 systems to ensure that the boot device order has the installed hard drive as the first device. JS20 blade servers should have the hard disk listed as second in the boot order. If not, update this using the BladeCenter Management Module.

Additional RHEL installation boot options

For the complete list of boot and kernel options for the RHEL installation program, refer to the following Web site:

<http://www.redhat.com/docs/manuals/enterprise/RHEL-4-Manual/ppc-multi-install-guide/ap-bootopts.html>

19.3.4 Novell SUSE

For information about installing Linux on @server i5 and @server p5 systems or a partition, refer to the *SUSE Linux Enterprise Server 9 Architecture-Specific Information for IBM POWER*, available at:

<http://www.novell.com/documentation/sles9/index.html>

These are the steps for performing a *basic* installation of SLES on @server i5 and @server p5 systems or a partition from a CD:

1. At the prompt (boot:), press Enter to begin the installation.

Note: Refer to the “Additional SLES install boot options” on page 282 for information about providing additional options, such as using VNC.

2. At the prompt, What type of terminal do you have?, select **4) X Terminal Emulator (xterm)**.
3. After you have read the License Agreement, select **I Agree**.
4. Select your language. Click **Accept**.
5. Select **New Installation**.
6. Select **Partitioning**.
7. In the “Suggested Partitioning” window, select **Create custom partition setup**, and then click **Next** for a list of the available hard drives.

Important: @server i5 and @server p5 installations *require* /sda1 to be a PReP boot partition. Changing the size of or removing this partition will cause the system to fail during boot.

8. Select the drive you want to use, and then click **Next**.
9. To create a full system partition, click **Use entire hard disk**.
10. A warning appears; select **Yes, install** to begin the installation. During this part of the installation procedure, you might need to change the CD. Follow the prompts. At the end of this phase of the installation, the system restarts.

Note: The SUSE Linux Enterprise Server installer automatically sets the SMS boot order for the @server i5 and @server p5 systems, so the partition restarts from the correct device. The boot order for a JS20 blade server should be changed using the BladeCenter Management Module.

11. After the boot : prompt appears, the system starts automatically after several seconds.
12. When the installation window reopens, you are prompted to create a password for root. Enter the password for the root user twice, as prompted. Click **Next**.
13. In the Network Configuration window, select **Change**.
14. Select **Network Interfaces**.
15. In the Network cards configuration window, select **Change** under Ethernet card 0.
16. Select **Edit**.
17. Select and enter the settings appropriate for your environment, such as IP address, host name and name server, and routing. Click **Next**.
18. In the Network cards configuration overview window, click **Finish**. Click **Next**.

19. In the Test Internet Connection window, select **No, Skip This Test**. Click **Next**.
20. In the Service Configuration window, make sure that the Common Name and the Server Name fields have the correct values for your environment. Click **Next**.
21. For User Authentication Method, select **Local (/etc/passwd)**. Click **Next**.
22. For LDAP Client Configuration, click **Next**.
23. In the Add a New LDAP User window, if you want to create a new user, do so here. Otherwise, do not create a new user. Click **Next**, and select **Yes** if you are warned about Empty user login.
24. After you have read the Release Notes window, click **Next**.
25. In the Hardware Configuration window, click **Next**.
26. When you see the Installation Completed window, click **Finish**.
27. After all of the settings are configured, the system exits the installation procedure and a login prompt is displayed.

Additional SLES install boot options

A partial list of options that can be added to the boot prompt (boot:) command line before beginning installation include:

- ▶ **Install:** *URL* (nfs, ftp, hd, and so on)
Specifies the installation source as a URL. Possible protocols include cd, hd, nfs, smb, ftp, http, and tftp. The URL syntax corresponds to the common form used in Web browsers, for example:
 - `nfs://<server>/<directory>`
 - `ftp://[user[:password]@]<server>/<directory>`
- ▶ **Netdevice:** *<eth0>*
The Netdevice: keyword specifies the interface that linuxrc should use, if there are several Ethernet interfaces available.
- ▶ **HostIP:** *<10.10.0.2>*
Specifies the IP address of the host.
- ▶ **Gateway:** *<10.10.0.128>*
This specifies the gateway through which the installation server can be reached, if it is not located in the subnetwork of the host.
- ▶ **Proxy:** *<10.10.0.1>*
The Proxy: keyword defines a proxy for the FTP and HTTP protocols.

- ▶ ProxyPort: <3128>
This specifies the port used by the proxy, if it does not use the default port.
- ▶ Textmode: <0|1>
This keyword enables starting YaST in text mode.
- ▶ VNC: <0|1>
The VNC parameter controls the installation process through VNC, which makes the installation more convenient for hosts that do not have a graphical console. If enabled, the corresponding service is activated. Also see the VNCPassword keyword.
- ▶ VNCPassword: <password>
This sets a password for a VNC installation to control access to the session.
- ▶ UseSSH: <0|1>
This keyword enables access to linuxrc through SSH when performing the installation with YaST in text mode.
- ▶ SSHPassword: <password>
This sets the password for the user root to access linuxrc.
- ▶ Insmod: <module parameters>
This specifies a module the kernel should load and any parameters needed for it. Module parameters must be separated by spaces.
- ▶ AddSwap: <0|3|/dev/hda5>
If set to 0, the system does not try to activate a swap partition. If set to a positive number, the partition corresponding to the number is activated as a swap partition. Alternatively, specify the full device name of a partition.

19.3.5 YaBoot OpenFirmware boot loader

The POWER technology-based platform uses the YaBoot boot loader, which is loaded by OpenFirmware after identifying the PReP boot partition on the disk. Only disks containing the PReP boot partition can be selected in SMS.

YaBoot starts and scans all the primary disk partitions, searching for its own configuration file (usually /etc/yaboot.conf) and a kernel image to load. YaBoot unfortunately cannot scan software mirrored md devices or LVM logical volumes. It also contains a built-in knowledge of only some file systems:

- ▶ ext2
- ▶ ext3
- ▶ reiserfs
- ▶ fat

For information about configuring the YaBoot bootloader, refer to your distribution's POWER technology-specific documentation and the official YaBoot Web site located at:

<http://penguinppc.org/bootloaders/yaboot/>

19.4 eServer i5 and eServer p5 virtualization

The IBM POWER5 technology-based processor provides some unique virtualization capabilities that you can use in a Linux-only solution. Some of the capabilities that we mention in this section include:

- ▶ Shared processing
- ▶ Virtual SCSI
- ▶ Virtual Ethernet
- ▶ Virtual console

Shared processing allows logical partitions to share the processors in the shared processor pool. The shared processor pool includes all processors on the server that are not dedicated to specific logical partitions. Each logical partition that uses the shared processor pool is assigned a specific amount of processor power from the shared processor pool.

Virtual SCSI consists of a virtual SCSI server and multiple virtual SCSI clients. The virtual SCSI server is configured to serve block devices to corresponding virtual SCSI clients. All of the configuration is done on the virtual SCSI server side, while the virtual SCSI client detects the devices as regular SCSI devices when the virtual SCSI client kernel module is loaded. The virtual SCSI client can also rescan the virtual SCSI bus if new SCSI devices have been configured on the virtual SCSI server.

Virtual Ethernet enables you to create a separate virtual local area network (VLAN) that can be shared among multiple partitions. Virtual Ethernet adapters are created on each partition that is to be part of the VLAN. One adapter can also be configured as a trunk adapter, which can then be bridged to a physical adapter in the partition. This allows a route to an external network for other clients connected to the VLAN.

Virtual consoles consist of virtual console servers and their corresponding virtual console clients. Each Linux partition contains two virtual console servers by default, with the HMC providing the default console client for each partition. However, a Linux virtual console client can also connect to this virtual console server using the `/dev/hvcs` device. Typically, one partition contains all of the virtual console clients for the remaining client partitions in a system.

Table 19-2 shows the supported virtual functions found in both @server-supported Linux distributions and the IBM VIO Server.

Table 19-2 Supported virtual I/O functions

Virtual I/O function	RHEL 4	SLES 9	IBM VIO Server
Shared processing	Yes	Yes	Yes
Virtual SCSI server	No	Yes	Yes
Virtual SCSI client	Yes	Yes	N/A
Virtual Ethernet server	Yes	Yes	Yes
Virtual Ethernet client	Yes	Yes	N/A
Virtual console server	Yes	Yes	Yes
Virtual console client	Yes	Yes	N/A

Although IBM VIO Server is based on IBM AIX 5L, it is quite different. You only use the VIO Server commands; some standard AIX 5L features are not present, and some typical AIX 5L setups are not allowed, such as Logical Volume Manager (LVM)-level mirroring of VIO client logical volumes. For in-depth information about this and POWER virtualization, refer to the IBM Redbook *Advanced POWER Virtualization on IBM System p5*, SG24-7940.

The following topics describe how to create profiles for a virtual I/O server and client partition and configure them in Linux:

- ▶ Create a virtual I/O server partition profile
- ▶ Create a client partition profile
- ▶ Configure Linux on a virtual I/O server
- ▶ Configure Linux on a virtual client partition

19.4.1 Create a virtual I/O server partition profile

The virtual I/O server is the most important partition you will create because it provides the virtual adapters to the other client partitions. After the system is connected to the HMC, you can proceed with partitioning it. This section covers the process of creating the virtual I/O server partition using an HMC.

1. To begin, right-click **Partitions** under the system name (from the HMC) and select **Create** → **Logical Partition** to start the Create Logical Partition Wizard.
2. Select **Virtual I/O server** and assign the partition the name, as shown in the example in Figure 19-15 on page 286.

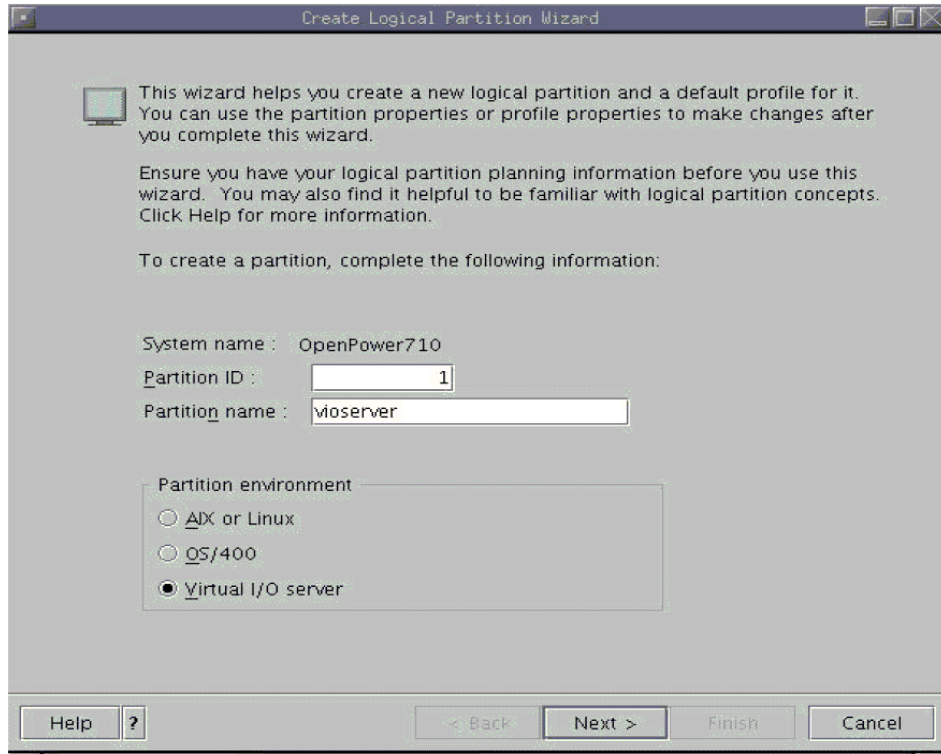


Figure 19-15 Create Logical Partition Wizard

3. Take the default (**No**) for the Workload Management Groups function.
4. Assign a name to the profile.
5. Select the amount of memory you want to assign to the partition.
6. Select at least one **Dedicated** processor.
7. Assign physical I/O devices, such as CD/DVD drives and physical network adapters, to the partition.
8. Click **Next** in the I/O Pools window, because this function is not supported in Linux.
9. Select **Yes, I want to specify virtual I/O adapters**, as shown in Figure 19-16 on page 287.

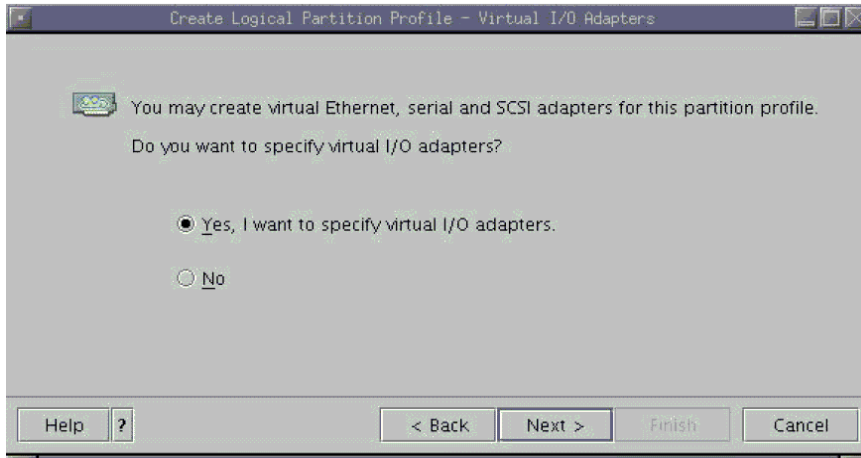


Figure 19-16 Create Logical Partition Profile - Virtual I/O Adapters

10. The next window enables you to create all of the virtual I/O adapters you will use in the virtual I/O server. See Figure 19-17 for an example.

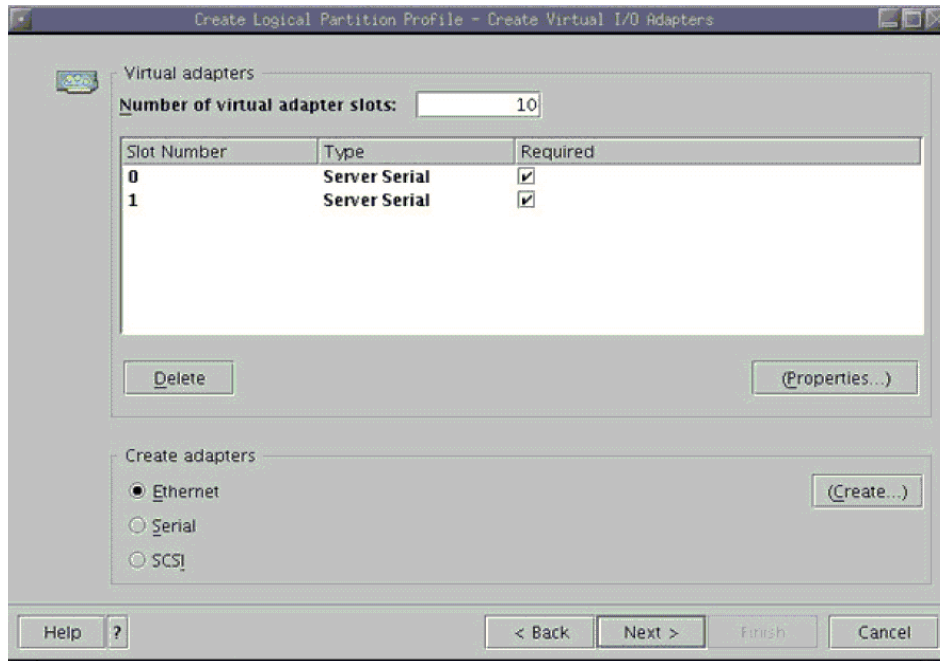


Figure 19-17 Create Logical Partition Profile - Create Virtual I/O Adapters

- For the virtual Ethernet adapter, select **Trunk adapter**, as shown in Figure 19-18, and then click **OK** to return to the previous menu. In order to bridge two networks, one adapter in the virtual network must be designated as the trunk adapter. The trunk adapter manages all frames with unknown MAC addresses and routes them to the physical network. This virtual Ethernet adapter will be bridged with the VIO server's physical Ethernet adapter to an external network.

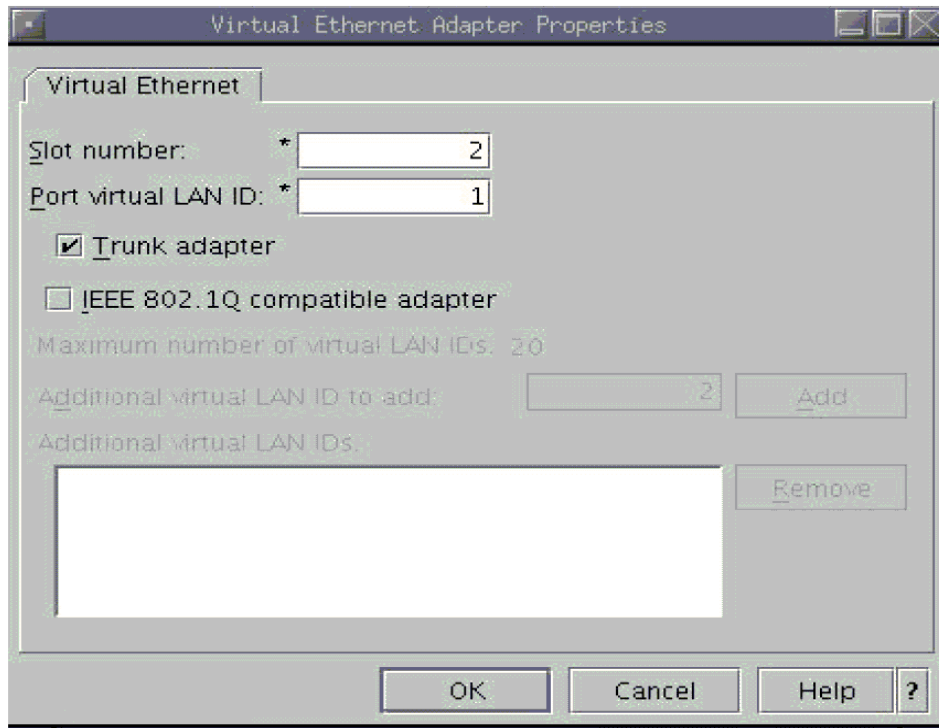


Figure 19-18 Virtual Ethernet Adapter Properties

- For the virtual SCSI, leave **Server** selected for the Adapter Type (as shown in Figure 19-19) because this partition provides the disk or block devices to the client partitions. You can also leave **Any remote partition and slot can connect** selected because you do not know which remote partitions this slot should map to yet. After you create the client partitions, you can come back and change this to the correct client partition and slot number.

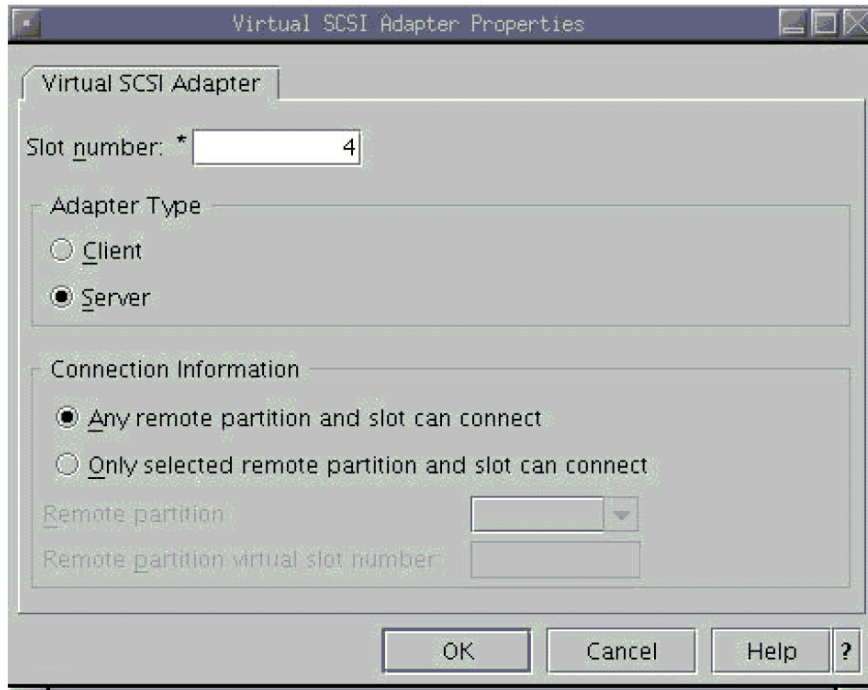


Figure 19-19 Virtual SCSI Adapter Properties

Click **OK** to create the adapter and return to the Create Virtual I/O Adapter window.

Serial adapters are not created until the client partitions are created later.

11. Figure 19-20 is an example of what you should see after creating your virtual SCSI and Ethernet adapters.

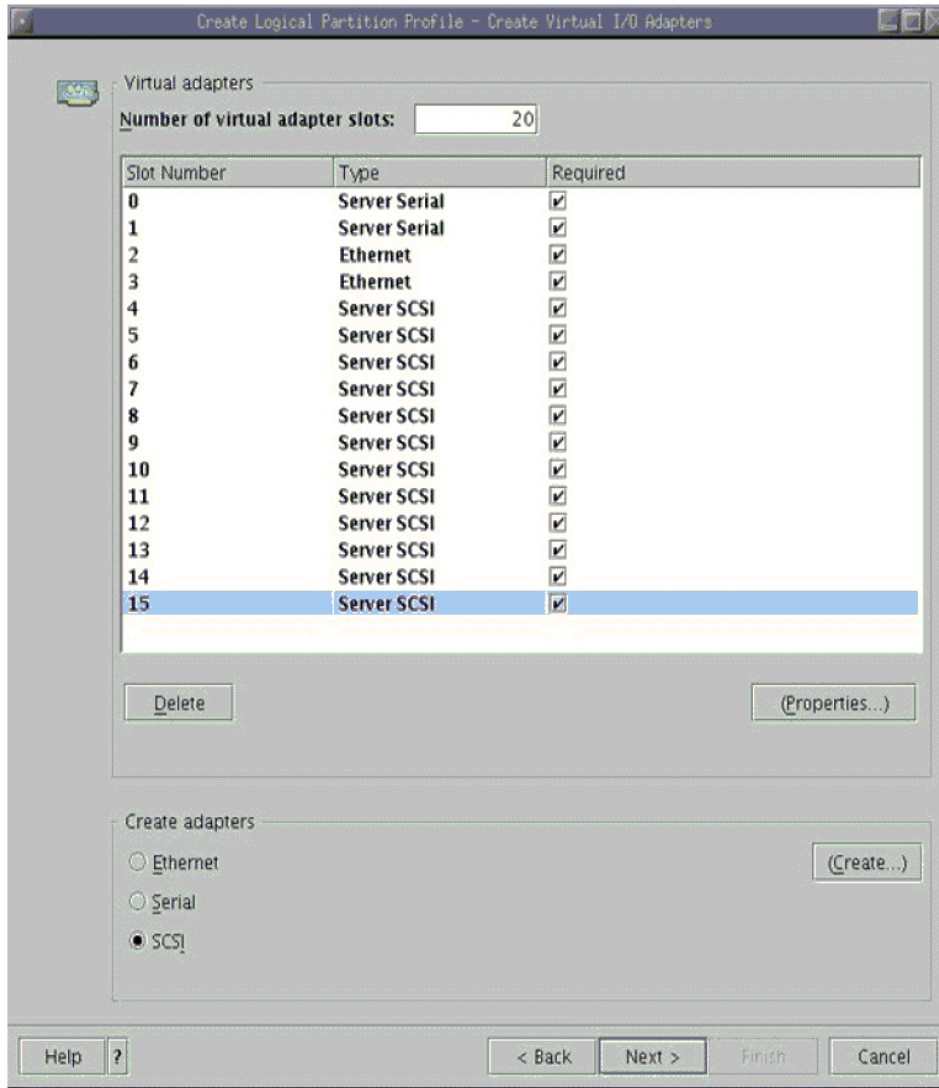


Figure 19-20 Example of created virtual I/O adapters

12. Click **Next** in the Create Virtual Adapters window.

13. Click **Next** in the Power Controlling Partition window, because this function is not supported in Linux.

14. Select desired optional settings and **Normal** under the Boot modes section of the Optional Settings menu. Click **Next**.
15. Click **Finish** in the Profile Summary window.

19.4.2 Create a client partition profile

Creating client partitions is similar to creating the virtual I/O server, except the client partitions only use virtual I/O adapters. This section covers the process of creating a client partition using an HMC. Perform the following steps:

1. To begin, right-click **Partitions** under the system name (from the HMC) and select **Create** → **Logical Partition** to start the Create Logical Partition Wizard.
2. Select **AIX or Linux** and assign the partition the name.
3. Select the amount of memory you want to assign to the partition.
4. Unlike the in the virtual I/O server partition, assign shared processors to the client processors. Make sure to select **Shared** from the Processors menu. Then, click **Next**.
5. The remaining processors need to be divided among the client partitions. The processing unit settings shown in Figure 19-21 illustrate an example of a client partition. Refer to the IBM Redbook *Advanced POWER Virtualization on IBM System p5, SG24-7940*, for more information about shared processors.

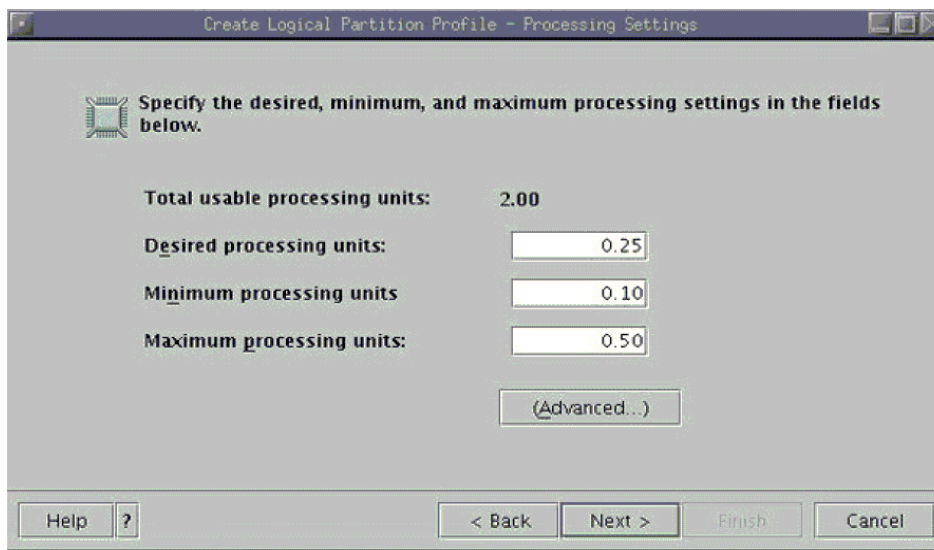


Figure 19-21 Example of client profile processor settings

Click **Next** after configuring the processor settings.

6. Usually, there are no physical devices to assign because the client partitions only use virtual I/O. Click **Next** in the I/O window to continue.
7. Select **Yes, I want to specify virtual I/O adapters**, as shown in Figure 19-16 on page 287.
8. The next window enables you to create all of the virtual I/O adapters you will use in the client. See Figure 19-17 on page 287 for an example of this window.
 - For the virtual Ethernet adapter, you can leave all the defaults, as shown in Figure 19-22. This is the virtual LAN that will be bridged to the physical network.

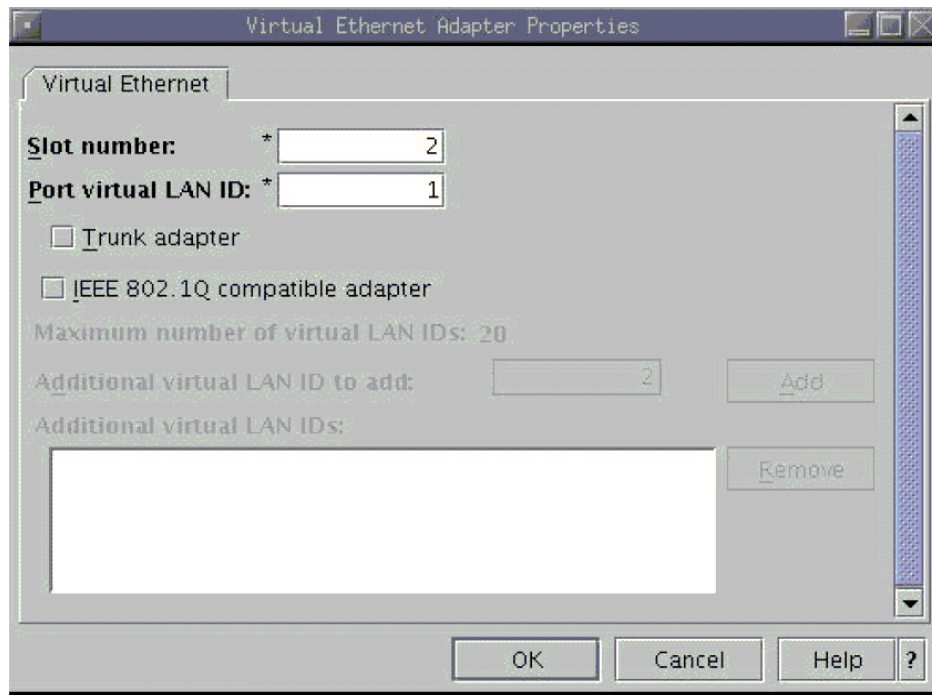


Figure 19-22 Client Virtual Ethernet Adapter Properties

Click **OK** to create the adapter and return to the Create Virtual I/O Adapter window.

- For the virtual SCSI, select **Client** for the Adapter Type (as shown in Figure 19-23).

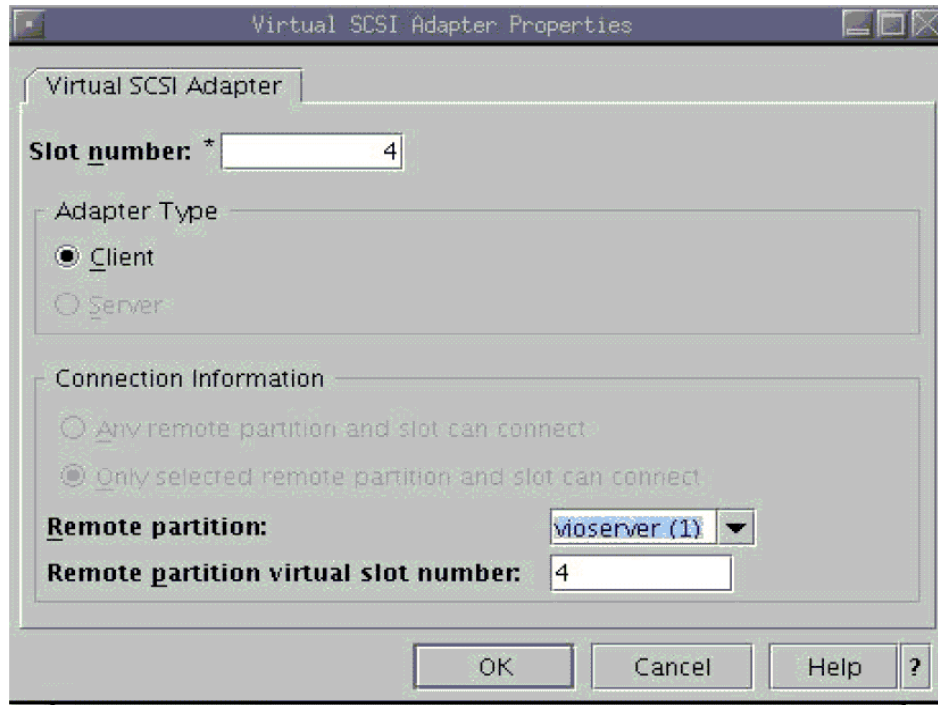


Figure 19-23 Client virtual SCSI Adapter Properties

Map the remote partition and slot number to the virtual SCSI server adapter slot you created for the virtual I/O server created earlier. For example, the first virtual SCSI adapter created for the virtual I/O server in 19.4.1, “Create a virtual I/O server partition profile” on page 285 used slot 4. So enter 4 in the Remote partition virtual slot number field for this partition. Then, click **OK**.

Serial adapters are not created for client partitions.

9. Click **Next** in the Create Virtual Adapters window.
10. Click **Next** in the Power Controlling Partition window, because this function is not supported in Linux.
11. Select desired optional settings and **Normal** under the Boot modes section of the Optional Settings menu. Click **Next**.
12. Click **Finish** in the Profile Summary.

13. Return to the virtual I/O server profile to create the virtual serial client adapter. Start by right-clicking the default profile under the virtual I/O server partition and selecting **Properties**.
14. Go to the Virtual I/O tab and under Create adapters, select **Serial** and click **Create**. This launches a separate window to configure the properties of the virtual serial adapter.
15. Leave the type as **Client** and assign the Remote Partition to the corresponding client partition. Enter 0 for the Remote partition virtual slot number, as shown in Figure 19-24.

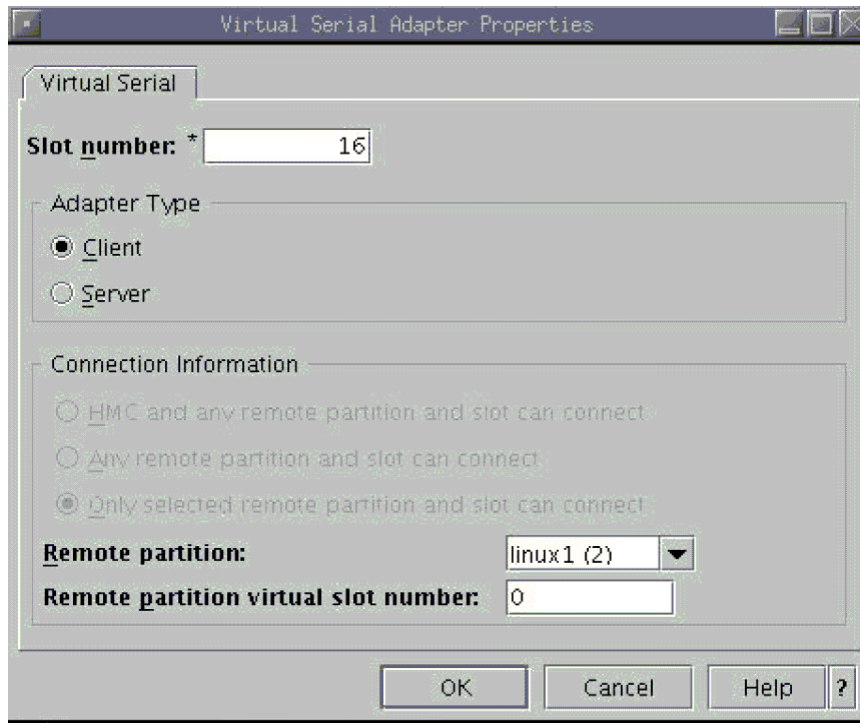


Figure 19-24 Virtual I/O server Virtual Serial Adapter Properties

16. You can now go through each of the virtual server SCSI adapters you created earlier and assign the correct remote partition to them. For each of the virtual Server SCSI adapters, select **Only selected remote partition and slot can connect** in the Connection Information section.
17. Mapping the slot numbers is a little trickier. You must assign the correct client partition and slot to its corresponding SCSI server slot. Figure 19-25 on page 295 shows an example of the mapping between a SCSI server adapter and an associated SCSI client adapter.

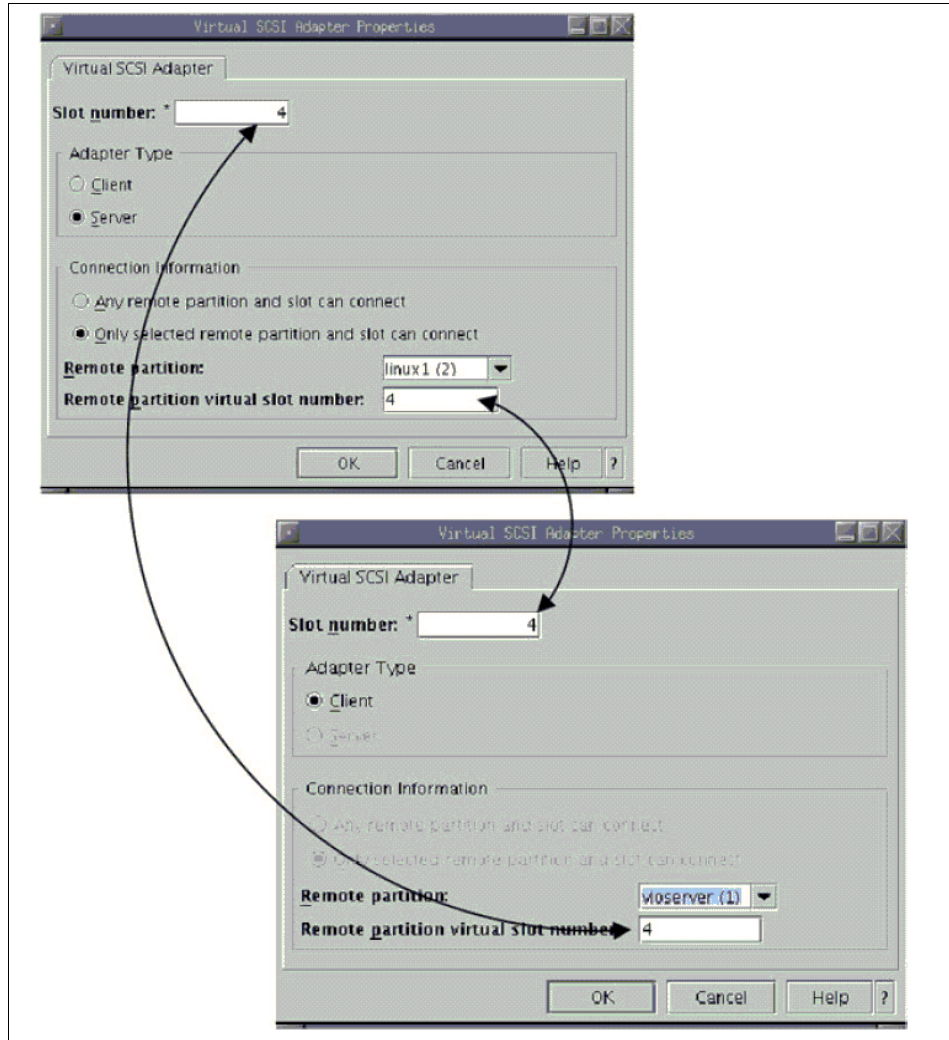


Figure 19-25 Virtual SCSI client/server mapping example

The client and server configuration is now complete. The next steps involve installing and configuring Linux to recognize and use the virtual devices on the server and client.

19.4.3 Configure Linux on a virtual I/O server

In this section, you learn how to set up the services and devices the virtual I/O server will provide. First, you must install SLES9 SP1 or later on the virtual I/O server partition. SLES9 SP1 and later is currently the only IBM-supported

distribution that contains the virtual SCSI server driver. After installing SLES9 SP1 or later, install the bridge-utils and ckermit packages found in your SLES installation media.

Configure the Ethernet

This section describes the steps for configuring the physical and virtual Ethernet devices and the process involved in bridging these devices together on the virtual I/O server. This exercise uses a dual-port Gigabit Ethernet adapter. In this example, one of the ports is configured to connect to the external network, while the second port is bridged to the virtual network and provides a route to the external network. Figure 19-26 is a graphical representation of what we are configuring.

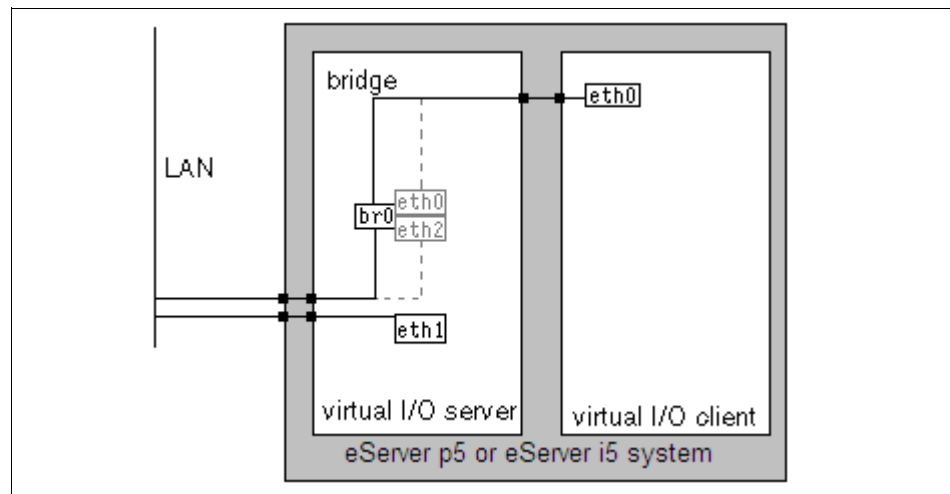


Figure 19-26 Diagram of virtual LAN configuration

Ethernet device configuration

We recommend using YaST to configure the Ethernet devices:

- ▶ Configure the first physical device attached to the external network according to your LAN configuration.
- ▶ Do not configure the second physical or virtual Ethernet device.

Physical and virtual network bridging

Now that all of the Ethernet devices are configured, it is time to set up the bridging between the physical interface and the virtual interface. You do this by using tools found in the bridge-utils package and network scripts associated with the physical interface. Using the `ifconfig` command and the MAC address (from when you configured the interface) of the physical interface, you can determine

which interface is your second physical interface. For this example, it happens to be eth2. You can also use the same method to determine the first virtual Ethernet interface, which is eth0 for this exercise. Example 19-1 shows both of these examples.

Example 19-1 ifconfig output

```
eth2  Link encap:Ethernet HWaddr 00:0D:60:DE:01:A9
      inet6 addr: fe80::20d:60ff:fede:1a9/64 Scope:Link
            UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
      RX packets:794359 errors:0 dropped:0 overruns:0 frame:0
      TX packets:16 errors:0 dropped:0 overruns:0 carrier:0
      collisions:0 txqueuelen:1000
      RX bytes:56965484 (54.3 Mb) TX bytes:1124 (1.0 Kb)
      Base address:0xec00 Memory:b8100000-b8120000

eth0  Link encap:Ethernet HWaddr 2A:34:F0:00:10:02
      inet6 addr: fe80::2834:f0ff:fe00:1002/64 Scope:Link
            UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1
      RX packets:0 errors:0 dropped:0 overruns:0 frame:0
      TX packets:794177 errors:0 dropped:0 overruns:0 carrier:0
      collisions:0 txqueuelen:1000
      RX bytes:0 (0.0 b) TX bytes:53775670 (51.2 Mb)
      Interrupt:184
```

Notice how neither interface has an IP address defined.

Next, use the bridging tools available in the bridge-utils package to create a couple scripts that configure the bridge when eth2 is activated and take down the bridge when eth2 is deactivated. The network configuration files for SLES9 are located in /etc/sysconfig/network and are labeled ifcfg-eth-id-<MAC_ADDRESS>. Using the MAC address of eth2, you can identify which configuration file needs to be edited. For this exercise, the configuration file is:

```
/etc/sysconfig/network/ifcfg-eth-id-00:0d:60:de:01:a9
```

The variable POST_UP points to a script that will run after the device is activated, while POST_DOWN points to a script that will run after the device has been deactivated. The two scripts are **bridge-up.sh** and **bridge-down.sh** and can be stored in /usr/local/bin or /etc/sysconfig/network/. Set these variables in your Ethernet configuration file. You should have entries that look similar to these in your configuration file:

```
POST_UP_SCRIPT='/etc/sysconfig/network/bridge-up.sh'
```

```
POST_DOWN_SCRIPT='/etc/sysconfig/network/bridge-down.sh'
```

Example 19-2 shows a sample **bridge-up.sh** script with comments describing each line.

Example 19-2 Sample bridge-up.sh script

```
#!/bin/sh

# the ethX device that called this script
CFG=$1 # the ethX device that called this script

# source the eth config script to set some variables
. /etc/sysconfig/network/ifcfg-$CFG

# create the ethernet bridge
/sbin/brctl addbr br0

# reset addresses for virtual ethernet (eth0)
/sbin/ifconfig eth0 0.0.0.0

# reset address for physical ethernet (eth2)
/sbin/ifconfig eth2 0.0.0.0

# add the virtual ethernet to the bridge
/sbin/brctl addif br0 eth0

# add the physical ethernet to the bridge
/sbin/brctl addif br0 eth2

# bring up the bridge interface with the IP
# address our physical interface was using
/sbin/ifconfig br0 $IPADDR netmask $NETMASK
```

Example 19-3 shows a sample **bridge-down.sh** script with comments describing each line.

Example 19-3 Sample bridge-down.sh script

```
#!/bin/sh

# bring down virtual ethernet device eth0
ifconfig eth0 down

# bring down physical ethernet device eth2
ifconfig eth2 down

# delete eth0 from bridge interface br0
brctl delif br0 eth0

# delete eth2 from bridge interface br0
```



```
brctl delif br0 eth3

# bring down bridge interface br0
ifconfig br0 down

# remove bridge instance
brctl delbr br0
```

Using the techniques described here, you should now have all three Ethernet interfaces configured in the virtual I/O server partition (see Figure 19-26 on page 296).

Configure the virtual console

The virtual console is provided by the IBM Hypervisor Virtual Console kernel module. You can load this on the virtual I/O server partition with the command:

```
# modprobe hvcs
```

Add this command to `/etc/rc.d/boot.local` so that it executes on every boot. Looking at your kernel messages in `/var/log/messages`, you will see something similar to the following:

```
Initializing IBM hvcs (Hypervisor Virtual Console Server) Driver
vio_register_driver: driver hvcs registering
HVCS: Added vty-server@30000010.
HVCS: driver module inserted.
```

Note: Slots are in hexadecimal in the kernel log. For example, 0x10 above is slot 16.

Using these mappings, you can create console connections on the virtual I/O server partition to the corresponding client partition. For this tutorial, we use the **kermit** serial program, which is part of the **ckernit** software package. The following command creates a console connection to the client:

```
# kermit -l /dev/hvcs0 -c
```

The device `/dev/hvsc0` is the first device and maps to the first `vty-server`. If additional clients were configured, they would map accordingly. After you run the **kermit** command, you should see a screen similar to the following:

```
Connecting to /dev/hvcs0, speed 38400
Escape character: Ctrl-\ (ASCII 28, FS): enabled
Type the escape character followed by C to get back,
or followed by ? to see other options.
-----
```

Configure the virtual SCSI server

Now that all the network services are up and running, the last part of setting up your virtual I/O server is configuring the SCSI disks that will be served to the client partition. The IBM virtual SCSI server module is called `ibmvscsis`. Add this module to `/etc/rc.d/boot.local` with the following line:

```
# modprobe ibmvscsis
```

This causes the module to load automatically on boot.

The virtual SCSI server supports 64 target devices per bus and eight buses per virtual adapter. It defaults to one target and one bus per virtual adapter when the IBM virtual SCSI module is loaded.

The virtual SCSI adapter supports multiple block devices, including an entire SCSI disk, a partition of a SCSI disk, and a shared file mounted over a loopback device.

Modify a virtual SCSI adapter's settings through the `sysfs` file system (see “/sys” on page 90). The virtual SCSI adapters are located under `/sys/devices/vio` and all start with `300000xx`, where `xx` is the slot number of the SCSI adapter.

If the virtual I/O server partition has slot 4 configured as its first virtual SCSI adapter. It is represented in the `sysfs` file system under `/sys/devices/vio/30000004/`. To modify the number of buses this adapter supports, you must modify the setting `/sys/devices/vio/30000004/num_buses`. To modify the number of targets a certain bus supports, `bus0` in this example, you modify `/sys/devices/vio/30000004/bus0/num_targets`.

Here is an example of how to configure one adapter:

1. Set the number of targets to 1 for this bus:

```
# echo 1 > /sys/devices/vio/30000004/bus0/num_targets
```

2. Assign the block device `/dev/sdb` to this adapter:

```
# echo /dev/sdb > /sys/devices/vio/30000004/bus0/target0/device
```

```
# echo b > /sys/devices/vio/30000004/bus0/target0/type
```

3. Do not enable read-only “connect” mode:

```
# echo 0 > /sys/devices/vio/30000004/bus0/target0/ro
```

4. Activate the device:

```
# echo 1 > /sys/devices/vio/30000004/bus0/target0/active
```

You can add a function to `/etc/rc.d/boot.local` to make setting up the virtual SCSI adapters easier, as shown in Example 19-4 on page 301.

Example 19-4 Virtual SCSI function in boot.local

```
#!/bin/sh
#
# Copyright (c) 2002 SuSE Linux AG Nuernberg, Germany. All rights reserved.
#
# Author: Werner Fink <werner@suse.de>, 1996
# Burchard Steinbild, 1996
#
# /etc/init.d/boot.local
#
# script with local commands to be executed from init on system startup
#
# Here you should add things, that should happen directly after booting
# before we're going to the first run level.
#

# load the Hypervisor Virtual Console driver
modprobe hvcs

# load the IBM Virtual SCSI server driver
modprobe ibmvscsis

function vscsisetup {
    if [ $# -ne 2 ]; then
        echo "vscsisetup: slot_number physical_device"
        exit 1
    fi
    vslot=$1
    physical_device=$2
    if [ ! -e /sys/devices/vio/$vslot ]; then
        echo "vscsisetup: Error - Slot number $vslot does not exist"
        exit 1
    fi
    if [ ! -e $physical_device ]; then
        echo "vscsisetup: Error - physical device $physical_device does not
exist"
        exit 1
    fi
    echo 1 > /sys/devices/vio/$vslot/bus0/num_targets
    echo $physical_device > /sys/devices/vio/$vslot/bus0/target0/device
    echo b > /sys/devices/vio/$vslot/bus0/target0/type
    echo 0 > /sys/devices/vio/$vslot/bus0/target0/ro
    echo 1 > /sys/devices/vio/$vslot/bus0/target0/active
    rc=`cat /sys/devices/vio/$vslot/bus0/target0/active`
    if [ $rc -ne '1' ]; then
        echo "IBMVSCSI Server configuration failed for device $vslot"
    fi;
}
}
```

```
# assign block devices to virtual SCSI adapters
# setup server slot 4 with /dev/sdb assigned to client partition
vscsisetup 30000004 /dev/sdb
```

The virtual I/O server should be configured at this point.

For this tutorial about virtualization, we do a CD installation on the client. In order to accomplish this, we must relocate the CD/DVD device to the client partition for installation. For additional installation methods, refer to *Linux virtualization on POWER5: A hands-on setup guide* on IBM developerWorks:

<http://www.ibm.com/developerworks/edu/1-dw-linux-pow-virtual.html>

Perform the following steps:

1. Power off the virtual I/O server partition.
2. Remove the CD/DVD device from the virtual I/O server partition profile. Refer to “Hosted (partitioned)” on page 262 for details.
3. Add the CD/DVD device to the client partition profile.
4. Power on the virtual I/O server partition and ensure that all the services and virtual functions are set up properly.

19.4.4 Configure Linux on a virtual client partition

This section describes the steps involved in installing Linux on a client partition using CDs. Perform the following steps:

1. Insert the first CD from the RHEL4 installation media into your @server i5 or @server p5 system.
2. Activate the client partition through the HMC.
3. Open a terminal to the client partition.
4. Use the SMS menus to boot from the CD. Refer to “Hosted (partitioned)” on page 262 for details.

Both supported Linux distributions will automatically load virtual SCSI client and virtual Ethernet drivers before beginning the installation. You can now proceed to install the partition without any special virtualization related steps. Refer to 19.3.3, “Red Hat” on page 279, for information about installing RHEL4 and 19.3.4, “Novell SUSE” on page 280, for information about installing SLES9 on an @server i5 or @server p5 partition.

19.5 POWER technology platform service and productivity tools

There are various hardware service diagnostic aids and productivity tools available for @server i5 and @server p5 and JS20 blade systems running Linux. These service and productivity tools perform the following functions:

- ▶ Base tools
 - Access POWER technology-based platform features that are provided by the system's firmware
 - Gather hardware inventory and microcode level
 - Access system boot lists, LEDs, reboot policies, and so on from the operating system command line
 - Communicate with an attached Hardware Management Console
- ▶ Service tools
 - Analyze errors or events and perform actions to increase system availability or protect data integrity
 - Communicate errors to an attached Hardware Management Console or to IBM Service
 - Survey hardware and communicate the results to the IBM Machine Reported Product Data database
- ▶ Productivity tools
 - Hot-plug add, remove, or replace PCI devices
 - Dynamically add or remove processors or I/O slots from a running partition using an attached Hardware Management Console

Table 19-3 describes some common tool packages that are available for Linux on POWER installations.

Table 19-3 Common packages available for Linux on POWER platforms

Package	Description
librtas	Contains a library that allows applications to access certain functionality provided by platform firmware.
ppc64-utils	Enables a number of reliability, availability, and serviceability (RAS) features.
lsvpd	Contains the lsvpd , lscfg , and lsmcode commands. These commands, along with a boot-time scanning script called update-lsvpd-db , constitute a hardware inventory system.
diagela	Provides automatic analysis and notification of errors reported by the platform firmware.

19.5.1 Red Hat

All tools for RHEL4 on POWER technology-based installations are at the following Web sites:

- ▶ JS20 blade
<http://techsupport.services.ibm.com/server/lopdiags/redhat/blades>
- ▶ Monolithic @server installation
<http://techsupport.services.ibm.com/server/lopdiags/redhat/otherserver>
- ▶ Hosted @server installation
<http://techsupport.services.ibm.com/server/lopdiags/redhat/hmcserver>

19.5.2 Novell SUSE

All tools for RHEL4 on POWER installations are at the following Web sites:


- ▶ JS20 blade
<http://techsupport.services.ibm.com/server/lopdiags/suselinux/blades>
- ▶ Monolithic @server installation
<http://techsupport.services.ibm.com/server/lopdiags/suselinux/otherserver>
- ▶ Hosted @server installation
<http://techsupport.services.ibm.com/server/lopdiags/suselinux/hmcserver>

19.6 References and further readings

The following is a list of references for topics covered in this chapter.

- ▶ IBM Redbooks
<http://www.ibm.com/redbooks>
 - *Linux on IBM @server i5 Implementation Guide*, SG24-6388
 - *The IBM @server BladeCenter JS20*, SG24-6342
 - *Advanced POWER Virtualization on IBM System p5*, SG24-7940
- ▶ IBM Information Center publications
<http://publib.boulder.ibm.com/infocenter/eserver/v1r3s/index.jsp>
 - *Managing the Advanced System Management Interface (ASMI)*
 - *Managing the Hardware Management Console (HMC)*
 - *Managing your server using the Hardware Management Console*

- *Partitioning for Linux*
- *Installing Linux*
- ▶ Distribution guides
 - Red Hat: *Installation Guide for the IBM POWER Architecture*
<http://www.redhat.com/docs/manuals/enterprise/>
 - Novel SUSE: *SUSE Linux Enterprise Server 9 Architecture-Specific Information for IBM POWER*
<http://www.novell.com/documentation/sles9/index.html>
- ▶ IBM developerWorks
 - <https://www.software.ibm.com/developerworks>
 - *Linux virtualization on POWER5: A hands-on setup guide*



IBM eServer zSeries and IBM System z hardware platform specifics

Linux, when run on IBM @server zSeries, IBM System z, or the IBM S/390 hardware platform, has a set of considerations above and beyond what needs to be considered for the @server xSeries and pSeries platforms. The S/390 and zSeries platforms have different capabilities that must also be considered. We use the term “mainframe” to refer to both platforms in this section unless a specific difference or capability must be identified.

The majority of Linux management concepts, tools, and techniques are the same on all hardware platforms. This chapter describes resources, concepts, tools, and techniques specific to Linux and IBM mainframes.

In this chapter, we discuss the following topics:

- ▶ Planning
- ▶ S/390 and zSeries overview
- ▶ Installation methods and techniques
- ▶ Booting, system initialization, and shutdown
- ▶ Device management

- ▶ Performance monitoring and tuning
- ▶ Troubleshooting and diagnostics
- ▶ S/390 and zSeries-specific Linux commands
- ▶ High availability
- ▶ Security
- ▶ References

20.1 Planning

As with any server, you must plan an installation. You must define the resources your applications need to function optimally. These resources, referred to as a footprint, should be the starting point of your planning. With this information, you can size and configure your server requirements appropriately.

You will need several resources to properly size, deploy, and tune Linux on a mainframe. We cover these at a high level in this section and refer you to other Web sites or documents for more in-depth information. The information in this section will help you prepare for a Solution Assurance Review (SAR). All Linux on mainframes deployments should have a SAR at the beginning of the project. Ask your IBM zSeries representative to host a SAR for you. This will be a question and answer session where your team and IBM can discuss the behavior of your applications, your requirements, and expectations, and help identify what hardware resources you will need to be successful in your deployment.

Running Linux under z/VM in an LPAR is considered the best practice. Virtual Machine (VM) has long been recognized as a robust computing platform, spanning the family of S/390 and zSeries servers. z/VM provides a highly flexible test and production environment for enterprises deploying the latest e-business solutions. z/VM supports Linux, one of the world's leading open source operating systems. Within the VM environment, Linux images benefit from the ability to share hardware and software resources and use internal high-speed communications. z/VM offers an ideal platform for consolidating workloads on a single physical server, which enables you to run tens to hundreds of Linux images. Although you can run Linux in an LPAR without z/VM, this chapter assumes that you are considering hosting Linux images with z/VM.

Refer to the following brief list of Internet references for Linux on mainframes:

- ▶ Linux for S/390 project
This Web site contains useful links to IBM Redbooks, other related Web sites, how-tos, man pages, and more:
<http://www.Linuxvm.org/>
- ▶ z/VM publications
This Web site provides links to documents for specific z/VM releases and a link to another Linux library that has articles discussing specific workloads:
<http://www.vm.ibm.com/pubs/>

- ▶ IBM @server zSeries and IBM System z
This Web site provides links to documents for specific zSeries models and a link to another Linux library that has articles discussing solutions, success stories, and technical support:

<http://www.ibm.com/servers/eserver/zseries/>

- ▶ z/VM overview
This Web site provides an overview of the features available in current releases of z/VM:

<http://www.vm.ibm.com/overview>

A set of planning worksheets is available in IBM Redbook *z/VM and Linux on zSeries: From LPAR to Virtual Servers in Two Days*, SG24-6695.

20.1.1 Hardware resources

You need the following hardware components to deploy Linux on a mainframe. A more detailed description of these components follows in subsequent sections.

- ▶ zSeries logical partition (LPAR).
- ▶ Processors or CPUs: One Integrated Facility for Linux (IFL) minimum; two or more is recommended.
- ▶ Memory: Central and expanded at a 3:1 ratio.
- ▶ Direct Access Storage Devices (DASD, or disk drives): It is helpful to have DASD on different channel-path identifiers (CHPIDs).
- ▶ Open Systems Adapter (OSA) network cards: OSA-Express in Queued Direct I/O (QDIO) recommended.
- ▶ A computer that acts as a Network File System (NFS) or File Transfer Protocol (FTP) server.

20.1.2 Software resources

You need the following software and documentation to deploy Linux on a mainframe:

- ▶ z/VM installation media and documentation
- ▶ Linux installation media and documentation
- ▶ A workstation or desktop that has network access to the mainframe:
 - A 3270 emulator:
 - Attachmate Extra, Hummingbird Host Explorer or IBM Personal Communications for Windows desktops

- A 3270 emulator named x3270 (included for free in most distributions) for Linux desktops
- A remote login program:
 - A Linux Secure Shell (SSH) client such as PuTTY (recommended) or TeraTerm for Windows desktops
 - For Linux desktops, the SSH client, which is built-in

20.1.3 Networking resources

You need the following TCP/IP resources:

- ▶ A TCP/IP address for z/VM
- ▶ One TCP/IP address for each Linux virtual server
- ▶ Associated TCP/IP information:
 - Domain Name System (DNS) host name
 - DNS domain
 - DNS server TCP/IP address
 - TCP/IP gateway
 - TCP/IP subnet mask
 - TCP/IP broadcast address (usually calculated from address and subnet mask)
 - TCP/IP MTU size

20.1.4 Linux distributions and z/VM

We discuss two distributions throughout this section: Red Hat Enterprise Linux Advanced Server V4 and Novell SUSE Linux Enterprise Server V9. Each distribution has its own configuration limits. You can find this information at the following Web sites:

http://www.redhat.com/en_us/USA/rhel/details/limits/

http://www.novell.com/products/Linuxenterpriseserver/kernel_limits.html

Linux can run on the “bare hardware” of the mainframe without any special software or support required. The actual internals of the Linux OS are identical from platform to platform. The only changes that are made in the mainframe implementation of Linux relate to the instructions that the operating system passes to the hardware. As mentioned earlier, running Linux under z/VM in an LPAR is considered the best practice.

20.2 S/390 and zSeries overview

IBM @server zSeries hardware is designed for continuous availability, so zSeries systems offer a set of reliability, availability, and serviceability features (RAS). The zSeries architecture has two instruction units to compare every operation in order to discover possible errors. The main storage and the L2 cache uses Error Checking and Correction (ECC) technology. In addition, memory sparing is implemented to replace failed memory units. CPU sparing is also available. A detailed description of the zSeries hardware can be found in the following IBM Redbooks:

- ▶ *IBM @server zSeries 900 Technical Guide*, SG24-5975
- ▶ *IBM @server zSeries 890 Technical Introduction*, SG24-6310
- ▶ *Technical Introduction: IBM @server zSeries 800*, SG24-6515
- ▶ *IBM @server zSeries 990 Technical Guide*, SG24-6947
- ▶ *IBM System z9 109 Technical Guide*, SG24-7124

20.2.1 Processing units

The multichip module (MCM) contains multiple processing units (PUs). On other platforms, these are commonly referred to as central processing units (CPUs).

All processing units (PUs) are physically the same. At power-on reset (POR), distinct microcode that is loaded in the PU determines its processor type. The processor types pertinent to z/VM and Linux are:

- ▶ Central processor (CP)
CPs are enabled to execute any operating system available on zSeries.
- ▶ Integrated Facility for Linux (IFL)
IFLs are processors enabled specifically for Linux workloads. Both z/VM and Linux (either in LPAR mode, or as a z/VM) can execute on an IFL.

Three other types of processors are available on zSeries platforms: System Assist Processor (SAP), Internal Coupling Facility (ICF) and zSeries Application Assist Processor (ZAAP). Linux does not use these processor types.

20.2.2 Memory

There are three different address space sizes for Linux distributions.

- ▶ 64 GB: 64-bit word lengths are available on Intel, AMD, POWER, and zSeries processors. A 64-bit word length, generally speaking, provides applications with a 64 GB address space with current technology.

- ▶ 4 GB: 32-bit word lengths are available on Intel and earlier POWER or RISC processors. A 32-bit word length provides applications with a 4 GB address space.
- ▶ 2 GB: 31-bit address spaces are available on S/390 and zSeries processors. One bit of the 32-bit word length is reserved as a flag for the older 24-bit address space. This shorter 31-bit word length provides applications with a 2 GB address space (approximately).

IBM S/390 only supports 31-bit address spaces. zSeries can run both 31-bit and 64-bit Linux guests. Because of the more limited address space in the 31-bit environment, we recommend that you deploy 64-bit Linux on zSeries. 31-bit applications are fully supported on 64-bit Linux on zSeries.

Central and expanded storage

IBM mainframes have two types of memory, referred to as storage:

- ▶ Central storage

Central storage is the real memory that the operating systems and applications can access.
- ▶ Expanded storage

Expanded storage is memory set aside for z/VM paging (swapping). In addition, virtual disks (VDISKs) can be created in the expanded storage. This reduces the amount of memory available for z/VM paging. The VDISK is analogous to a RAM disk on other hardware platforms.

20.2.3 The channel subsystem

I/O devices are attached to the zSeries system through the channel subsystem (CSS). Components of the channel subsystem include:

- ▶ Control unit

A control unit (CU) provides the logic to operate and control an I/O device. An I/O device can be attached to one or more control units. Control unit function can be:

 - Implemented in a separate physical unit
 - Integrated inside the I/O device
 - Integrated within the channel itself
- ▶ Channel

A channel is a specialized processor that communicates with the control unit. Channels are located in the zSeries I/O cage and manage data movement between central storage and the control unit.

▶ Subchannel

One subchannel is dedicated to each I/O device accessed by the channel subsystem. The subchannel provides information about the attached I/O device to the channel subsystem (such as the CHPID, device status, and channel path availability). Subchannels are addressed using the system-unique 16-bit subchannel number. The actual number of available subchannels depends on the system model, but is limited to a maximum of 65,536 per system.

▶ Channel path

A control unit is attached to the channel subsystem by one or more channel paths. Depending on the zSeries model and configuration, an I/O device can be accessed by as many as eight different channel paths. Types of channel paths supported on zSeries include:

- Enterprise Systems Connection (ESCON®)
- Fibre Connection (FICON®)
- Open Systems Adapter-2 (OSA-2)
- OSA-Express

Channel paths are addressed using the system-unique 8-bit channel-path identifier (CHPID). The actual number of available channel paths depends on the system model, but is limited to a maximum of 256 per system.

▶ System assist processor (SAP)

The SAP schedules an I/O operation. It finds an available channel path to the intended device and guarantees completion of the I/O operation. However, it is not involved in moving data between central storage and the channel.

ESCON

ESCON provides bidirectional serial bit transmission, in which the communication protocol is implemented through sequences of special control characters and through formatted frames of characters.

The ESCON link data rate is 200 megabits per second (Mbps), which results in a maximum aggregate data rate of up to 17 megabytes per second (MBps). The maximum unrepeated distance of an ESCON link using multimode fiber optic cables is 3 km (1.87 miles) when using 62.5 micron fiber.

FICON

FICON provides all the strengths of ESCON, plus more. The link data rate has increased from 200 Mbps to 2 Gbps, with expected effective data transfer rates between 60 to more than 170 MBps. The FICON implementation enables full duplex data transfer, so data travels in both directions simultaneously, rather than

the ESCON half-duplex data transfer. In addition, multiple concurrent I/Os can occur on a single FICON channel, a fundamental difference between FICON and ESCON.

zSeries FCP support

The Fibre Channel Protocol (FCP) feature is provided with zSeries servers equipped with a FICON card. A FICON card has two ports. Each port can serve only one LPAR to link to SCSI devices configured in an IBM TotalStorage® Enterprise Storage Server® (ESS) environment. Channel sharing is not implemented for this architecture.

zSeries FCP support enables you to connect a Linux running on zSeries 800, zSeries 900, or zSeries 990 with industry-standard SCSI devices. This means that you are able to configure your Linux to use a Storage Area Network (SAN) environment.

20.2.4 Tape drives

The Linux installation program can be loaded from tape drives. The only requirement is that a tape subsystem must be attached to the mainframe and available to the Linux guest or LPAR.

IBM supported devices such as the Enterprise Storage Server, Enterprise Tape System 3590, and Enterprise Tape Library 3494 are available. These devices can be attached through ESCON or FICON channels.

Two types of storage controllers with Fibre Channel interfaces are currently available for zSeries:

- ▶ Those supporting the z/Architecture-specific FICON interface based on Channel Control Word (CCW) architecture, with the CKD command set for disk storage and similar sets of commands for tapes
- ▶ Those supporting the SCSI-based FCP protocol, with SCSI command sets specific to the device type, such as disk or tape

Linux provides the `mt` command for control of tape devices.

20.2.5 Disk drives

Disk drives are attached to the mainframe through the channel subsystem. The protocols used to access the disk drives can be SCSI, ESCON, or Fibre Channel/FICON.

Disk drives are identified with the `/dev/dasd` prefix instead of `/dev/sd` (SCSI) or `/dev/hd` (IDE). The disk names are created in this order: `/dev/dasda` through `/dev/dasdz`, then `/dev/dasdaa`, and so on.

Linux directories that contain disk drive information are `/proc/scsi`, `proc/bus`, and `/proc/dasd`.

Disk drives can be accessed as full volumes or partitioned as z/VM minidisks. A special type of disk, the VDISK, is a non-persistent RAM disk that can be created in expanded storage.

In many cases, the FICON implementation coexists with, and can reuse, a customer's existing ESCON fiber cabling infrastructure. The FICON features on zSeries servers also support full fabric connectivity under the Linux operating system for the attachment of Small Computer System Interface (SCSI) devices using the Fibre Channel Protocol (FCP).

20.2.6 Network

Setting up real and virtual networking is described in Chapter 4 of *Linux for IBM System z9 and zSeries*, SG24-6694 and in *Networking Overview for Linux on zSeries*, REDP-3901. Red Hat and SUSE both provide specific instructions for identifying and configuring network connections.

HiperSockets

HiperSockets™ technology is available for IBM @server zSeries servers (z800, z890, z900, and z990) and IBM System z9™ 109. HiperSockets is an integrated any-to-any virtual TCP/IP network that provides interconnectivity between multiple LPAR or virtualized images under z/VM. With HiperSockets, you can connect any server running on the same zSeries and improve response time due to low latency. HiperSockets is a feature of the zSeries 800 and 900 processors. It provides a high-speed network connectivity method, which can be used to link operating systems running in logical partitions on a single zSeries processor.

Note: Connectivity in a z/VM virtual network is limited to guests running in a single z/VM image. Virtual networks cannot be used for inter-LPAR communication.

Virtual networks

Three types of virtual networks are available to Linux guests:

- ▶ Virtual channel-to-channel (VCTC):
VCTC networks provide point-to-point connectivity between guests without real channel allocation (as required for real channel-to-channel connectivity).

A real channel-to-channel (CTC) adapter is used to connect a real mainframe to another using the channel protocol. z/VM provides the ability to define virtual channel-to-channel adapters so that users can connect virtual machines using the CTCA protocol. On z/VM, this is useful for connecting Linux virtual machines to other virtual machines that do not support inter-user communication vehicle (IUCV), such as VSE/ESA™, OS/390®, and z/OS. Virtual VCTC networks can also connect Linux virtual machines.

▶ Inter-user communication vehicle (IUCV):

Point-to-point TCP/IP connections between Linux guests can be established using IUCV. IUCV is a VM-unique, virtual machine-to-virtual machine communication protocol. The Linux for S/390 and zSeries kernels include an IUCV driver, enabling you to connect two Linux virtual machines. Linux treats IUCV connections as any other TCP/IP network connection. This results in memory-speed networking between the connected Linux servers. The speed of the IUCV line is a factor of processor speed. The faster your zSeries server is, the faster your IUCV network is. When you upgrade to a faster real processor, you automatically increase the speed of your virtual network. IUCV is also supported by the z/VM TCP/IP stack, so you can use IUCV to connect Linux images to the VM TCP/IP stack.

▶ z/VM guest LAN and virtual switch:

This type of network enables local area network (LAN) connectivity between z/VM guests. With VM guest LAN and VSWITCH technology, you can connect a large number of Linux servers to a real network using fewer real network adapters, exploiting the virtual networking capabilities of z/VM to essentially share the real network connections among your Linux servers. IBM recommends that you to implement VM guest LAN over point-to-point for your network connectivity. It also recommends the VSWITCH architecture over router virtual machine.

TPC/IP direct connection

The network interface your installation uses will, to a large extent, determine whether you can support all your Linux guests using direct connection. For example, an OSA-2 interface can only support 16 entries in its OSA Address Table per port, which means that each port can be shared by only eight TCP/IP stacks. The OSA-Express interface provides a very high level of shareability, with up to 4096 IP addresses per port available.

The easiest way to directly connect Linux guests to the network is by sharing a port (or ports) on an OSA-Express interface. Each stack sharing the OSA port is directly attached to the network with an individual LAN IP address.

Under z/VM, you can dedicate the required sets of device addresses to your Linux guests, giving them direct access to the OSA hardware.

Important: If you want the ability to move an OSA CHPID from LPAR to LPAR for recovery purposes, defining the CHPID as shareable is not a good idea. Doing so reduces the maximum usable capacity of the OSA by at least half, because you need to provide the same definitions on both the production and backup LPARs (to provide access for all the Linux guests that you recover). Defining the CHPID as reconfigurable is better in this case.

If you must share the OSA with another LPAR and also use it for recovery, defining the CHPID as shareable is the only solution. You need to take into account the reduced capacity when designing your Linux guest connectivity. The candidate list is the list of LPARs that can have access to the CHPID. LPARs not defined in the candidate list cannot configure the CHPID online.

The OSA-Express IP Assist function offloads the following protocol functions from the IP stack for processing in the adapter:

- ▶ Broadcast filtering
- ▶ ARP processing
- ▶ Building MAC and LLC headers

Offloaded processing can reduce the cycles consumed by your Linux guests for network traffic, giving a slight performance improvement. In a single guest, the effect might not be significant, but in a z/VM LPAR with Linux guests generating a moderate-to-high volume of network traffic, we would expect an overall saving.

Stack-to-stack routing

When IP stacks are sharing an OSA, and one stack sends traffic destined to another stack sharing the same OSA port, the OSA sends the traffic directly to the destination stack. Network traffic processed in this way does not appear on the LAN. This action takes place transparently to the sending stack. This can provide a performance benefit when a number of guests send network traffic to each other, because it will be sent directly between the stacks rather than out across the network.

If you are considering an OSA-sharing configuration for your large-scale Linux installation, you need to be aware of some points regarding this connection method.

An IBM Technical Flash (document number Flash 10144) describes some considerations to be observed when sharing an OSA-Express port defined in QDIO mode. It details the number of LPARs or guests that can share an OSA (depending on the hardware type) and some zSeries code-level dependencies. In summary, a z900 with up-to-date microcode, or any z800, can support up to 80

IP stacks on an OSA-Express port in QDIO mode. Other combinations can support considerably fewer than that. What is not clear is the level of performance that can be expected when the OSA is configured to its peak. Performance testing to date has focussed on throughput from a single stack, rather than saturation across a fully-configured OSA. To access this Flash, enter the Flash number 10144 at the following site:

<http://www.ibm.com/support/techdocs/>

20.2.7 Printers

There are two specific considerations for creating print servers on Linux on zSeries.

Printers attached to and directly managed by z/VM that you can connect locally with S/390 and zSeries hardware platforms are not supported by CUPS. On these platforms, network printing is required. For information about configuring network printers, see *Printing with Linux on IBM eServer zSeries Using CUPS and Samba*, REDP-3864.

In general, a root file system of a single 3390-3 does not leave sufficient disk space for a print server. There are many possible configurations for laying out file systems. One suggested configuration is three 3390-3s, one for each of / (root), /usr, and /var.

20.2.8 Logical partition concepts

The Processor Resource/System Manager (PR/SM™) is a standard feature of all zSeries central processor complexes (CPCs). This allows a CPC to be divided into multiple logical partitions (LPARs). LPARs allow workloads to be isolated in different system images, so you can run production work separately from test work, or even consolidate multiple servers into a single processor.

A LPAR has the following properties:

- ▶ Each LPAR is a set of physical resources controlled by an operating system image.
- ▶ Physical processors and channels can be shared between multiple LPARs, or dedicated to single LPAR. In general, sharing processors maximizes mainframe system utilization.
- ▶ Physical storage used by an LPAR is dedicated to the LPAR.

20.2.9 Virtualization

The z/VM Control Program controls the virtual machine environment. Control Program provides each z/VM user with a working environment referred to as a virtual machine. A virtual machine is the functional equivalent a real system. Virtual machines share real resources: processors, storage, console, and input/output devices. The real resources will be managed with a user directory provided by z/VM. There does not need to be an actual correspondence between physical and logical resources for virtualization to be used.

A guest operating system such as Linux running in the virtual machine sees the virtual resources as real hardware. No modifications are required to the operating system to operate in a virtual machine; the mapping of these virtual resources to real resources is handled by the virtualization software.

With z/VM, you can grow your server workload vertically or horizontally. Vertical growth is realized when you add more system resources such as additional processing power to an existing virtual machine. You can give a Linux virtual machine more processor capacity, more virtual memory, more I/O devices and bandwidth, plus more networking bandwidth.

Horizontal growth is a typical way to grow workloads. It is easily accomplished by adding another virtual machine, or two, or three, or more on z/VM. There is an added benefit in the efficient use and sharing of system resources. This makes adding more Linux virtual servers a particularly attractive option when you consider the limited scalability of a single Linux instance.

The added scalability of Linux workloads is realized when you choose to exploit data-in-memory and sharing techniques of z/VM. Virtual disks in storage and minidisk cache can boost the performance of Linux servers on z/VM by making disk I/O operations execute at processor memory speeds, avoiding trips to the I/O subsystem, and waiting on spinning disks of storage. By sharing data and programming on shared, read-only minidisks, server deployment time can be reduced, as well as the maintenance effort required to replicate changes to data and programming among multiple distributed servers.

20.3 Installation methods and techniques

There are three basic installation methods for Linux on mainframes. These methods are:

- Installation as a native operating system. No other LPARs can be defined. All resources are dedicated to Linux. Linux cannot take advantage of virtualization techniques. This is not available on all IBM mainframes.

- ▶ Installation into a logical partition (LPAR). Linux in an LPAR is typically used when the single Linux server can process a large, critical workload. For example, SAP and IBM Lotus® Domino® servers are often run in an LPAR.
- ▶ Installation into a z/VM guest. Linux shares the LPAR's resources with other z/VM guests. Virtualization techniques are available to Linux.

20.3.1 Preparing the z/VM guest resources

As with any other hardware platform, any other operating system, and any other application, the basic resources are the processors, RAM, network, and disk devices required to complete the workload. Because an LPAR contains non-Linux and Linux guests, your planning should include all guests that will run in the LPAR.

Linux has been available on mainframes since the 2.2 kernel. During that time, best practices have been identified. For best practices for preparing z/VM guest resources, refer to:

<http://www.ibm.com/servers/eserver/zseries/library/whitepapers/pdf/gm130647.pdf>
<http://www.ibm.com/developerworks/linux/linux390/perf/>

20.3.2 Server farms or cloned environments

z/VM setup for cloned environments is covered in more detail in several other sources, such as the following IBM Redbooks and Redpaper:

- ▶ *z/VM and Linux on zSeries: From LPAR to Virtual Servers in Two Days*, SG24-6695
- ▶ *Linux on IBM @server zSeries and S/390: ISP/ASP Solutions*, SG24-6299
- ▶ *Linux on IBM @server zSeries and S/390: Cloning Linux Images in z/VM*, REDP-0301

We present a quick overview of the process in this section.

The first step in setting up the Linux cloning environment is to decide which directories or file systems will be split into read/write and read-only mount points. One suggested method is to have one Linux guest mount all its file systems in read/write mode so that updates can be applied. The remaining Linux guests, or clones will:

- ▶ Mount directories, such as /opt, /usr, /bin, /sbin, and /lib, as read-only file systems. These file systems usually contain system utilities and executables that end users should not need to modify.

- ▶ Mount directories, such as /home, /proc, /dev, /mnt, and /media, as read/write file systems. The root (/ , not to be mistaken for /root) directory should also be mounted in read/write mode.

This particular approach incurs additional work for the system administrator. However, the benefit of mounting parts of the operating system in the read-only mode means that mistakes will impact a much smaller area of the system. On any given clone, a user can only damage the configuration of that user's own applications.

Installation methods

The installation methods listed in Table 20-1 apply to both 31-bit and 64-bit versions of Red Hat and SUSE distributions. Both distributions recommend 512 MB RAM (minimum 256 MB) for the installation program.

Table 20-1 Installation techniques

Installation technique	Red Hat Enterprise Linux	SUSE Linux Enterprise Server
Network installation into z/VM guest using ISO images on remote server	NFS protocol is supported.	NFS protocol is supported.
Network installation into z/VM guest from remote directory	NFS, FTP, and HTTP protocols are supported.	NFS protocol is supported.
Installation using tape		Yes
LPAR installation using CD	Requires access to the Hardware Management Console (HMC) or Support Element Workplace™ (SEW). CD images must be accessed through FTP.	
Installation from S/390 hard drive	Disk must be formatted ext2 or ext3.	
Graphical installation	Yes	
Text mode installation	Yes	
SSH	Yes	
Telnet	Yes	
VNC	Yes	

Red Hat Enterprise Linux installation overview

Red Hat Enterprise Linux 4 Installation Guide for the IBM S/390 and IBM @server zSeries Architectures, [rhel-ig-s390-multi-en.pdf](#), describes the mainframe installation. We describe the high-level sequence of events in this section.

Perform the following steps:

1. First, you must log in to z/VM as the Linux guest account using a 3270 emulator program.

Enter the Conversational Monitor System (CMS) mode with the following command:

```
ipl cms
```

The Red Hat Enterprise Linux installation requires the following files:

- kernel.img (the Linux kernel)
- initrd.img (the initial RAM disk)
- redhat.parm
- redhat.conf

2. Update the redhat.parm and redhat.conf files.
3. Load (**punch**) the files into the z/VM guest's virtual reader queue and then boot (**ipl**) the guest using the files in the reader's queue. The installation program starts. You will be notified when you can login through **ssh**, **telnet**, or **vnc** to continue with the installation.

Sample Red Hat Enterprise Linux installation files

This section contains sample redhat.parm, redhat.conf, and rhel4_exec files used to initiate Red Hat installation. Example 20-1 presents a sample redhat.parm file, and Example 20-2 presents a sample redhat.conf file.

Example 20-1 Example installation file: redhat.parm

```
root=/dev/ram0 ro ip=off ramdisk_size=40000
CMSDASD=191 CMSCONFFILE=redhat.conf
vnc
```

Example 20-2 Example installation file: redhat.conf

```
HOSTNAME="Linux100.redbook.com"
DASD="200-202"
NETTYPE="qeth"
IPADDR="192.168.1.100"
SUBCHANNELS="0.0.0300,0.0.0301,0.0.0302"
PORTNAME="rhLinux"
NETWORK="192.168.1.0"
NETMASK="255.255.255.0"
BROADCAST="192.168.1.255"
SEARCHDNS="redbook.com"
GATEWAY="192.168.1.1"
DNS="192.168.1.254"
```

You can manually enter the commands to start the installation, but most system administrators find it easier to create a script with the necessary commands. The z/VM editor is invoked with the **xedit** command. The editor is closed and the file saved with the **save** subcommand. For example:

```
xedit rhel4 exec a
```

Example 20-3 shows the contents of the `rhel4 exec a` file in REXX format. To execute this file after you have saved it, type the **rhel4** command at the CMS prompt and the installation will begin.

Example 20-3 Example installation file: rhel4 exec a, in REXX format

```
/* */
'close rdr'
'purge rdr all'
'spool punch * rdr'
'punch kernel img a (noh'
'punch redhat parm a (noh'
'punch initrd img a (noh'
'change rdr all keep nohold'
'ipl 00c clear'
```

Notice that the `redhat conf a` (`redhat.conf`) file is not punched to the reader. You only need to punch the images and the `parm` file to begin the installation. The installation program will then read the contents of the configuration file when it is required.

From this point on, you should refer to `rhel-ig-s390-multi-en.pdf`, *Red Hat Enterprise Linux 4 Installation Guide for the IBM S/390 and IBM eServer zSeries Architectures*, for details about the specific type of installation (CD, ISO image, NFS) and access mode (SSH, Telnet, VNC) that you have chosen.

SUSE Linux Enterprise Server installation overview

SUSE Linux Enterprise Server Architecture-Specific Information, `sles-prep-zseries.pdf`, describes the mainframe installation. We describe a high-level sequence of events in this section.

Perform the following steps:

1. First, you must log in to z/VM as the Linux guest account using a 3270 emulator program.

Enter the Conversational Monitor System (CMS) mode with the following command:

```
ipl cms
```

SUSE Linux Enterprise Server installation requires the following files on the installation CD-ROM:

- vmrdr.ikr (the Linux kernel)
- initrd (the initial RAM disk)
- parmfile

These files are usually saved on the z/VM guest's minidisk with names such as:

- sles9.image
- sles9.initrd
- sles9.parm

2. Load (**punch**) the files into the z/VM guest's virtual reader queue and then boot (**ip1**) the guest using the files in the reader's queue. The installation program starts. You will be notified when you can login through **ssh**, **telnet**, or **vnc** to continue with the installation.

Sample SUSE Linux Enterprise Server installation files

This section contains sample sles9 parm a and sles9 exec a files used to initiate a SUSE installation. The sles9 parm file is limited to 10 lines. You place multiple parameters on a single line. Example 20-4 shows a sample of the sles9 parm a file.

Example 20-4 Example installation file: sles9 parm a

```
ramdisk_size=65536 root=/dev/ram1 ro init=/Linuxrc TERM=dumb
IP_HOST=Linux100.redbook.com IP_SEARCH=redbook.com IP_DNS=192.168.1.254
IP_ADDR=192.168.1.100 IP_GATEWAY=192.168.1.1
IP_INTERFACE=qeth IP_NETMASK=255.255.255.0 IP_BROADCAST=192.168.1.255
READ_DEVNO=0300 WRITE_DEVNO=0300 DATA_DEVNO=0302 PORTNAME=linport
```

You can manually enter the commands to start the installation, but most system administrators find it easier to create a script with the necessary commands. The z/VM editor is invoked with the **xedit** command. The editor is closed and the file saved with the **save** subcommand. For example:

```
xedit sles9 exec a
```

Example 20-5 on page 326 presents the contents of the sles9 exec a file in REXX format. To execute this file after you have saved it, type the **sles9** command at the CMS prompt and the installation will begin.

Example 20-5 Example installation file: *sles9 exec a*

```
/**/  
'close rdr'  
'purge rdr all'  
'spool punch * rdr'  
'punch sles9 image a (noh'  
'punch sles9 parm a (noh'  
'punch sles9 initrd a (noh'  
'change rdr all keep nohold'  
'ipl 00c clear'
```

From this point on, refer to *sles-prep-zseries.pdf*, *SUSE Linux Enterprise Server Architecture-Specific Information*, for details about the specific type of installation (CD, ISO image, NFS) and access mode (SSH, Telnet, VNC) you have chosen.

20.4 Booting, system initialization, and shutdown

This section describes the procedures for the system boot, initialization, and shutdown processes for a z/VM Linux guest. We do not discuss native and LPAR installations.

Booting your Linux guest

Typically, Linux is booted from a storage device defined to the Linux guest user virtual machine. To boot the Linux guest, issue the following CP command, where *vdev* is the virtual device number containing the Linux guest operating system:

```
ipl vdev clear
```

The Linux boot process displays the boot messages on the z/VM (3270) console.

To automate the process of IPLing a Linux guest during logon, include the following statement in the PROFILE EXEC A of the Linux guest user ID:

```
ipl vdev clear
```

Before including this statement in PROFILE EXEC, ensure that Linux has been successfully installed. Otherwise, you might get into a situation where z/VM will go into a continuous CP READ state.

The **clear** operand to the **ipl** command is optional. If specified, the content of the virtual machine in storage is set to binary zeros before the guest operating system is loaded.

ZIPL

The mainframe versions of Linux use the ZIPL bootloader. The configuration file is `/etc/zipl.conf`. If you make changes to `/etc/zipl.conf` you must run the `zipl` command to activate the changes.

For example, you would run the `zipl` command when directed to after applying kernel updates or after you added different boot parameters such as additional disk drives.

20.5 Device management

Management of devices in native and LPAR installations requires specialized knowledge of mainframe configuration and management. This section describes the facilities provided by z/VM to Linux guests.

The standard Linux commands to mount, unmount, format, and otherwise manage devices are available. The following section describes Linux reconfigurability, analogous to plug and play on the xSeries platform.

20.5.1 Linux reconfigurability

The channel device layer introduced with the Linux 2.4 kernel allows for new devices to be added to a Linux guest without an IPL. This means that you can use z/VM commands to add new devices, such as guest LAN NICs or real HiperSockets connections, and have these available to Linux without an IPL. In most cases, you can also use these new devices in TCP/IP immediately. Note that this applies only to devices attached through the channel subsystem. This does not apply to processors or storage.

3270 z/VM session

You can issue CP commands from the 3270 z/VM console by prefacing the command with the `#CP` control sequence. For example, to query the virtual machine minidisk assignments, issue the command:

```
#cp q v dasd
```

Telnet or SSH session

In order to issue CP commands from a Linux Telnet session, you need to install the `cpint` package. The `cpint` package is included in SUSE distributions. You can issue CP commands using the `hcp` command. For example, to display the status of virtual direct access storage devices, issue the command:

```
hcp q v dasd
```

There are two important items to note when using the **hcp** command:

- ▶ Do not enter the **hcp** command without any parameters; this will cause the virtual machine to enter the CP READ state. If this happens, you can log in using a 3270 emulator and enter the “**b**” command to return control of the guest to Linux.
- ▶ Remember to quote special characters (such as *) to prevent expansion by the bash shell.

You can also download the cpint package from the following site:

<http://www.linuxvm.org>

20.5.2 DASD hot-plug example

DASD hot-plug is available on SLES 9 and Red Hat Enterprise Linux AS V4. For this example, we logged in to a 64-bit SLES 9 Linux guest “linux041” as root:

1. First, we listed the z/VM virtual DASD and Linux DASD devices, as shown in Example 20-6.

Example 20-6 DASD device listing

```
linux041:~ # hcp q v dasd
DASD 0190 3390 51RRES R/O      107 CYL ON DASD  D08A SUBCHANNEL = 0008
DASD 0191 3390 VMRU01 R/W       2 CYL ON DASD  D061 SUBCHANNEL = 0002
DASD 019D 3390 51RW01 R/O     146 CYL ON DASD  D08B SUBCHANNEL = 0009
DASD 019E 3390 51RW01 R/O     250 CYL ON DASD  D08B SUBCHANNEL = 000A
DASD 0200 3390 LICOFO R/W     3338 CYL ON DASD  D0F0 SUBCHANNEL = 000C
DASD 0201 3390 LICOEO R/W     3338 CYL ON DASD  D0E0 SUBCHANNEL = 000D
DASD 0202 3390 LICOEB R/W     3338 CYL ON DASD  D0EB SUBCHANNEL = 000E
DASD 0203 3390 LICOEC R/W     3338 CYL ON DASD  D0EC SUBCHANNEL = 000F
DASD 0204 3390 LICOED R/W     3338 CYL ON DASD  D0ED SUBCHANNEL = 0010

linux041:~ # cat /proc/dasd/devices
0.0.0200(ECKD) at ( 94:  0) is dasda      : active at blocksize: 4096,
600840 blocks, 2347 MB
0.0.0201(ECKD) at ( 94:  4) is dasdb      : active at blocksize: 4096,
600840 blocks, 2347 MB
0.0.0202(ECKD) at ( 94:  8) is dasdc      : active at blocksize: 4096,
600840 blocks, 2347 MB
0.0.0203(ECKD) at ( 94: 12) is dasdd      : active at blocksize: 4096,
600840 blocks, 2347 MB
0.0.0204(ECKD) at ( 94: 16) is dasde      : active at blocksize: 4096,
600840 blocks, 2347 MB
```

2. We linked to an additional disk with the following command:

```
linux041:~ # hcp link licdasd d158 0500 mr
```

These messages appeared in `/var/log/messages`:

```
Jan 10 13:04:21 linux041 kernel: crw_info : CRW reports slct=0, oflw=0,
chn=0, rsc=3, anc=1, erc=4, rsid=3
Jan 10 13:04:21 linux041 /etc/hotplug/ccw.agent[1926]: Configuring channel
0.0.0500 for event 'ccw'
```

3. The YaST2 GUI provides a graphical method for activating devices. We chose to use a command line utility to activate DASD 0500:

```
linux041:~ # dasd_configure 0.0.0500 1 0
Configuring device 0.0.0500
Setting device online
```

These messages appeared in `/var/log/messages`:

```
Jan 10 13:05:41 linux041 kernel: dasd(eckd): 0.0.0500: 3390/0A(CU:3990/04)
Cyl:3338 Head:15 Sec:224
Jan 10 13:05:41 linux041 kernel: dasd(eckd): 0.0.0500: (4kB blks):
2403360kB at 48kB/trk linux disk layout
Jan 10 13:05:41 linux041 kernel: dasdf:CMS1/ LIC158: dasdf1
Jan 10 13:05:42 linux041 /etc/hotplug/block.agent[2305]: new block device
/block/dasdf
Jan 10 13:05:42 linux041 /etc/hotplug/block.agent[2307]: new block device
/block/dasdf/dasdf1
```

4. We listed the z/VM virtual DASD and Linux DASD devices again and found these additional entries:

```
linux041:~ # hcp q v dasd
...
DASD 0500 3390 LIC158 R/W          3338 CYL ON DASD  D158 SUBCHANNEL = 0003
```

```
linux041:~ # cat /proc/dasd/devices
...
0.0.0500(ECKD) at ( 94: 20) is dasdf : active at blocksize: 4096, 600840
blocks, 2347 MB
```

At this point, we can access the device as `/dev/dasdf`. Any DASD operations, such as partitioning and formatting, are now available for use with this drive.

5. We wanted our new DASD device to be found during subsequent boots or IPLs, so we performed addition (optional) operations, as shown in Example 20-7, using `mkinitrd` and `zipl`.

Example 20-7 Using `mkinitrd` and `zipl` to make new DASD devices persistent

```
linux041:~ # mkinitrd
Root device: /dev/system/sysvol (mounted on / as ext3)
Module list: jbd ext3 dm_mod dm-mod dm-snapshot dasd_eckd_mod

Kernel image: /boot/image-2.6.5-7.191-s390x
Initrd image: /boot/initrd-2.6.5-7.191-s390x
```

```

Shared libs:  lib64/ld-2.3.3.so lib64/libacl.so.1.1.0 lib64/libattr.so.1.1.0
lib64/libblkid.so.1.0 lib64/libc.so.6 lib64/libdevmapper.so.1.01
lib64/libdl.so.2 lib64/libpthread.so.0 lib64/librt.so.1 lib64/libselinux.so.1
lib64/libuuid.so.1.2
Modules:      kernel/fs/jbd/jbd.ko kernel/fs/ext3/ext3.ko
kernel/drivers/md/dm-mod.ko kernel/drivers/md/dm-snapshot.ko
kernel/drivers/s390/block/dasd_mod.ko
kernel/drivers/s390/block/dasd_eckd_mod.ko
DASDs:       0.0.0200(ECKD) 0.0.0201(ECKD) 0.0.0202(ECKD) 0.0.0203(ECKD)
0.0.0204(ECKD) 0.0.0500(ECKD)
Including:     udev dm/lvm2

```

initrd updated, zipl needs to update the IPL record before IPL!

```

linux041:~ # zipl -V
Using config file '/etc/zipl.conf'
Target device information
  Device.....: 5e:00
  Partition.....: 5e:01
  Device name.....: dasda
  DASD device number.....: 0200
  Type.....: disk partition
  Disk layout.....: ECKD/compatible disk layout
  Geometry - heads.....: 15
  Geometry - sectors.....: 12
  Geometry - cylinders.....: 3338
  Geometry - start.....: 24
  File system block size.....: 4096
  Physical block size.....: 4096
  Device size in physical blocks...: 22476
Building bootmap in '/boot/zipl'
Adding IPL section 'ipl' (default)
  kernel image.....: /boot/image at 0x10000
  kernel parmline...: 'root=/dev/system/sysvol selinux=0 TERM=dumb
elevator=cfq' at 0x1000
  initial ramdisk...: /boot/initrd at 0x800000
Preparing boot device: dasda (0200).
Syncing disks...
Done.

```

6. After rebooting the Linux guest, our new DASD device became available at /dev/dasdf.

20.6 Performance monitoring and tuning

The most critical phase of performance tuning is the design phase. The basic considerations are allocating the CPU, RAM, network, and DASD devices your applications and operating system will require. Allocate a RAM equivalent to your peak requirement to reduce swapping within the Linux guest.

Review the documented best practices and conduct a Solution Assurance Review prior to beginning any proof of concept or proof of technology effort.

The basic considerations are allocating and configuring the processor, memory, network, and DASD devices that your applications and operating system will require. For example, allocate enough memory to a Linux guest so that it does not swap.

20.6.1 Performance monitoring

Performance monitoring tools are available for both z/VM and Linux. Many third-party tools are available.

z/VM monitoring tools

The *Performance Toolkit for VM* enables you to monitor the resource utilization of your Linux guests, including CPU, memory, and I/O.

The Distributed Data Server (DDS) acts to feed Linux monitor data to the Data Collector. DDS is distributed as the server component of the RMF™ PM for Linux. If you install this package, the Performance Toolkit will be able to monitor information about a Linux image, including:

- ▶ Linux system details
 - Processes created per second
 - Context switches per second
 - CPU utilization, both user and kernel
 - Memory utilization
 - Network activity
 - File system usage
- ▶ Apache performance data

You can download the RMF PM for Linux utility from:

<http://www.ibm.com/servers/eserver/zseries/zos/rmf/rmfhtmls/pmweb/pm1in.html>

What's new in Performance Toolkit for VM in Version 5.1.0, gm130637.pdf, provides a high-level overview of setting up and using the Performance Toolkit for VM and RMF PM.

FCON/ESA support for Linux on z/VM

With release 3.2.03 of FCON/ESA, a Linux performance data interface has been added. Data retrieval and display of Linux internal performance data is based on the Linux Distributed Data Server (DDS) interface, originally written for use with Resource Measurement Facility (RMF) Performance Monitoring (PM). The DDS interface must be installed and active on all Linux systems that are to be monitored. Performance data retrieval is based on Extensible Markup Language (XML) requests, sent to the Linux systems via TCP/IP. Only the data actually needed for building a specific Linux performance report is retrieved. Performance data is collected only in the Linux systems, and performance history data is only available from the file systems of the Linux systems. FCON/ESA does not collect or save any Linux internal performance data.

Linux monitoring tools

Linux provides the same monitoring commands that are available on other hardware platforms. For example, the **sar**, **vmstat**, **iostat**, **mpstat**, **top**, and **free** commands are available. These commands provide a view of how a single Linux guest views its performance or behavior. These commands do not incorporate any information about any other guests in an LPAR or any z/VM resource utilization.

It is difficult to determine actual or real resource utilization by a Linux guest since mainframe resources are shared and virtualized. There are many advanced z/VM and Linux monitoring tools which focus on either z/VM or Linux, but no native z/VM or Linux tools that monitor resources for both points of view.

Consider, for example, a Linux guest that shows high CPU utilization with the **top** or **sar** command. This guest might actually get only a few CPU-cycles from z/VM. *Accounting and monitoring for z/VM Linux guest machines, redp3818.pdf*, describes one way of integrating output from the z/VM CP monitor facility and output from the Linux **sar** command to identify more realistically the CPU utilization of the Linux guest.

More advanced performance monitoring tools are described in the IBM Redbook *Linux on IBM @server zSeries and S/390: Performance Measurement and Tuning, SG24-6926*.

20.6.2 Performance tuning

System administrators will find this reference material extremely useful as they begin to tune mainframe distributions of Linux:

- ▶ Linux on zSeries: Tuning hints and tips
<http://www.ibm.com/developerworks/linux/linux390/perf/index.html>
- ▶ Linux Performance when running under VM
<http://www.vm.ibm.com/perf/tips/linuxper.html>
- ▶ Linux on IBM System z library
<http://www.ibm.com/servers/eserver/zseries/os/linux/library/index.html>
- ▶ *Linux on IBM @server zSeries and S/390: Performance Measurement and Tuning*, SG24-6926
- ▶ *Implementing Linux on IBM @server zSeries and S/390: Best Practices*
<http://www.ibm.com/servers/eserver/zseries/library/whitepapers/pdf/gm130647.pdf>

We describe high-level information from this reference material in this section.

z/VM performance tuning

z/VM offers many tuning parameters for allocating resources to virtual machines. Only a few of these parameters increase the amount of resources available to a guest. The rest of the parameters take away resources from one virtual machine and reallocate them to another one depending on its needs.

Tuning does not increase the total amount of system resources. Resources are defined at the LPAR level. Tuning only allows defined resources to be used more effectively. Therefore, if your workload exceeds the capacity of your defined resources, you will not be able to achieve the performance you desire.

This list contains some of the z/VM items that can be tuned to increase Linux guest performance:

- ▶ Tune memory requirements:
 - Reduce storage for idle servers.
 - Use a cloned server farm to share the kernel.
- ▶ Tune the Linux swap device:
 - Disable minidisk caching or use a VDISK.
 - Use multiple swap devices with different priorities.
- ▶ Tune scheduling:
 - Assign LPAR weights.

- Configure the CP scheduler.
- ▶ Tune DASD:
 - Use more, smaller disks and appropriate stripe sizes.
 - Reduce channel subsystem contention.
 - Configure minidisk caching (for some workloads).
- ▶ Tune network:
 - Virtual LAN or direct attachment to OSA adapter.

Linux performance tuning

As with z/VM, you can achieve optimum performance only if the necessary resources are defined to the Linux guest. This list contains important Linux components for tuning on the mainframe:

- ▶ Tune memory requirements and processor performance:
 - Apply the timer patch if it is not already incorporated into the kernel.
 - Do not run unnecessary cron jobs.
 - Do not run unnecessary processes.
- ▶ Tune the Linux swap device:
 - Use multiple, smaller swap devices.
- ▶ Tune the file systems:
 - Use the appropriate file system (ext2, ext3, reiserfs) for your application.
 - Use LVM or software RAID to spread I/O across multiple drives.

The preceding list is not comprehensive. For example, Linux kernel parameters can be tuned and process priorities can be set with the **nice** and **renice** commands.

20.7 Troubleshooting and diagnostics

Third-party software is available for both z/VM and Linux troubleshooting and diagnostics. This section describes standard z/VM and Linux troubleshooting and diagnostic tools and utilities.

20.7.1 z/VM troubleshooting and diagnostics

IBM introduced the Performance Toolkit for VM in the announcement of z/VM V4.4. The Performance Toolkit for VM provides enhanced capabilities for monitoring and reporting performance data. New function provided with z/VM

V5.1 includes high-level Linux reports based on application monitor records from Linux guests. The Performance Toolkit for VM processes Linux performance data obtained from the Resource Management Facility (RMF) Linux performance gatherer, `rmfpms`. Performance monitor data can be viewed from Web browsers or PC-based 3270 emulator graphics.

For information concerning the `rmfpms` application and the Performance Toolkit for VM, see:

<http://www.ibm.com/servers/eserver/zseries/zos/rmf/rmfhtmls/pmweb/pmlin.html>

<http://www.vm.ibm.com/related/perfkit/>

20.7.2 Linux troubleshooting and diagnostics

Red Hat Enterprise Linux provides the `sysreport` package and command. SUSE Linux Enterprise Server provides the `sig` package and command and the `dbginfo.sh` command. These three commands can be used to gather system information that might be needed for troubleshooting a problem.

Standard Linux troubleshooting and diagnostics techniques also work on the mainframe. Here are some examples:

- ▶ Set the log level for `syslog` and other applications.
- ▶ Standard network techniques such as `ping`, `netstat`, `ifconfig`, `route`, and `traceroute` are available.
- ▶ Development debugging tools such as `gdb` (GNU debugger), `ddd` (Data Display Debugger), `nana`, `gprof` (GNU profiler), and the `MALLOC_CHECK_` environment variable are available.
- ▶ Commands such as `strace`, `ltrace`, `ps`, `top`, `free`, `iostat`, and `vmstat` are available.

You can use the bash parameters `-v` and `-x` to debug scripts.

20.8 S/390 and zSeries-specific Linux commands

Mainframe distributions of Linux provide the commands common to all hardware platforms. These distributions also provide a unique set of commands that allow Linux to interact with the host operating system and virtualized devices.

20.8.1 Packages specifically for the mainframe

Red Hat Enterprise Linux and SUSE Linux Enterprise Server each provide packages with tools and utilities specifically designed to work with the mainframe. The packages are:

- ▶ Red Hat:
 - s390-utils
- ▶ SUSE:
 - cpint
 - s390-tools
 - s390
 - yast2-s390 (graphical interface for tools)

20.8.2 Commands specifically for the mainframe

These commands are provided by the packages listed in the previous section.

The cpint package, while not included in Red Hat Enterprise Linux, can be downloaded from:

<http://www.linuxvm.org>

If the cpint package is installed, the **#cp** and **hcp** commands are available. Table 20-2 does not list the commands because the cpint package is not included in the default distribution.

Table 20-2 S/390 and zSeries-specific commands

Task	RHEL	SLES
Send CP commands to the controlling VM from a 3270 session. The command prefix depends on your 3270 emulator settings.		#cp \$cp
Modify generic attributes of channel attached devices.	chccwdev	chccwdev
Read files from CMS volumes.	cmsfscat	
Check a CMS formatted volume.	cmsfsck	
Copy files from CMS volumes to Linux on S/390 file systems.	cmsfscp	
List files on CMS formatted volumes.	cmsfs1st	
Report CMS formatted volume status.	cmsfsvol	

Task	RHEL	SLES
Format disk drive.	dasdfmt	dasdfmt
Display DASD and VTOC information.	dasdview	dasdview
Create output similar to sga and sysreport (which are also available).		dbginfo.sh
Write a partition table to a DASD in the form of a VTOC (volume table of contents).	fdasd	fdasd
Send CP commands to the controlling VM from a Telnet session.		hcp
List channel subsystem devices.	lscss	lscss
List channel attached direct access storage devices (DASD).	lsdasd	lsdasd
List all qeth-based network devices with their corresponding settings.	ipconfig	lsqeth
List channel-attached tape devices.	lstape	lstape
Prepare a device for use as S/390 dump device.		mkdump
Get MAC and IP addresses from OSA and HiperSockets. Flush the ARP table.	qetharp	qetharp
Configure IBM QETH function IPA, VIPA, and Proxy ARP.	qethconf	qethconf
Create a 32-bit S/390 environment on S/390x.		s390
Create a 64-bit S/390 environment on S/390x.		s390x
Display messages on a zSeries tape device.	tape390_display	tape390_display
Adjust tunable parameters on DASD devices.	tunedasd	tunedasd
Convert VMDUMPs into lkcd dumps.		vmconvert
Copy dumps.	zgetdump	zgetdump
Boot loader for S/390 and zSeries architectures.	zipl	zipl

20.9 High availability

High availability (HA) techniques are available at the mainframe, z/VM, and Linux levels. All are possible, and some can be used in conjunction with others, but no single high availability solution currently exists that covers both Linux and z/VM requirements. Only Linux high availability products are available today.

The basic rule of implementing a highly available environment is to identify and remove single points of failure. A general rule is to use redundant components (hardware and software). HA scenarios for mainframes and Linux on mainframes are described in various IBM Redbooks listed in 20.11, “References” on page 342. We describe some of the built-in HA features of the IBM mainframe and present high-level overviews of HA scenarios in this section.

20.9.1 Hardware HA

The mean time between failures (MTBF) on mainframe hardware is now measured in decades, and all mainframes are shipped with at least one “hot spare” processor that can be instantaneously substituted for a failing CPU. The mainframe hardware is able to detect CPU errors and transparently switch to another processor for continuous operation (CPU sparing). This function is transparent to the operating system. The zSeries memory is also self-checking and self-repairing, as is the I/O subsystem.

Practically speaking, in the vast majority of cases, the only way a “hardware failure” can occur that will disrupt the entire environment is if the power supply to the mainframe CEC is somehow interrupted, or if all of the physical cable paths (I/O, network, or both) leading into the Linux LPAR or guest were cut.

An additional hardware exposure to the mainframe environment occurs when a disruptive maintenance microcode patch has to be applied. In this case, the entire machine must be brought down. This type of maintenance is relatively rare, but does occur.

In the event of a failure such as those just described, you need a backup mainframe with a similar configuration to provide HA.

20.9.2 z/VM HA

z/VM is designed using address spaces, memory protection, and fault-tolerant code. In an HA environment, z/VM can be configured to run in another LPAR:

- ▶ As a hot-standby, IPLed with backup guests but no work being processed.
- ▶ Cold-standby, defined but not IPLed.

- ▶ In a parallel processing configuration for high availability.
- ▶ IPLed with some guests active and processing work, and with some non-IPLed guests defined to serve as a “warm standby” for other operational z/VM systems.

Error recovery

The zSeries microcode and z/VM try to recover most errors, including such things as intermittent and permanent machine errors and system I/O errors, without manual intervention. When an error occurs, CP records it and sends a message to your z/VM primary system console. If the error condition does not damage system integrity and can be assigned to a single guest, z/VM makes a soft abend of the guest and operation continues. This means that in almost all cases a crash of a guest system does not affect CP operation, and other guests can operate without disruption.

Network HA

A critical component that might be overlooked when designing connectivity in a large-scale Linux installation is the z/VM TCP/IP stack itself. There can be many times when, as a result of misconfiguration or failure in a Linux system, you cannot get access through the network to a Linux guest. On these occasions, being able to obtain a connection through z/VM TCP/IP to the Linux console lets us do enough to get network connectivity again, log on, and fix the problem. Being able to get reliable access to the z/VM TCP/IP stack is, therefore, very important.

The z/VM TCP/IP stack supports the VIPA function. We suggest that you look at using multiple interfaces to the z/VM stack, and use VIPA to address z/VM.

The IBM Redbook *Linux on IBM @server zSeries and S/390: ISP/ASP Solutions*, SG24-6299, discusses a method of using multiple attachments in combination with a virtual IP address configured on a dummy interface. This still can form the basis of a high availability connectivity solution. As an alternative to using a dummy interface, you can configure additional addresses against the IP loopback interface (lo). This means you do not need to have another module installed.

The way that the Linux kernel processes incoming IP traffic does not require the virtual IP address to be associated with a dummy or loopback interface. Your virtual address can be just as easily defined against one of your real interfaces.

20.9.3 Automating Linux guest boot and shutdown

For regular z/VM system start (IPL), or after a z/VM termination, an automated restart of z/VM can be configured. During system initialization, CP automatically

logs on (starts) user AUTOLOG1. AUTOLOG1 can automatically start other virtual machines when issuing the AUTOLOG command in the PROFILE EXEC of AUTOLOG1. If a certain start order is necessary due to application dependencies, you can implement a startup procedure for the guest systems. This startup procedure is called within the PROFILE EXEC of user AUTOLOG1. The procedure is written in the script language REXX.

20.9.4 Linux-HA

Linux high availability functions specific to the zSeries platform include the Linux Virtual Server and the Heartbeat option.

Linux Virtual Server

The Linux Virtual Server (LVS) concept ties a group of server machines together using various network and scheduling techniques so that they present a single server image to clients requesting service. A single load balancing and scheduling machine initially receives all the traffic, which it then sends to back-end servers. A sample implementation of LVS is described in the IBM Redbook *Linux on IBM @server zSeries and S/390: Distributions*, SG24-6264.

Heartbeat and network option

TCP/IP Heartbeat techniques can be used for z/VM Linux guests in same LPAR, in different LPARs in the same mainframe, or in different mainframes.

20.9.5 Backup and recovery options

There are three basic options for backing up and restoring Linux data on mainframes:

- ▶ The first option is to use the z/VM DASD Dump/Restore (DDR) Facility to back up and restore Linux data. This is generally the most common strategy that customers use for Linux and z/VM backups. DDR only provides volume-level backups, not file-level backups. That is, you can back up and restore entire minidisks, but not individual files. To restore individual files in this scenario, customers typically restore the minidisk that the file in question was on, and then mount that volume to the Linux guest and retrieve the lost file.
- ▶ The next basic option for Linux backups is to modify an existing z/OS® backup strategy to include Linux volumes. The benefit of using existing z/OS backup facilities (most notably DFSMSdss™) is that Linux DASD volumes can generally be treated like any other volume that is dumped and restored using dss. Although z/OS cannot access the data in the volume itself, the volume can be dumped and restored just like any other DASD volume.

- ▶ The backup strategy that is least used currently, but is growing the fastest, is to use tools that run on Linux to back up Linux volumes and files. The most attractive argument for this option is that these backup tools provide easy access to both file-level and volume-level backups.

20.10 Security

Linux running in native mode in an LPAR can only rely on the security measures provided by the Linux distribution. Linux running as a z/VM guest, however, has access to several additional security features provided by the mainframe hardware and z/VM or z/OS:

- ▶ Consider HiperSockets, guest LANs, and virtual switches. These networking options are implemented either in memory or directly through the host system's network adapters. Network communication between operating systems running on the mainframe never need to leave the mainframe. That means there is nothing on the physical network to be “sniffed” or captured by other programs.
- ▶ The security provided by LPAR isolation is certified by Evaluation Assurance Levels (EAL) 5 Common Criteria.
- ▶ Linux can access zSeries specialized cryptographic hardware.
- ▶ Each z/VM guest has its own user ID and password. Each guests runs in its own virtual address space and cannot access another guest's memory.
- ▶ Linux guests can use z/OS LDAP and RACF® facilities for authentication.

The following publications describe the concepts and techniques for implementing these interfaces between Linux, z/VM, and zSeries hardware:

- ▶ *Linux on IBM zSeries and S/390: Securing Linux for zSeries with a Central z/OS LDAP Server (RACF)*, REDP-0221
- ▶ *Linux on IBM @server zSeries and S/390: Best Security Practices*, SG24-7023

For documentation concerning z/VM security and integrity resources such as the z/VM Common Criteria, refer to:

<http://www.vm.ibm.com/security>

This Web site also provides links to more information about zSeries, z/VM, and Linux security features.

The standard Linux system administration facilities such as PAM, OpenSSL, OpenSSH, and OpenLDAP are available, too. Linux guests can secure communications using **scp**, **sftp**, and **iptables**. User authentication can be

through /etc/passwd, Kerberos and sudo. Linux security is Linux security, even on the mainframe.

20.11 References

For more information, refer to the following publications:

- ▶ *Linux on IBM zSeries and S/390: Server Consolidation with Linux for zSeries*, REDP-0222
- ▶ *z/VM and Linux on zSeries: From LPAR to Virtual Servers in Two Days*, SG24-6695
- ▶ *Linux for S/390*, SG24-4987
- ▶ *Linux on IBM @server zSeries and S/390: Large Scale Linux Deployment*, SG24-6824
- ▶ *Linux on IBM zSeries and S/390: High Availability for z/VM and Linux*, REDP-0220
- ▶ *Linux on IBM @server zSeries and S/390: Building SuSE SLES8 Systems under z/VM*, REDP-3687
- ▶ *Printing with Linux on zSeries Using CUPS and Samba*, REDP-3864
- ▶ *Linux with zSeries and ESS: Essentials*, SG24-7025
- ▶ *IBM System z9 and @server zSeries Connectivity Handbook*, SG24-5444
- ▶ *Linux on IBM @server zSeries and S/390: Best Security Practices*, SG24-7023
- ▶ *Accounting and Monitoring for z/VM Linux guest machines*, REDP-3818
- ▶ *Linux on IBM @server zSeries and S/390: Performance Measurement and Tuning*, SG24-6926
- ▶ *Linux on IBM zSeries and S/390: Securing Linux for zSeries with a Central z/OS LDAP Server (RACF)*, REDP-0221
- ▶ *Linux on IBM @server zSeries and S/390: Best Security Practices*, SG24-7023
- ▶ *Linux for IBM System z9 and zSeries*, SG24-6694
- ▶ *Networking Overview for Linux on zSeries*, REDP-3901
- ▶ *IBM @server zSeries 900 Technical Guide*, SG24-5975
- ▶ *IBM @server zSeries 890 Technical Introduction*, SG24-6310
- ▶ *Technical Introduction: IBM @server zSeries 800*, SG24-6515
- ▶ *IBM @server zSeries 990 Technical Guide*, SG24-6947
- ▶ *IBM System z9 109 Technical Guide*, SG24-7124
- ▶ *What's new in Performance Toolkit for VM in Version 5.1.0*
<http://www.vm.ibm.com/library/gm130637.pdf>

Appendixes

Appendix A, “Tasks reference” on page 345 provides quick reference tables that present methods for accomplishing equivalent tasks between Solaris and Linux.

Appendix B, “Commands and configuration files reference” on page 365 provides quick reference tables for the comparison of configuration files, comparable commands, and common commands.

Appendix C, “UNIX to Linux Porting: A Comprehensive Reference (table of contents and sample chapter)” on page 371 provides sample content from a related publication: Mendoza, et al., *UNIX to Linux Porting: A Comprehensive Reference*, Prentice Hall, 2006, ISBN 0131871099.

Appendix D, “Example: System information gathering script” on page 397 provides a description of the `getsysinfo` sample diagnostic scripts, used to gather detailed system configuration information.

Appendix E, “Additional material” on page 401 provides information about how to access the additional material provided with this IBM Redbook, available on the Internet.



A

Tasks reference

This appendix contrasts common tasks between the Solaris and Linux operating systems. Tasks are grouped according to the following major categories. Each major category is contained within a table. Tables can also include file location information or pertinent information that is related to the category they contain.

This reference provides information about Linux and Solaris in the following categories:

- ▶ Packaging
- ▶ Installing and upgrading tasks
- ▶ Booting and shutting down
- ▶ User management tasks
- ▶ Device management and configuration
- ▶ Network management and configuration
- ▶ Managing system resources
- ▶ Printer management and configuration
- ▶ Disk and file system management
- ▶ Logical volume management
- ▶ Network troubleshooting

Packaging

Table A-1 provides information that shows differences in Linux and Solaris packaging-related tasks.

Table A-1 Packaging comparison

Units	Solaris	Linux
Smallest installable unit.	Package Patch	Package
Single installable image; must be distributed and installed as a unit.	Package	Package
Logical grouping of packages.	Software cluster Patch cluster	Package
Logical grouping of packages and software clusters.	Software configuration clusters, for example: <ul style="list-style-type: none"> ▶ Core: Required operating system files ▶ End-User System Support: Core plus window environment ▶ Developer System Support: End-User plus the development environment ▶ Entire Distribution: Developer System plus enhanced features ▶ Entire Distribution Plus OEM: Entire Distribution plus third-party hardware drivers (on SPARC only) 	<ul style="list-style-type: none"> ▶ Workstation install (stand-alone client) ▶ Developer install (above, including developer tools, compiler) ▶ Server install (server services and software) ▶ Full Install (all software/services) ▶ Custom (choose packages manually)

Installing and upgrading tasks

Table A-2 provides information that shows differences in Linux and Solaris installation and upgrade tasks.

Table A-2 Installing and upgrading tasks

Task	Solaris	Red Hat	SUSE
Install packages	pkgadd		rpm -i

Task	Solaris	Red Hat	SUSE
Display installed packages	pkginfo or pkgparam	rpm -qa	
Remove software package	pkgrm	rpm -e	
Upgrade/install a package	pkgadd	rpm -U	
Verify correct installation	pkgchk	rpm -V	
Install a patch	patchadd	rpm -F (upgrades only if already present)	
List content of installed package	Look in <code>/var/sadm/install/contents</code>	rpm -ql	
Check which file belongs a package	Look in <code>/var/sadm/install/contents</code>	rpm -qf	
Check package information	Look in <code>/var/sadm/pkg/PKGNAME/pkginfo</code>	rpm -qi	
Remove a patch	patchrm	N/A	
Display installed patches	showrev -p	N/A	
Install OS on another disk (alternate disk installation)	Live Upgrade	Install different OS on different disk/partition, dual boot using GRUB	
Create an installation server for network installation	setup_install_server <i>install_dir_path</i>	anaconda (see http://www.tldp.org/HOWTO/Network-Install-HOWTO.html)	autoyast (see http://www.tldp.org/HOWTO/Network-Install-HOWTO.html)
Create a boot server for network installation	setup_install_server -b <i>bootdirpath</i>	grub (see http://www.tldp.org/HOWTO/Clone-HOWTO/)	grub (see http://www.tldp.org/HOWTO/Clone-HOWTO/)
Set up a client for network installation	add_install_client	anaconda (see http://www.tldp.org/HOWTO/Network-Install-HOWTO.html)	autoyast (see http://www.tldp.org/HOWTO/Network-Install-HOWTO.html)

Booting and shutting down

Table A-3 provides information that shows differences in the process and locations of items that are involved in booting and shutting down a system in Linux and Solaris.

Table A-3 Boot and shutdown tasks

Task/location	Solaris	Linux
Boot process	<p>Phases:</p> <ol style="list-style-type: none"> 1. OpenPROM: Display system information, run POST, load bootblk, and locate ufsboot. 2. Boot programs: bootblk loads and executes the ufsboot. 3. Kernel initialization: ufsboot loads and executes the core kernel, initializes core kernel data structures, loads other kernel modules based on the <code>/etc/system</code> file, and starts <code>/sbin/init</code> program. 4. <code>init</code>: Starts other processes based on the <code>/etc/inittab</code> file. 5. <code>/sbin/rcX</code> runs the corresponding <code>/etc/rcX.d</code> scripts to start up other components. 	<p>Phases:</p> <ol style="list-style-type: none"> 1. BIOS: Checks the system and peripheral devices. Locates and runs the Master Boot Record (MBR). 2. MBR loads GRUB (GRand Unified Bootloader). 3. GRUB boots the kernel using information in <code>/boot/grub/menu.lst</code>. 4. Kernel starts initializing devices and loads additional device drivers from <code>initrd</code> image, and then starts <code>init</code> process from <code>/sbin/init</code>. 5. <code>init</code>: Starts other processes based on the <code>/etc/inittab</code> file. 6. <code>/etc/rc.d/rc</code> in RHEL or <code>/etc/init.d/rc</code> in SLES runs the corresponding <code>/etc/rcX.d</code> scripts to start up other components.
Default kernel location	<ul style="list-style-type: none"> ▶ <code>/kernel</code> - Contains common components. ▶ <code>/platform/'platform-name'/kernel</code> - Contains the platform-specific kernel components. ▶ <code>/platform/'hardware-class-name'/kernel</code> - Contains kernel components specific to this hardware class. ▶ <code>/usr/kernel</code> - Contains kernel components common to all platform within a particular CPU set. 	<p><code>/boot</code> partition or directory, contains the kernels and <code>initrd</code> images.</p>

Task/location	Solaris	Linux
Default kernel modules location	<p>In each of the subdirectories mentioned in the preceding cell, they might contain the following subdirectories:</p> <ul style="list-style-type: none"> ▶ drv - Contains loadable device drivers. ▶ exec - Contains modules that run programs stored in different file formats. ▶ fs - Contains file system modules. ▶ misc - Contains miscellaneous system-related modules. ▶ sched - Contains operating system schedulers. ▶ strmod - Contains loadable System V STREAMS modules. ▶ sys - Contains loadable system calls. ▶ cpu - Contains CPU type-specific modules. ▶ tod - Contains time-of-day hardware modules. ▶ sparcv9 - Contains 64-bit versions of specific modules. 	<p>Use the output of the following command: <code>/lib/modules/`uname -r`/</code></p>
System run levels		
Run level S or s	Single user level (boot -s). Only some file systems are mounted as opposed to run level 1.	Minimal single user mode on SUSE. In Red Hat, it is same as initlevel 1.
Run level 0	Shuts down the operating system. Does not attempt to turn off the power. Returns to OpenPROM ok prompt.	Power-down state, shuts down the operating system and tries to turn power off if supported by the system. Equivalent to Solaris run level 5.
Run level 1	Administrative single user mode. Can access all available file systems. User logins disabled.	Single user mode.
Run level 2	Multiuser without NFS resources shared.	Multiuser without NFS resources shared. Text only login.

Task/location	Solaris	Linux
Run level 3	Multiuser with NFS resources shared (default run level for Solaris). The login is text or graphical depending on the console capabilities.	Multiuser with NFS resources shared. Text only login.
Run level 4	N/A (alternate multiuser).	N/A (alternate multiuser).
Run level 5	Power-down state, shuts down the operating system and tries to turn power off if supported by the system.	Full multiuser, NFS resources shared, and graphic login with X display manager.
Run level 6	Reboot.	Reboot. Same as in Solaris.
Emergency run level	N/A	Ignore the /etc/inittab file and launch the default shell.
Determine a system's run level	who -r	who -r runlevel
Change a system's run level	telinit run-level-number init run-level-number	telinit run-level-number init run-level-number
Startup script	/sbin/rc run-level-number	In RHEL: /etc/rc.d/rc run-level-number In SLES: /etc/init.d/rc run-level-number
Use new kernel	N/A	grub
Display boot loader information	N/A	cat /boot/grub/menu.lst
Display or alter the list of boot devices	boot	BIOS
Boot methods		
Default boot mode	boot or if auto-boot is set to true, it is automatic. Note: The commands listed in this column is assumed that you are at the OpenPROM prompt.	Automatic or press Enter at grub prompt. Note: The commands listed in this column assume that you are at the GRUB prompt.
Single user mode	boot -s	Add -s , single , S , or 1 at the end of boot line.
Reconfiguration mode	boot -r	Not needed.

Task/location	Solaris	Linux
Verbose boot mode	boot -v	In RHEL, remove rhgb quiet from the boot command line; in SLES, press Esc when booting.
Interactive boot mode	boot -a	Edit grub boot line to specify other kernel and options.
Recovery boot mode	boot cdrom boot net	Boot from a rescue CD-ROM or from network.
Interactive start of services while booting	N/A	Possible but vendor dependent.
Emergency boot mode	N/A	Add emergency or -b option at the end of boot line.
Shutdown and reboot	reboot shutdown -i 6	reboot shutdown -r now
Shutdown	halt poweroff shutdown	halt poweroff shutdown

User management tasks

Table A-4 provides information that shows differences in Linux and Solaris user management-related tasks.

Table A-4 User management tasks

Task/locations	Solaris	Linux
Run multiple tasks in a GUI environment	admintool	In RHEL: system-config-users In SLES: yast2
Add a user	useradd	useradd
Remove a user	userdel	userdel
Change a user	usermod	usermod
List users	listusers	N/A
Password files	/etc/passwd and /etc/shadow	/etc/passwd and /etc/shadow
Group files	/etc/group	/etc/group

Task/locations	Solaris	Linux
Process resource limits for users	N/A	/etc/security/limits.conf
System-wide environment file	N/A	/etc/profile
Configuration information for user authentication	/etc/pam.conf	/etc/login.defs
Profile templates	/etc/skel/	/etc/skel/

Device management and configuration

Table A-5 provides information that shows differences in Linux and Solaris device management and configuration tasks.

Table A-5 Solaris versus Linux device management commands

Device management task	Solaris	Red Hat	SUSE
Run multiple tasks in a GUI environment	admintool	system-config-* (keyboard, mouse, printer, network, soundcard, and so on)	yast2
Configure a device	devfsadm	/dev/MAKEDEV kudzu	/dev/MAKEDEV hwinfo
Define a device	devfsadm	mknod kudzu	mknod hwinfo
Remove a device	devfsadm -C or -c	/dev/MAKEDEV	
List devices	sysdef	ls /sys/devices cat /proc/devices	
Install a device driver kernel module	modload	insmod	
List installed device driver kernel modules	modinfo	lsmod	
Remove a device driver kernel module	modunload	rmmmod	

Network management and configuration

Table A-6 provides information that shows differences in Linux and Solaris network management and configuration tasks.

Table A-6 Network management and configuration tasks

Task	Solaris	Linux
Run multiple tasks in a GUI environment	N/A	system-config-network in RHEL, yast2 in SLES
Configure TCP/IP	Editing the following files: <ul style="list-style-type: none"> ▶ /etc/hostname.* ▶ /etc/inet/* ▶ /etc/defaultrouter ▶ /etc/defaultdomain ▶ /etc/nodename ▶ /etc/netmasks 	RHEL: /etc/sysconfig/network and /etc/sysconfig/networking/* SLES: /etc/sysconfig/network/*
Display interface settings	ifconfig	ip ifconfig
Display interface status and statistics	netstat -i	netstat -i ifconfig
Configure interface	ifconfig	ip ifconfig
Check various network statistics	netstat	netstat
Change resolver	vi /etc/resolv.conf	vi /etc/resolv.conf
Configure name services	vi /etc/nsswitch.conf	vi /etc/nsswitch.conf
Display kernel network parameters	ndd /dev/ip \? ndd /dev/tcp \?	sysctl -a grep ^net
Configure kernel network parameters	ndd	sysctl -w variable=value
Check for network link	ndd	ethtool mii-tool
Rename a network interface	N/A	ip
Check routing table	netstat -r	netstat -r route
Testing for connectivity	ping	ping
Check IP path	traceroute	traceroute

Task	Solaris	Linux
Capture network packets	<code>snoop</code>	<code>tcpdump</code> <code>tethereal</code> <code>ethereal</code>

NFS management and configuration

Table A-7 provides information that shows differences in Linux and Solaris NFS management and configuration tasks.

Table A-7 Solaris NFS and Linux NFS differences

Task/configuration file	Solaris	Linux
Manually start the NFS server	<code>/etc/init.d/nfs.server start</code>	RHEL: <code>/etc/init.d/nfs start</code> SLES: <code>/etc/init.d/nfsserver start</code>
Mount a resource on a NFS client	<code>mount -F nfs server://resource /mount/point</code>	<code>mount server://resource /mount/point</code>
NFS server general config file	<code>/etc/default/nfs</code>	<code>/etc/sysconfig/nfs</code>
NFS logging options config file	<code>/etc/default/nfslogd</code>	<code>/etc/sysconfig/nfs</code>
Share all exported file systems	<code>shareall</code>	<code>exportfs -a</code>
Share a new file system	<code>share</code>	<code>exportfs -o</code>
Config file of shared file systems	<code>/etc/dfs/dfstab</code>	<code>/etc/exports</code>

Managing system resources

Table A-8 provides information that shows differences in querying Linux and Solaris systems for general information. For more detailed descriptions of the commands and their usage, refer to Chapter 9, “Managing system resources” on page 139.

Table A-8 System information tasks

Task	Solaris	Linux
List system information	<code>uname</code>	<code>uname</code>
List processor information	<code>psrinfo</code>	<code>cat /proc/cpuinfo</code>

Task	Solaris	Linux
List system memory size	<code>prtconf grep Memory</code>	<code>cat /proc/meminfo grep MemTotal</code>
List disk space usage	<code>df</code>	<code>df</code>
List file space usage	<code>du</code>	<code>du</code>
List physical disks	<code>format</code>	<code>fdisk -l</code>
Show the system's host name	<code>hostname</code>	<code>hostname</code>
List system's network configuration	<code>ifconfig</code>	<code>ifconfig</code>
List active processes	<code>prstat</code>	<code>ps -e</code>
List system configuration	<code>prtconf</code>	Red Hat: <code>lshal</code> SUSE: <code>hwinfo</code>
List system diagnostic information	<code>prtdiag</code>	Red Hat: <code>lspci</code> and <code>/proc/*</code> files SUSE: <code>hwinfo</code>

Managing system services

Table A-9 provides information that shows differences in Linux and Solaris service management and configuration tasks.

Table A-9 Summary of system service tasks

Task	Solaris	Red Hat	SUSE
Add a service that does not already exist in <code>chkconfig</code>	Copy start script into the appropriate run level directory, prefixed with <code>S</code> .	<code>chkconfig --add service name</code>	<code>chkconfig -a</code> or <code>--add service name</code>
Show existing service links	<code>ls</code> run level directory.	<code>chkconfig --list grep service name</code>	<code>chkconfig service name</code>
Enable service to start at boot time	Copy start script into the appropriate run level directory, prefixed with <code>S</code> .	<code>chkconfig service name on</code>	<code>chkconfig -s</code> or <code>--set service name on</code>

Task	Solaris	Red Hat	SUSE
Disable service from running at boot time	Rename the service start script to a name that does not start with S (typically lowercase s).	chkconfig service name off	chkconfig -s or --set service name off

Managing scheduling and cron

Table A-10 provides information that shows differences in Linux and Solaris scheduling and cron management and configuration tasks.

Table A-10 Scheduling and cron service tasks

Task	Solaris	Linux
Run a command at a predefined time	at	at
Run a command or set of commands periodically	cron and crontab	cron and crontab
Run a command or set of commands periodically, running when possible if system is off at the scheduled time	N/A	anacron and anacrontab

Managing quota

Table A-11 provides information that shows differences in Linux and Solaris quota management and configuration tasks.

Table A-11 Quota tasks

Task	Solaris	Linux
Display disk usage and limits	quota	quota
Edit users quotas	edquota	edquota
Check consistency between a file system and the disk quota file	quotacheck	quotacheck

Managing process accounting

Table A-12 provides information that shows differences in Linux and Solaris process accounting management and configuration tasks.

Table A-12 Process accounting tasks

Task	Solaris	Linux
Turn on process accounting	accton	accton
List user's connection time	N/A	ac
List commands run previously	lastcomm	lastcomm
List user's connection time and idle time	who	who

Printer management and configuration

Table A-13 provides information about tasks that are involved in printer management and configuration in Linux and Solaris. For detailed instructions about configuring Linux printers, see Chapter 10, "Printing services" on page 151.

Table A-13 Printer management and configuration

Task	Solaris	Linux
Run multiple tasks in a GUI environment	admintool	RHEL: system-config-printer SLES: yast2
Add a printer	lpadmin	lpadmin
Start a print queue	enable	lpc
Stop a print queue	disable	lpc
Display print queue status	lpstat	lpc or lpq
Cancel a print job	cancel	lprm
Add a print queue	lpadmin	lpadmin
Change a print queue	lpadmin	lpadmin
Remove a print queue	lpadmin	lpadmin
Display settings of a print queue	lpadmin	lpadmin

Disk and file system management

Table A-14 and Table A-15 provide information that shows differences in Linux and Solaris disk and file system management and configuration tasks.

Table A-14 Disk management tasks

Task	Solaris	Linux
Create device files for new device	devfsadm	udev
To create partitions on the disk.	format or fmthard	fdisk or parted
Display partition table	prtvtoc	fdisk or parted
Maximum usable partitions	7	15 for SCSI 63 for IDE

Table A-15 File system management tasks

Task	Solaris	Linux
Run multiple tasks in a GUI environment	N/A	N/A
Format a disk	format	fdisk
Check a file system	fsck	fsck
Mount a file system	mount	mount
Display available file system space	df	df
Partition a disk	format	fdisk
List a volume's table of contents	prtvtoc	fdisk
Format a file system	newfs or mkfs	mkfs
Unmount a file system	umount	umount
Back up file systems, files, directories	ufsdump	dump
Restore file systems, files, directories	ufsrestore	restore
Change a file system	tunefs	RHEL and SLES: resize2ext SLES: resize_reiserfs
Display file system information	cat /etc/vfstab	cat /etc/fstab
Display file system mount table	/etc/mtab	/etc/mtab

Swap management

Table A-16 provides information that shows differences in Linux and Solaris swap management and configuration tasks.

Table A-16 Swap tasks

Task	Solaris	Linux
Create a swap space	<code>swap -a</code>	<code>mkswap</code>
Enable a swap space	<code>swap -a</code>	<code>swapon</code>
Enable all swap spaces	N/A	<code>swapon -a</code>
Disable a swap space	N/A	<code>swapoff</code>
Disable all swap spaces	N/A	<code>swapoff -a</code>
Delete a swap space	<code>swap -d</code>	N/A
Display the swap space usage	<code>swap -s</code>	<code>swapon -s</code>
Display the swap space status	<code>swap -l</code>	N/A
Set swap priority level	N/A	<code>swapon -p</code>

Logical volume management

Table A-17 and Table A-18 on page 361 provide information that shows differences in Linux and Solaris logical volume management and configuration tasks.

Table A-17 Logical volume management

Volume management task	Solaris	Linux
Initialize a disk for VM	Create partition area/size: <code>metainit</code>	Create partition area/size: <code>pvcreate</code> <code>mkraid, mdadm</code>
Create a volume or volume group (VG)	<code>metainit volumename raidtype devices...</code>	<code>lvcreate LVname</code> <code>vgcreate VGname devicename</code> <code>mkraid, mdadm</code>
Enable the volume or volume group	N/A	<code>lvchange -a y LVname</code> <code>vgchange -a y VGname</code> <code>raidstart</code>

Volume management task	Solaris	Linux
Disable the volume or volume group	N/A	<code>lvchange -a n LVname</code> <code>vgchange -a n VGname</code> <code>raidstop, mdadm</code>
Delete the volume or volume group	<code>metaclear</code>	<code>lvremove LVname</code> <code>vgremove VGname</code>
Add a device to the volume or volume group	<code>metattach</code> or <code>metainit</code>	<code>lvextend LVname</code> <code>vgextend VGname newdevname</code>
Delete a device from the volume or volume group	<code>metadetach</code>	<code>lvreduce LVname</code> <code>vgreduce VGname devicename</code> <code>raidreconf</code>
Create a soft partition or logical volume (no RAID)	<code>metainit -p</code>	<code>lvcreate -Lsize -nLVname VGname</code>
Create a soft partition or logical volume (RAID 0)	<code>metainit</code> with RAID 0 on devices first, and then <code>metainit -p</code> to create the soft partition volume.	<code>lvcreate -iNumOfStripes -IStripeSize -nLVname VGname mdadm, mkraid</code>
Create a soft partition or logical volume on a specific device	Same as above, but the second <code>metainit -p</code> will have the devicename at the end of the command line.	Same as above, but add the devicename to the end of the command line.
Delete a soft partition or logical volume	<code>metaclear</code>	<code>lvremove /dev/VGname/LVname</code> <code>raidreconf</code>
Extend a volume or logical volume	<code>metattach volume [size or devicename]</code>	<code>lvextend -Lsize /dev/VGname/LVname</code> <code>raidreconf</code>
Extend a file system after volume has been grown	<code>growfs</code>	<ul style="list-style-type: none"> ▶ Red Hat - ext2/ext3: <code>resize2fs</code> ▶ SUSE - ext2: <code>resize2fs</code> ▶ Red Hat - reiser: <code>resize_reiserfs</code> <p>Note: View the man pages for unmount requirement, and check if online version is available.</p>
Reduce a volume or logical volume	<code>metadetach Volname devicename</code>	<ul style="list-style-type: none"> ▶ ext2: <code>resize2fs</code> <code>lvreduce</code> ▶ reiser: <code>resize_reiserfs</code> <code>lvreduce</code> <p>Note: View the man pages for unmount requirements.</p>

Table A-18 Volume management commands

Volume management task	Solaris	Linux
Set up or display metadb	metadb	vgdisplay, lvdisplay, lsraid
Display metadevice status	metastat	vgdisplay, lvdisplay cat /proc/mdstat
Initialize raw devices to metadevices	metainit	pvcreate
Attach metadevices	metattach	vgchange, lvchange
Detach metadevices	metadetach	vgchange, lvchange
Clear and remove metadevices	metaclear	vgremove, lvremove
Replace a metadevice	metareplace	raidreconf
Rename a volume	metarename	raidreconf
Check metadevice ID configuration	metadevadm	mdadm
Manage hot spares	metahs	mdadm, raidhotadd, raidhotremove
Set submirrors offline	metaoffline	mdadm
Set submirrors online	metaonline	mdadm
Change volume parameters	metaparam	vgchange, lvchange
Back up volume group metadata	N/A	vgcfgbackup
Recover soft partition configuration information	metarecover	vgcfgrestore
Set up root file system for mirroring	metaroot	N/A
Administer disk sets for sharing	metaset	N/A
Resynchronize volume during reboot	metasync	N/A
Expand a file system size	growfs	vgextend, lvextend, resize2fs, resize_reiserfs, raidreconf
Start up the kernel metadevice modules	/etc/system	N/A
Set the default number of available volumes	/kernel/drv/md.conf	/etc/sysconfig/lvm /etc/raidtab

General troubleshooting

Table A-19 provides general troubleshooting information highlighting differences between methods and configuration file locations between Solaris and Linux (RHEL, SLES).

Table A-19 General troubleshooting

Task/location	Solaris	Red Hat	SUSE
Change a host name	Minimum change required for the following files: <ul style="list-style-type: none"> ▶ /etc/nodename ▶ /etc/hosts ▶ /etc/hostname.* ▶ /etc/net*/hosts 	hostname (while running) or edit /etc/sysconfig/network (for permanent change after network restart)	/etc/HOSTNAME and /etc/hosts
List of well-known networking services and port numbers	/etc/services	/etc/services	
List of well-known protocols	/etc/protocols	/etc/protocols	
Display NFS and RPC statistics	nfsstat	nfsstat	
Display a snapshot of virtual memory	N/A	cat /proc/meminfo	
Display virtual memory statistics	vmstat	vmstat	
Display I/O statistics	iostat	iostat	
Report system activity	sar	sar	
Display simple and complex lock contention information	N/A	View /var/lock directory	
Report CPU usage	top	top	
Display system error log	dmesg	dmesg	
Display paging/swapping space	swap -l	cat /proc/swaps	
Disk crash dump utilities	dumpadm savecore	/etc/init.d/diskdump savecore	1kcd

Task/location	Solaris	Red Hat	SUSE
Disk crash config files	<code>/etc/dumpadm.conf</code>	<code>/etc/sysconfig/ diskdump</code>	<code>/etc/sysconfig/dump</code>
Network crash dump utilities	N/A	<code>/etc/init.d/netdump /etc/init.d/netdump -server</code>	<code>lkcd</code>
Network crash config files	N/A	<code>/etc/sysconfig/netdump /etc/netdump.conf</code>	<code>/etc/sysconfig/dump</code>
Crash analyzing tools	<code>mdb</code>	<code>crash</code>	<code>lcrash</code>
Crash dump default store directory	<code>/var/crash</code>	<code>/var/crash</code>	<code>/var/log/dump</code>
Specify users who have access to cron (every user has access to cron if the access file does not exist.)	<code>/etc/cron.d/cron. allow</code>	<code>/etc/cron.d/cron.allow</code>	
Specify users who have no access to cron	<code>/etc/cron.d/cron.deny</code>	<code>/etc/cron.d/cron.deny</code>	
Specify remote users and hosts that can execute commands on the local host	<code>/etc/hosts.equiv</code>	<code>/etc/hosts.equiv</code>	
Default superuser log	<code>/var/adm/su<code>log</code></code>	<code>/var/log/messages</code>	
Configure syslogd logging	<code>/etc/syslog.conf</code>	<code>/etc/syslog.conf</code>	
Display physical RAM	<code>prtconf</code>	<code>cat /proc/meminfo</code>	
Start or stop scripts directory	<code>/etc/init.d</code>	<code>/etc/init.d and /etc/rc.d</code>	<code>/etc/init.d</code>
Devices directory	<code>/dev and /devices</code>	<code>/dev</code>	

Network troubleshooting

Table A-20 on page 364 provides network troubleshooting information highlighting differences in methods and configuration file locations between Solaris and Linux (RHEL, SLES).

Table A-20 Troubleshooting network problems

Tasks	Solaris	Linux
Display interface settings	<code>ifconfig</code>	<code>ip</code> <code>ifconfig</code>
Display interface status and statistics	<code>netstat -i</code>	<code>netstat -i</code> <code>ifconfig</code>
Configure interface	<code>ifconfig</code>	<code>ip</code> <code>ifconfig</code>
Check various network statistics	<code>netstat</code>	<code>netstat</code>
Check DNS resolver	<code>more /etc/resolv.conf</code>	<code>more /etc/resolv.conf</code>
Check name services config	<code>more /etc/nsswitch.conf</code>	<code>more /etc/nsswitch.conf</code>
Display kernel network parameters	<code>ndd /dev/ip (\?)parameter</code> <code>ndd /dev/tcp (\?)parameter</code>	<code>sysctl -a grep ^net</code>
Configure kernel network parameters	<code>ndd -set driver parameter</code>	<code>sysctl -w variable=value</code>
Check for network link	<code>ndd driver link_status</code>	<code>ethtool</code> <code>mii-tool</code>
Query DNS	<code>nslookup</code> <code>dig</code>	<code>nslookup</code> <code>dig</code>
Check routing table	<code>netstat -r</code>	<code>netstat -r</code> <code>route</code>
Check ARP entries	<code>arp -a</code>	<code>arp</code>
Testing for connectivity	<code>ping</code>	<code>ping</code>
Check IP path	<code>traceroute</code>	<code>traceroute</code>
Capturing network packets	<code>snoop</code>	<code>tcpdump</code> <code>tethereal</code> <code>ethereal</code>



B

Commands and configuration files reference

This appendix contains the following tables:

- ▶ Table B-1 on page 366: Configuration file comparisons between Solaris and Linux
- ▶ Table B-2 on page 367: Command equivalencies between the two operating systems
- ▶ Table B-3 on page 369: Commands common to both systems

Configuration and other files

Table B-1 provides a quick reference to compare names and locations of configuration files between Solaris and Linux.

Table B-1 Configuration and other files

Solaris	Linux	Comments
/etc/auto_master, /etc/auto_home	/etc/auto.master, /etc/auto.home	
/etc/default/login	/etc/securetty	
/etc/default/nfs	/etc/sysconfig/nfs	
/etc/default/nfslogd	/etc/sysconfig/nfs	
/etc/defaultdomain, /var/yp/binding/'domainname'/yp servers	/etc/yp.conf	
/etc/dfs/dfstab	/etc/exports	
/etc/ftpusers (Solaris 8) /etc/ftpd/ftpusers (Solaris 9)	/etc/vsftpd.ftpusers(vsftpd) /etc/ftpusers (wu-ftpd)	
/etc/inet/ (directory)	/etc/hosts, /etc/HOSTNAME, /etc/sysconfig/network (directory)	
/etc/inet/services	/etc/services	
/etc/inetd.conf	/etc/xinetd.conf, /etc/xinetd.d (directory)	
/etc/krb5/krb5.conf	/etc/krb5.conf	Different format.
/etc/logadm.conf	N/A	See Chapter 12, "Monitoring and performance" on page 173.
/etc/logindevperm	/etc/logindevperm	Different formats.
/etc/mime.types	/etc/mime.types	Different format.
/etc/mnttab	/etc/mnttab	
/etc/nscd.conf	/etc/nscd.conf	Cache names different.
/etc/pam.conf	/etc/pam.d (directory)	
/etc/printers.conf	/etc/cups/printers.conf	Different format (see Chapter 10, "Printing services" on page 151).

Solaris	Linux	Comments
/etc/rmmount.conf	N/A	See Chapter 6, “Device management” on page 85.
/etc/security/policy.conf	N/A	See Chapter 11, “Users and groups” on page 159.
/etc/syslog.conf	/etc/syslog.conf	Slightly different format.
/etc/system	/etc/sysctl.conf	
/etc/vfstab	/etc/fstab	
/etc/vold.conf	N/A	See Chapter 6, “Device management” on page 85.
/usr/share/lib/zoneinfo/*	/usr/share/zoneinfo/*	
/var/adm/messages, /var/log/syslog	/var/log/messages, /var/log/syslog, /usr/adm/messages, /var/log/maillog	
/var/spool/cron/crontabs/root	/etc/crontab, /var/spool/cron	

Comparable commands

Table B-2 provides a quick reference for *comparable* commands between Solaris and Linux.

Table B-2 Comparable commands

Solaris	Linux	Usage
/sbin/swapadd	/sbin/swapon	Enable swap devices
adb	gdb	Debugger
admintool	yast2 (SUSE) /usr/bin/system-* (Red Hat)	Admin GUI
apprace	strace and ltrace	Application API tracing
arch -k	uname -m	List machine type
cc	gcc	C compiler
devfsadm	modprobe, kerneld, insmod, hotplug, cardctl	Add device without reboot

Solaris	Linux	Usage
<code>df -k</code>	<code>df</code>	File system allocation in units of kilobytes
<code>getfacl, setfacl</code>	<code>getfacl, setfacl</code>	Get, set ACLs
<code>gzcat file tar -xvf -</code>	<code>tar -xzvf file</code>	Unbundle compressed TAR file
<code>lpsched</code>	<code>lpd</code>	Printer daemon
<code>modinfo</code>	<code>lsmod modinfo</code>	List loaded kernel modules
<code>mountall</code>	<code>mount -a</code>	Mount all entries in fstab
<code>mvmir</code>	<code>mv</code>	Move a directory
<code>nawk</code>	<code>gawk</code>	Awk scripting language
<code>patchadd</code>	<code>rpm -U</code>	Update package
<code>patchrm, pkgrm</code>	<code>rpm -e</code>	Remove package
<code>pkgadd</code>	<code>rpm -i</code>	Add package
<code>pkgchk</code>	<code>rpm -V</code>	Verify package
<code>pkginfo, pkgparam</code>	<code>rpm -qa</code>	Query packages
<code>priocntl -e, nice</code>	<code>nice</code>	Start a process with a given priority
<code>priocntl -s, renice</code>	<code>renice</code>	Change the priority of a running process
<code>prstat</code>	<code>top</code>	Report process statistics
<code>prtdiag</code>	<code>dmesg</code>	Error reporting tool
<code>prvtoc</code>	<code>fdisk -l</code>	Read disk label
<code>ps -ef</code>	<code>ps [-]aux</code>	List all system processes
<code>snoop</code>	<code>tcpdump, ethereal</code>	Network packets sniffer
<code>tip</code>	<code>minicom</code>	Serial port access program
<code>truss, sotruss</code>	<code>strace, ltrace</code>	Tracing
<code>umountall</code>	<code>umount -a</code>	Unmount entries in mtab
<code>useradd</code>	<code>adduser</code>	Add a user account
<code>who -r</code>	<code>/sbin/runlevel</code>	Show run level

Solaris	Linux	Usage
xterm/dtterm	konsole/gnome-terminal	GUI terminal program

Common Solaris and Linux commands

Table B-3 provides a quick reference for *common* commands between Solaris and Linux.

Note: Although options and syntax might differ in some cases, the purpose of these utilities is the same.

Table B-3 Common commands and utilities

apropos	arp	as	at	atq	atrm
awk	bash	batch	break	bunzip2	bzcat
bzip2	cancel	captainfo	case	cat	cd
chgrp	chmod	chown	chroot	cksum	clear
col	comm	continue	cp	cpio	crontab
csh	csplit	ctags	cut	date	dc
dd	df	diff	dig	dirname	dirs
disable	dmesg	domainname	echo	ed	egrep
eject	enable	env	eqn	eval	ex
exec	exit	expand	export	expr	factor
false	fdformat	fdisk	fgrep	file	finger
fmt	format	fsck	ftp	function	fuser
gencat	getconf	getent	getfacl	getopt	getopts
gettext	gprof	grep	groups	gunzip	gzexe
gzip	head	help	history	host	hosted
hostname	iconv	init	install	ipcrm	ipcs
jobs	join	kdestroy	kill	killall	kpasswd
lastcomm	ldapadd	ldapdelete	ldapmodify	ldapmodrdn	ldapsearch
ldd	lesskey	let	link	ln	locale

localedef	logger	login	logname	logout	look
lp	lpq	lpr	lprm	lpstat	ls
mail	man	mesg	mkdir	mkfifo	mkisofs
mknod	mktemp	modinfo	more	mount	mpstat
mt	mv	neqn	netstat	newgrp	nfsstat
nice	nl	nroff	nslookup	nsupdate	passwd
paste	patch	pathchk	pax	perl	pgrep
ping	pskill	popd	printenv	profiles	ps
pushd	pwd	quota	ranlib	rcp	rdist
read	readonly	red	renice	return	rlogin
rm	rmdir	rpcinfo	rsh	rup	scp
script	sdiff	sed	select	set	setfacl
sftp	sh	shift	sleep	sort	source
split	ssh	ssh-add	ssh-agent	sshd	ssh-keygen
strace	strings	sty	su	sum	suspend
sync	syslogd	tar	tbl	tee	tic
time	timeout	times	touch	tput	tr
trap	troff	true	tset	tsort	tty
typeset	ul	ulimit	umask	umount	uname
unexpand	uniq	units	unlink	unset	uptime
users	vi	view	vmstat	wall	wc
whatis	whereis	which	whoami	whois	write
xargs	yacc	ypcat	ypinit	ypmatch	yppasswd
ypwhich	zcat	zipinfo			



UNIX to Linux Porting: A Comprehensive Reference (table of contents and sample chapter)

This appendix contains the Table of Contents and Chapter below, excerpted from *UNIX to Linux Porting: A Comprehensive Reference (ISBN 0131871099)*, reprinted courtesy of [Pearson Education, Inc.](#), © Pearson Education, Inc., 2006.

We provide this content to provide you with some technical perspective on the subject of application porting. Porting of applications and programs from the Solaris operating system platform to Linux is a broad technical topic that is outside the scope of this book. Nonetheless, completing required porting tasks will be a key part of any successful migration project. We are pleased to provide you with this introductory content about the subject of porting.

The sample content is presented in this appendix in two parts:

- ▶ Table of contents
- ▶ Chapter 1 Porting Project Considerations

Note: If you are interested in obtaining a copy of *UNIX to Linux Porting: A Comprehensive Reference*, you can receive a 35% discount on the publisher price by using the following URL and coupon code to order:

<http://www.prenhallprofessional.com/mendoza>

Coupon Code: MENDOZA-IBM

Table of contents

UNIX to Linux Porting: A Comprehensive Reference (ISBN 0131871099)

Authors:

Alfredo Mendoza
Chakarat Skawratananond
Artis Walker

Preface

Chapter 1: Porting Project Considerations

Chapter 2: Scoping

Chapter 3: Analysis

Chapter 4: Porting Solaris applications

Chapter 5: Porting AIX applications

Chapter 6: Porting HP-UX applications

Chapter 7: Testing and Debugging

Appendix A: Solaris to Linux Reference Tables

Appendix B: AIX to Linux Reference Tables

Appendix C: HP-UX to Linux Reference Tables

Appendix D: Linux on POWER

Appendix E: gprof helper

Chapter 1 Porting Project Considerations

Application porting refers to the process of taking a software application that runs on one operating system and hardware architecture, recompiling (making changes as necessary), and enabling it to run on another operating system and hardware architecture. In some cases, porting software from one platform to another may be as straightforward as recompiling and running verification tests on the ported application. In other cases, however, it is not as straightforward. Some large applications that were written for older versions of operating systems and compiled with native compilers may not adhere to present language standards and will be more difficult to port, requiring the full attention of project management.

This chapter supplements currently available project management materials and books about application porting projects. Topics such as how to use formalized requirements processes, how to better communicate between software developers, and how to practice extreme project management are topics that modern-day project managers are already well informed of. However, because software development is not exactly the same as porting and migrating software, a gap exists in current publications, a gap that this chapter addresses.

The focus here is on software porting and migration process details and technical risks. All information in this chapter has been collected and gathered through countless porting engagements. This chapter presents best practices and methodologies to achieve high-quality ported applications. This chapter will appeal specifically to those involved in software porting and migration projects, especially project managers and application architects.

Software Application Business Process

Before embarking on a porting project, it is important to note which business processes within the application life cycle will be affected. Business processes currently in place must be modified to accommodate the ported application in terms of having a new platform to support, new test environment, new tools, new documents, and (most important) new customers to support and establish relations with.

The three main areas of the application life cycle to consider that may affect the business process are development and testing, customer support, and software application distribution:

- ▶ **Development and testing.** Within the development and test organization, personnel must be trained to develop and test on Linux in terms of application programming interface (API) differences, development tools, debugging tools,

performance tools, third-party applications needed by the application to be ported, and performance tools.

- ▶ **Customer support.** Within the customer support organization, support personnel must be trained on Linux system administration, third-party applications needed by the ported application, installation and administration procedures, package maintenance options in the Linux environment, debugging tools and procedures, and anything else that needs to be learned to provide customer support for the ported application.
- ▶ **Software application distribution.** Within the software application distribution organization, sales and consultants must be trained in overall Linux features and capabilities.¹ Personnel in the software distribution channels must be trained as trainers of the software application in a Linux environment. From a customer's perspective, they will also look for integration skills to integrate Linux within their IT infrastructure.

Porting an application to Linux means modifying business organization processes that can be affected by a newly ported application. The three main areas should be considered and included in the overall project plan before starting the porting process.

The Porting Process

Developers involved in porting projects tend to follow similar steps when porting any software application. These steps include scoping, analyzing, porting, and testing. Each step in the process builds a foundation for the next step in the process. Each step, when done properly, makes the next step of the process easier to accomplish. Figure C-1 shows the high-level steps taken during a porting project.

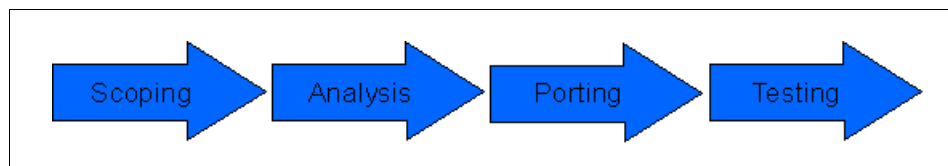


Figure C-1 High-level steps taken during a porting project

¹ Different Linux distributions exist that may require separate training sessions for each.

Scoping

Scoping is the step in which the project manager asks the porting expert² and a domain expert³ to get together to identify the products, development, and test environment the application to be ported relies on. The key areas to identify during the scoping process include product dependencies, development environment components, build environment components, and test environment components:

- ▶ **Product/software dependencies.** Identifying which products the application to be ported relies on means determining which versions of database, middleware, and third-party libraries it uses. By knowing the products and versions, the porting expert can assess whether those products and versions are available for the Linux platform.
- ▶ **Development environment components.** Assessing the development environment includes identifying what programming language the application is written in. Applications written in a more recent programming language such as Java tend to port more easily, whereas applications written in C or C++ usually require more analysis and porting effort.
- ▶ **Build environment components.** Assessing the build environment includes making sure the build tools are available on Linux. Platform-dependent compiler and link flags used on the source platform must be investigated to determine whether equivalent flags exist on Linux. Some build environments may have dependencies on the source platform that it will require a little more effort to port to Linux.
- ▶ **Test environment components.** Identifying the test environment for the ported application leads to questions pertaining to the ownership of testing after the application is ported. Usually porting engineers will do unit testing for the part of the application they port and then hand it off to a testing group for more verification and systems tests. But who is the testing group?

In most cases, the scoping step leads to identifying associated risks that will be assumed by the project as a whole when it begins. Some risks that can be identified in the scoping step are included in the following scenarios:

- ▶ Versions of the database, middleware, or third-party libraries are not available for the Linux platform.
- ▶ The application has some assembler routines that need to be translated to assembly language instructions on the Linux platform.

² A software developer who is experienced in porting applications and has knowledge of source and target platforms as well as other third-party products the application uses. We also refer to this person as a porting engineer in this book.

³ A person who is knowledgeable about the application to be ported. This could be the application architect or the chief developer of the application.

- ▶ The application was written using a set of APIs or a programming model unique to the source platform. This also includes assumptions about such things as word size or “endian-ness.”
- ▶ The application was written to draft-standard C++ semantics that relied on the source platform’s native compiler framework.
- ▶ The test environment requires complicated client/server framework.
- ▶ The development environment requires third-party tools that need to be compiled or ported to the Linux platform.
- ▶ The distribution or installation method of the application requires facilities or tools unique to the source platform.

Scoping is an involved step in the porting process that takes into account every new piece of information that can be learned from asking the right questions—questions about documentation, packaging, and performance tuning, to name a few. These and others are mentioned in the “sample porting questionnaire” at the end of this chapter.

Analysis

There are two views in the analysis step of the porting process: a project management point of view, and a porting point of view. From a project management point of view, analysis is the step that assesses the various porting issues and risks identified in the preceding step and what impact they bring to the porting project as a whole. The analysis step involves the formulation of the project plan, which includes identifying scope and objectives, creating work schedules, procuring resources needed, and assigning roles within the project.

Identification of scope and objectives defines the boundaries and responsibilities of the project manager and the members of the project team. Boundaries are clearly identified sets of work that need to be done for the project. For example, a simple statement such as “Module A in application XYZ needs to be ported and tested on platform B” can be a good start to defining the boundaries of the work.

After boundaries have been identified, tasks for the porting work can be defined, leading to a work breakdown schedule.⁴ The work breakdown schedule helps define which tasks need to be done and whether those tasks can be done in sequence or in parallel. In addition, the work breakdown schedule identifies the resources needed. An overall schedule for the entire porting project results from identifying tasks and resources needed.

From a porting point of view, analysis is a step in the porting process during which a porting engineer examines the application architecture in more detail.

⁴ A project management term used to denote work as an activity arranged in a hierarchical order that has tangible results otherwise referred to as a deliverable.

The porting engineer begins to identify the APIs and system calls used in the application. In addition, the application is assessed as to its use of dynamic linking and loading, networking, threads, and more. This analysis feeds information back to the project manager that can be used to formulate more detailed tasks and accurate schedules.

Porting

Porting is the step in the process during which the porting engineers perform their assigned tasks. Depending on the application and the work breakdown schedule that resulted from the preceding step, porting engineers may only be able to work their assigned tasks in a serial fashion. Serial tasks are a product of tightly coupled applications. This means that some modules of the application are highly dependent on other parts of the application and that these modules can be worked on only when the part it depends on is ported. A prime example of this is the build environment. If the build environment was designed to build the whole application as a monolithic procedure, chances are all modules depend on common configuration files that need to be modified before any porting work can be performed.

If the porting tasks are independent of each other, however, the work effort can be parallelized. Loosely coupled modules can be worked on separately and simultaneously by different porting engineers. A key example of this are shared libraries that can be built independently of each other, do not have common components, and exist for other modules to build on. Identifying which tasks can be done in parallel is important and should be done as part of the analysis step of the process.

The porting engineers' task of compiling the code on the Linux platform includes identifying and removing architectural dependencies and nonstandard practices if possible. Identifying and removing architectural dependencies means heeding compiler errors and warnings produced at compile time and correcting them as needed. Removing architectural dependencies and nonstandard practices involves examining and changing the code to use more portable structures or coding standards. Experienced and quality-conscious porting engineers usually do the latter as they correct compiler errors and warnings.

The porting effort includes porting the build environment to the Linux platform, too. This task should be identified during the scoping step. Although some build environments are portable, some are not. Making sure the build environment does not create any potential problems is a task easily overlooked and needs to be scoped and analyzed with extreme caution.

After the application has been ported—meaning it compiles on the Linux platform without errors—the porting engineer is expected to run unit tests on the application. The unit tests can be as simple as executing the application to

determine whether it produces any runtime problems. If runtime problems are detected, they are debugged and fixed before the application is handed to the testing group; the test group then runs more thorough tests on the ported software.

Testing

During testing, the assigned testers run the ported application against a set of test cases, which vary from simple execution of the application to stress-type tests that ensure the application is robust enough when run on the Linux platform. Stress testing the application on the target platform is where most porting problems related to architectural dependencies and bad coding practices are uncovered. Most applications—especially multithreaded applications—will behave differently during stress testing on a different platform, in part because of different operating system implementations, especially in the threading area. If problems are found during testing, porting engineers are called in to debug and fix them.

Some application porting may also include porting a test harness to test the application. Porting this test harness is a task that is also identified during scoping and analysis. Most of the time, the testers need to be trained on the application before they can test it. Learning about the application is a task that is independent of the porting tasks and can be performed in parallel.

If bugs are found, they are fixed and the source recompiled; the testing process repeats until the application passes all testing requirements.

Support

When the port is complete, the development phase ends, and the support phase begins. Some of the porting engineers are retained to help answer any porting-specific questions the customer may have. In addition, the developers are expected to train the customer on configuring and running the application on the Linux platform. The typical support phase lasts from between 60 to 90 days after the port—in which time, the porting engineers train and answer technical support and sales personnel questions regarding the newly ported application on the Linux environment. After the technical support and sales personnel have been trained in the newly ported product, the porting engineer's task is done.

Defining Project Scope and Objectives

Project scope is defined as the specific endpoints or boundaries of the project as well as the responsibilities of the project manager and team members.⁵ A clearly

⁵ *Radical Project Management*, Rob Thomsett, 2002, Prentice Hall

defined project scope ensures all stakeholders (individuals and organizations involved in or that might be affected by project activities—project team, application architect, testing, customer or sponsor, contracts, finance, procurement, contracts and legal) share a common view of what is included as the objectives of the project.

Clear project objectives/requirements lead to a better understanding of the scope of the project. During the analysis step in the porting process, customer objectives and requirements are gathered, and these objectives transform themselves to work breakdown structures and eventually project deliverables. Project objectives and requirements provide a starting point for defining the scope of the project. After all project objectives have been laid out, the project scope becomes more defined.

One way to define the scope of a project is to list objectives that will and will not be included in the project. A requirements list from the customer is a good start. After the requirements have been gathered, the project manager and the technical leader review the list in detail. If there are questions about the requirements list, they should appear in the “not to be included” list of objectives, at least initially. The list is then reviewed by the customer, who may make corrections to the list. The list is revised until all parties agree that the requirements list presents the true objectives of the project.

Again, it should be sufficiently detailed so that it lists requirements that will and will not be included in the project. Yes, it is really that important to include details that are beyond the scope of the project. Objectives that do not provide enough definition to the scope of the project may turn out to be a point of contention as the project draws closer to its release date. An example of this might be listing how pieces of the porting work will be delivered to the porting team—will it be in phases or will it be in a single delivery? Will each delivery be considered a phase, or will there be smaller pieces of work within a phase? How many phases will make up the entire project? The list not only defines the project itself, it also sets the expectations for all stakeholders in the project.

Here are the basic rules to follow when creating the project objectives list:

1. Define project scopes with as much detail as possible.
2. Confirm that all stakeholders agree to project scopes.
3. List unresolved items on the “not-included/not-in-scope” list until they are resolved.

Table C-1 on page 380 shows an example of a simple “in-scope” and “not-in-scope” table for a sample porting project.

Table C-1 *In-Scope and Not-in-Scope Objectives*

Project: Customer X Application Port to Linux	
In Scope	Not in Scope
Port Directory, Mail, and Order Entry modules to Linux to be divided up in three phases. Each module will be one phase.	System tests of in-scope modules are not included in the porting project. Customer will be responsible for systems test.
Functional verification tests on these modules will be conducted by porting team. Definition of functional verification tests and acceptance criteria will be defined in another section.	Packaging of application is not in scope. Fixing bugs found to be present in the original source code are not in scope.
Provide debugging assistance to customer developers during system verification tests.	External customer documentation of ported modules is not in scope. Customer to produce external customer documentations.

Part of defining the objectives is listing the acceptance or success criteria. A consensus regarding the porting project success criteria must be reached by all stakeholders. Project success may mean passing a percentage of system tests on the Linux platform or passing a level of performance criteria set out by the systems performance group—that is, if the project objectives are to run a specific set of tests or performance analysis, respectively. Regardless of how the project success is defined, all stakeholders must understand and agree on the criteria before the porting effort starts, if possible. Any changes to the criteria during the course of the porting cycle must be communicated to all stakeholders and approved before replacing the existing criteria.

Estimating

Many porting projects go over budget and miss schedules because risks were not managed properly. Risks play a major part in estimating the schedule as well as resources needed for porting projects. In an application porting project, such risks will come from different aspects that relate to application porting, including the following:

- ▶ Skill levels and porting experience
- ▶ Compiler used
- ▶ Programming language used
- ▶ Third-party and middleware product availability
- ▶ Build environment and tools
- ▶ Platform-dependent constructs

- ▶ Platform- and hardware-dependent code
- ▶ Test environment setup needed
- ▶ User interface requirements

Depending on the application to be ported, each of these aspects will present varying levels of complexity and risks to the project. Assessing complexity and risk level help you determine whether they are manageable.

Skill Levels and Porting Experience

The most glaring difference between porting applications and software development is the skill set of the programmers. Although software application developers tend to be more specialized in their knowledge domain, software developers who do porting and migration need broader and more generalized skill sets. An application developer can be someone who is an expert in Java and works on a Windows development environment. Another may be a programmer specializing in database programming on a Sun Solaris operating system environment. On the other hand, engineers who port code are expected to be experts in two or more operating system platforms, programming languages, compilers, debuggers, databases, middleware, and the latest Web-based technologies. They are expected to know how to install and configure third-party database applications and middleware.

Whereas application developers tend to be specialists, porting engineers need to be generalists. Application developers may work on an application for about 18 months (the typical development cycle), whereas porting engineers work on a 3- to 6-month project cycle and are ready to start on a new port as soon as the last one is finished. Finding skilled porting engineers who fit the exact requirements of a porting project may be difficult at times. This is most true when the porting efforts require porting from legacy technology to newer ones.

Compiler

The compiler and compiler framework used by the source platform makes a big difference when porting applications to the Linux environment. If the compiler used in both the source and target platforms are the same, the task becomes easier. An example in this case is when the GNU compiler is used in both the source and the target platform. Besides the `--g` and `--c` flags, different brand of compilers use flags differently. Compiler differences become more difficult if the programming language used is C++.

Another thing to consider is the version of the compilers. Source code compiled a few years back with older versions of compilers may have syntax that does not conform to present syntax checking by compilers (because of standards conformity). This makes it more difficult to port on different compilers because these compilers may or may not support backward compatibility for standards.

Even if the compilers support older standards, different compilers may implement support for these standards differently.

Third-Party and Middleware Product Availability

The complexity of the port increases when the application to be ported uses third-party and middleware products. Complexity increases even more when the versions of those products are not available for the target platform. In rare cases, some third-party products may not even be available on the target platform at all. In the past few years, middleware vendors such as IBM, Oracle, and Sybase and many more have ported their middleware products to Linux. They have made efforts to make their products available for companies that are ready and willing to make Linux their enterprise platform. This is partly why we are seeing more and more companies willing to port their applications to Linux.

Build Environment and Tools

The simpler the build environment, the less time the porting team takes to understand how to build the source code. More complicated build environments include multipass builds where objects are built with different compiler flags to get around dependencies between modules. After a first-pass build to build some modules, a second pass compiles yet more modules, but this time builds on top of the previously compiled modules (and so on until the build completes). Sometimes the build scripts call other scripts that automatically generate files on-the-fly based on nonstandard configuration files. Most of these files may be interpreted as ordinary files to be ported, but in fact, it is really the tools and the configuration files that need to be ported to work on the target module. These types of tools are mostly missed during assessment and analysis, which can result in less-than-perfect schedules.

One build environment that came into fashion sometime in the early 1990s was the use of imake. Imake is a makefile generator intended to ease build environment portability issues. Instead of writing makefiles, one writes imakefiles. Imakefiles are files that contain machine-independent descriptions of the application build environment. Imake creates makefiles by reading the imakefile and combines it with machine-dependent configuration files on the target platform. The whole concept of architecting a build environment around the imake facility to make it portable is a noble idea. Of the most recent 50 porting projects done by our porting group, only a few were built around the imake facility. Unfortunately, all of them needed as many modifications to their own environment and imake files, negating the ease that it is intended to provide in the first place. And because imake was rarely used, anyone who needed to do a port needed to “relearn” the utility every time.

Nowadays, makefiles are written for the open source make facility called GNU Make (or gmake, for short). Most makefile syntax used in different UNIX

platforms is syntax that gmake can understand or has an easy equivalent to. Architecting applications around the gmake⁶ facility is the most accepted and widely used method in build environments today.

Source code control is another aspect that must be considered. In the Linux environment, CVS is the most frequently used source code control environment. Other source code control environments are Subversion⁷ and Arch.⁸ A porting project needs to have source code control if there is a possibility of several porting engineers working on the same modules at the same time.

Platform-Dependent Constructs

When porting from UNIX platforms based on RISC architecture to x86-based platforms, a chance exists that application code needs to be assessed for byte-endian dependencies. For example, the application implements functions that use byte swapping for calculations or data manipulation purposes. Porting the code that uses byte-swapping logic to an Intel-based machine, which is little-endian architecture, requires that code be modified to adhere to little-endian semantics. In cases where byte-swapping logic is not changed between platforms, debugging becomes a chore because it is difficult to track where data corruption takes place when it happens several instructions before the application fails. Make sure to identify platform-dependent constructs in the scoping and analysis steps.

Platform- and Hardware-Dependent Code

Applications that require kernel extensions and device drivers to operate are the hardest to port. Kernel APIs for all platforms do not follow any standards. In this case, API calls, number of arguments, and even how to load these extensions into the kernel for a specific platform will be different. In most cases, new code that needs to run on Linux must be developed. One thing is for certain: Kernel code will usually, if not always, be written in C or platform-dependent assembler language.

Test Environment and Setup

When the port is done and the project scope includes system tests and verification, equipment required to test the code may contribute to the complexity of the port. If the application needs to be tested on specialized devices that need to be ordered or leased, this can add complexity to the project. If the application needs to be tested in a complicated clustered and networked environment, setting up the environment and scheduling the resource adds time to the project.

⁶ Some may favor Autoconf even though it may add some level of complexity compared to using just plain old Makefiles.

⁷ <http://subversion.tigris.org>

⁸ <http://www.gnu.org/software/gnu-arch/>

Testing may also include performance testing. Normally, performance tests require the maximum configuration setup one can afford to be able to load up the application to see how it scales in big installations.

Porting of the application test harness needs to be included in the porting schedule. A test harnesses that includes both software tests and specialized hardware configurations need to be assessed as to how much time it will take to port and configure.

User Interface Requirements

Porting user interfaces can range from being easy to complex. User interfaces based on Java technology are easy to port, whereas user interfaces based on X11 technology are more complex. In this type of application, the code may be written in C or C++.

Table C-2 is a summary of the various aspects involved. Each aspect has been ranked as easy, medium, or high complexity, depending on the technology used for each aspect.

Table C-2 Software Porting Risks and Complexities

Aspects	Easy	Medium Complexity	High Complexity
Compiler/programming Language	Java Shell scripts	Use of C, COBOL, Fortran language Use of nonportable syntax Original source code written to other standards of compiler	C++ using compiler-specific framework that supports varying levels of the C++ standard Source code written in languages other than Java or C Use of nonstandard compilers such as ASN or customer-built interpreters Use of assembler language Use of RPG language
Use of third-party products or middleware	None	Supported and available on target platform	Not supported on target platform Use of third-party tools written in C++

Aspects	Easy	Medium Complexity	High Complexity
Build environment and tools	Simple make files	Combination of make files and build scripts	Use of noncommon build tools such as imake, multiple pass builds, on-the-fly-generated code modules
Platform / hardware-dependent code	No platform- or hardware-dependent code	Platform/hardware-dependent code comes from third-party products already ported to target platform	Extensive use of kernel extensions and device driver code
Test environment and setup	Stand-alone	Client/server setup	Networked, high availability, clustered Needs external peripherals to test such as printers, Fibre Channel-attached disks
User interface	Java based	Standard-based interface such as X11, CDE	Nonportable user interface such as custom interfaces using proprietary code

Experience shows that real efforts needed to port the application are uncovered during the first two or three weeks of the porting project. This is the time when the code is delivered to the project team and porting engineers get a first glimpse of the software application. In other cases, this will be the first time porting engineers will work on the application in a Linux environment. They begin to dissect the application components and start preliminary build environment setups. Within the first two to three weeks, the porting engineers will have configured the build environment and started compiling some modules using GNU-based tools and compilers.

After two to three weeks, it is good to revisit the estimated time schedules to determine whether they need to be adjusted based on preliminary real experiences of porting the application into the Linux platform.

Creating a Porting Project Schedule

When creating the porting schedule, consider all risks, including technical and business-related issues. In terms of technical issues, common things such as

resource availability, hardware availability, third-party support, and Linux experience need to be considered. On the business side, issues such as reorganizations, relocations (moving offices), customer release dates, and business objective changes may affect the porting schedule as a whole.

These technical and business issues will create several dependencies external to the porting project that may not be controlled; therefore, it is advisable to consider each external issue to mitigate risks accordingly.

Creating schedules for porting projects is just like creating schedules for software development projects. Figure C-2 is an adaptation of a diagram from *Radical Project Management* by Rob Thomsett (Prentice Hall, 2002, p. 191). The figure shows that the average estimation error made before the project scopes and objectives are cleared up is +400 percent and -400 percent. As the objectives, scope, risks, and other technical and business issues are clarified, the estimated porting schedule comes closer to the real schedule.

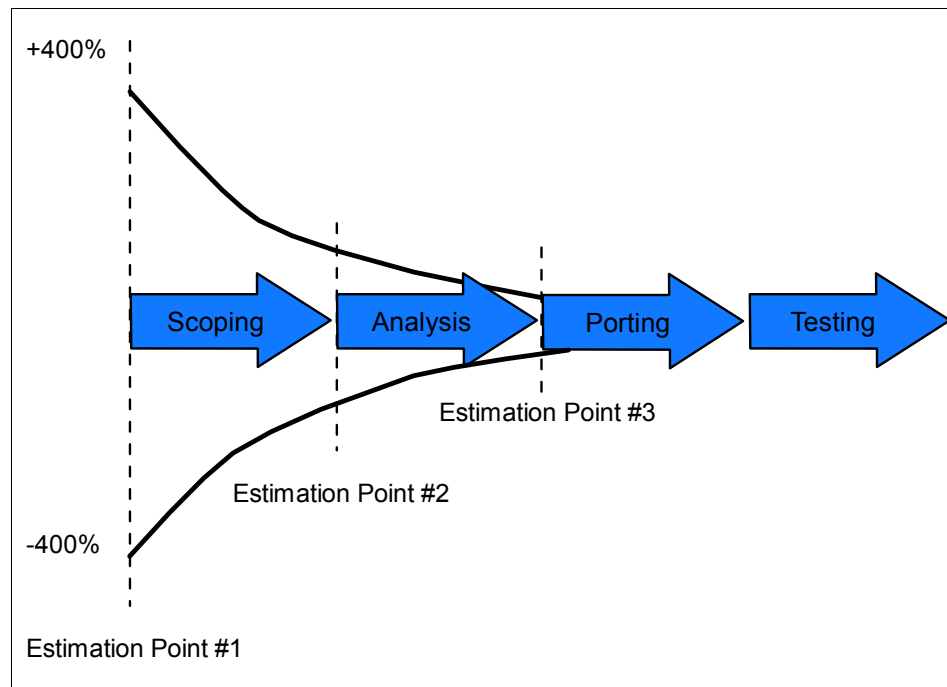


Figure C-2 Average estimation error made before scopes and objectives are cleared

Each transition to the next step in the porting process allows the project team to reassess the project estimates in terms of the schedule and resources. In essence, these transitions can and should serve not only as milestones for the project but also as checkpoints to reexamine the original estimates.

Porting Process from a Business Perspective

The porting process is not just about porting the software application; the resulting ported application will need the necessary business and support structures to make it a successful business initiative. Although porting efforts are taking place, the stakeholders need to prepare other organizations that will provide support for the application. Organizations such as customer support and software distribution will require documentation and training on the application running on the Linux environment.

Getting the necessary Linux training for personnel in the customer support and software distribution should be high on the priority list of objectives. As with any new product, new releases tend to generate the most questions from users and system administrators alike. Expect to answer system administration questions about Linux, too. Our experience shows that newly ported applications on a new operating environment will generate system administration questions and the use of the new operating system in three out of five support line calls.

As system administration and installation questions begin to subside, more technical questions about the application will arise. Support organizations need access to live test machines to duplicate customer problems. These machines may differ from development or porting machines depending on the application and technical problems that will require instances of the application to be running for debugging purposes.

From an overall project perspective, the need to train support personnel and the availability of physical resources to support the ported application will come at later stages of the porting project. As the technical aspects of the porting project near completion, the business aspects of the project pick up.

Annotated Sample Technical Questionnaire

This questionnaire, based on which other scoping questions can be formed, serves as a guide to the porting technical leader. The customer in this case is an internal or external organization that needs to port their application(s) to the Linux platform.

Platform-Specific Information

1. What is your current development platform for the application?

The question is about the development platform used to develop the application. It does not assume that the development platform is the same platform the application is deployed on. This is asked in the next question.

2. What platform does the application currently run on?

Porting engineers need to know the source or reference platform the application to be ported is using.

3. Has this application been deployed on any other platform than the development one? If yes, which version of the platform is it running on?

Asking this question gives you a sense of the portability of the application if it has been ported to other platforms. A word of caution: Although an application may have been ported to other platforms, it may have been done on older versions of the platform.

4. Describe any hardware (that is, graphics adapters, or cards) used by your application and whether the required drivers are available on the Linux platform.

Make sure any platform dependencies are available on Linux.

Application-Specific Information

1. Please describe your application and its architecture in detail.

This is where the customer describes the application architecture. Have them include an architectural diagram if possible. All components in the application need to be described. This should also give you what type of application framework, if any, the application runs on. Most Java applications run on product-specific frameworks such as WebSphere or Weblogic. If they are written in C++, they may run on a CORBA framework, which means you may have to deal with a specific CORBA framework that may or may not exist on Linux.

2. What are the different components of your software? Please provide name and version numbers for each component.

This will give you a sense of the breakdown of their applications. Being able to break down the application into different discrete components can mean that the porting work can be broken down into smaller independent tasks.

3. Which of these pieces will need to be ported or not ported? Please include the version number?

The customer needs to tell you what is in scope and what is not in scope.

4. What percentage of the application(s) to be ported is written in the following programming languages?
 - a. Java
 - b. C#
 - c. C
 - d. C++
 - e. Assembler
 - f. Visual Basic (Microsoft)
 - g. Shells (ksh, csh, perl, awk, others)

Ascertains the complexity of the application by asking what languages they are using and what percentage of those are in use.

5. Please provide a rough estimate of the number of lines of code in the software, listed by language types.

This is another way of asking Question 4. Asking questions in different light often brings up data points that contradict each other. This opens the door for open discussions that can result in better project scoping.

6. For Java apps: Does the application use the JNI⁹ to link native libraries? Please describe.

Ascertains the complexity of the application to be ported. Most of the time, Java applications that are not 100 percent pure Java need platform-dependent routines that can only be handled in native language such as C. Be aware of such platform-dependent code because they take more time to port.

7. Does the application use kernel modules? If yes, please describe.

Ascertain complexity of the application to be ported. Kernel modules and routines used by the application are nonportable. These will take more time to convert to equivalent routines in Linux.

8. Is it a 2D/3D graphics application? Please describe.

Ascertains complexity of the application to be ported. Make sure compatible graphics toolkits and development tools are available in Linux, whether they are supplied in Linux by default or by other third-party vendors.

⁹ Java native interface

9. Does the application use UNIX pipes, message queues, shared memory, signals, or semaphores? Please describe.

Most of these have standard UNIX interfaces that can be easily ported to Linux. Although the interfaces may be standard, implementation underneath the covers will differ. The catch is to make sure that the intended behavior remains the same when ported to Linux.

10. Is the application, or any of its components, multithreaded? If yes, which threads library is currently being used? Does the application rely on any proprietary threading priorities on your development platform?

Depending on the source platform, multithreading interfaces can vary from standard to nonstandard. Linux supports several threading libraries, but the one that is standard in present and future Linux distributions is the Native Posix Threads Library (NPTL) implementation. NPTL is discussed in other sections of this book. The point is the closer the source implementation is to NPTL, the easier it is to port.

11. Does the application perform operations that assume specific byte storage order? Is this likely to be an issue during the port?

This question relates to the “endianess” of the application. Most Linux ports will target the Intel platform, which is small endian, whereas most source platforms will be big endian. Nonportable code that assumes endian-specific characteristics will break when not ported correctly. Worse yet, the ported code will not exhibit errors during the porting phase. The problem usually crops up during system testing where it is harder to find.

12. Which compiler(s) and version is used in the development platform?

- a. GNU
- b. Java (what version)
- c. Platform (HP, AIX, Sun, Microsoft, NCR, AS/400, S390, True64) compiler?
Which platform?
- d. Others (please specify)

Ascertains complexity of the application to be ported. If the source application uses the GNU gcc or g++ compiler, it becomes easier to port to Linux because the native compiler for Linux is GNU gcc or g++. Applications that are developed on other platforms and are compiled in their native compilers tend to use native compiler semantics, which must be converted to GNU compiler semantics. C++ applications become harder to port than C applications when the application starts to use C++ features such as templates. Because some C++ standards are implemented differently by different compiler vendors, porting this type of code will take more time than simpler C++ code.

13. In addition to the development environment, are there any dependencies on debugging tools such as memory leak debuggers, performance analysis tools, exception handling, and so forth?

This goes back to scoping and dependencies. Third-party tools that may or may not exist in Linux need to be assessed. Who needs to provide for the license? Who is responsible for obtaining the tools? What will be the support structure if third-party support is needed?

14. Is it a socket-based application? If yes, does it use RPC? Please describe.

Although Linux supports standard-based socket and RPC semantics, the intent is ascertain portability. Asking this question may bring to light nonportable architecture the customer may have implemented in the application. This question can also lead to questions on what setup is needed at the testing phase.

15. Does the application use any third-party software components (database tools, application server, or other middleware)? If yes, which ones?

Every third-party software component adds complexity to the port. If any third-party software components are used, ask what version of the component is used and whether it is available on Linux. Third-party components may require extra time to learn and even configure or build if necessary

16. How is the application delivered and installed? Does it use standard packaging? Will the installation scripts need to be ported, too?

A Linux standard packaging mechanism is RPM. RPM is discussed in other parts of the book. Ascertain whether the customer will need the packaging part of the application ported, too.

17. Is the application or any of its components currently in 64 bit? Will any component need to be migrated to 64 bit?

With the advent of 64-bit platforms and operating systems, this question pertains to the level at which the application needs to run or be ported. Most 32-bit applications will port to a 64-bit environment without problems through the use of modern compilers. Today's compilers have become efficient at flagging syntax and semantic errors in the application at compile time, allowing the porting engineer to make necessary changes. The one consideration here is that it will require additional time for porting and debugging.

Database Information

1. What databases are currently supported? Please include version numbers.

Almost all enterprise applications today require a database back end. Making sure the database for the application is available on Linux is important. Differences in database products and versions can add substantial porting effort to the project.

2. What database is the ported application expected to run with?

In addition to Question 1 in this section, what database does the customer expect the ported application to run with on the Linux platform?

3. Does the application use any nonrelational or proprietary databases?

Any proprietary database needs to be ported to Linux. Make sure the code to run the database is available and is part of the scoped porting work.

4. How does the application communicate with the database?

- a. Programming language(s) (for example, Java, C/C++, other)
- b. Database interface(s) (for example, ODBC, OCI, JDBC, etc.)

Ascertains that programming languages and interfaces are available on Linux, whether or not supplied by third-party vendors.

5. Does the application require the use of extended data types (XML, audio, binary, video, and so on)?

This information can be used to assess the skills needed by the porting group for porting the application.

Porting Project Time Schedule Information

1. What is the desired General Availability date for the application on the target platform?

This question is asking whether any business objectives need to be considered when creating the porting schedule.

2. Has the porting of the application already started on the target platform?

Sometimes there may have been efforts in the past to port the application to Linux. This may be helpful in assessing complexities and problems encountered on the port.

3. What is the estimated port complexity level (low, medium, or high)?

Take the answer to this question with a grain of salt. There may be other factors present today that may have not been fully realized in previous porting efforts.

4. What factors were considered in this complexity ranking?

Any information from previous porting efforts needs to be assessed and compared to future porting efforts on the Linux platform.

5. If the application has been ported to another platform, how long did that port take? How many resources were dedicated to it? What problems were encountered?

This question attempts to compare previous porting efforts to the Linux port. This is only useful if the porting engineer technical lead has previous porting experience on other platforms as well as Linux.

6. What is your rough estimate of the project porting time and resources required?

The application or parts of it may have already been ported to other platforms, and knowing the time it took to port to those other platforms may be of some use. Experience and lessons learned from previous ports may come in handy. Knowing some of the lessons learned may help in avoiding problems when porting to Linux.

Testing-Specific Information

1. Please describe the acceptance testing setup.
2. What kind of networking and database setup would be required for unit testing?
3. How much testing will be required after porting (number of days, number of resources)?
4. Do you have established test scripts and application performance measurements?
5. Will benchmarks need to be run for comparison testing?
6. Is performance data available on current platforms?
7. When were the performance tests last executed?

All “testing-specific” questions pertain to application software testing on the Linux platform. Asking these questions may bring out other issues related to porting test scripts and the software application test harness, which will add risks and time to the whole project. Pay close attention to the response to Question 1 of this section. Question 1 relates to the type of acceptance criteria that needs to be agreed on before porting starts. The acceptance criteria can be something like passing all tests suites from tests scripts or test programs that tests the newly ported application. When the acceptance criteria are met, the port is considered complete and formal QA tests can then be performed on the application.

Porting Project Execution Information

1. Please select one or more options, according to how you would like to proceed with this project.
 - a. Technical assistance will be provided to the porting engineers as necessary.
 - b. Customer will be responsible for acquiring third-party licenses and technical support.
 - c. Other (please describe).

Add other items in this part of the questionnaire that you need the primary customer to consider. Some issues may relate to staff training or testing the application.

2. What kind of hardware will be needed for the project?

Consider asking this question to ascertain whether existing hardware may be used or whether extra hardware will be needed—for porting, testing, training, and support purposes if necessary.

Although this questionnaire is comprehensive, it should not be the only basis for scoping. Scoping should also include actual examination of application source code when pertinent. Software application documentation needs to be examined to learn more about the application from a customer usage and point of view.

Summary

Project managers and porting technical leads need to consider multiple aspects of a porting project before embarking on the actual port itself. Careful investigation of these aspects has led us to many successful porting projects in the past. Because every porting project is different, these aspects may take on different shapes and forms. Even so, the underlying concepts and descriptions will remain the same. A porting project can be summarized as follows:

- ▶ Scoping. Ask questions, inspect the code if available, ask more questions and investigate the risks, and develop the schedule.
- ▶ Analysis. Ask more questions, investigate the risks, and create the schedule based on technical and business issues.
- ▶ Porting. Set up the build environment, compile, and unit test.
- ▶ Testing. Set up the test environment, test, and analyze performance. Go back to porting process if bugs are found.

- ▶ Training. Train support and sales personnel on the application and operating environment.

Because complexities and risks associated with different technical aspects of the porting project may affect the overall schedule and budget, the transition point between each step of the porting process allows the project team to reassess its original estimates. Reassessing the original estimates helps set the proper expectations of all stakeholders early on before the real work of porting begins. As in all projects, the success of the project is not only defined by the successful delivery of the final product; it is also defined by how well the project was managed to the satisfaction of all parties involved. In the next chapters, we go through each step of the porting process. We present information about Linux that porting engineers can use as they go through the scoping, the analysis, the porting, and the testing phases of a porting project.



Example: System information gathering script

This appendix describes the use of two versions of a sample system diagnostic script. The scripts are designed to gather detailed system configuration information. One version is based on the ksh shell and works for Solaris systems. The other version is based on the bash shell and works on Linux systems. Therefore, when considering the two scripts together, you have a demonstration of how to port tools such as this (ksh to bash) from Solaris to Linux.

Both scripts are lengthy, so we do not present the source code in this appendix. We describe what they do and how to use them. You can download the script sources as part of the additional materials provided for this Redbook. See Appendix E, “Additional material” on page 401 for information about how to access this additional material.

The scripts are designed to gather the following groups of information:

- ▶ Basic system information:
 - Host name
 - OS release
 - Kernel information
 - Platform

- Number of CPUs
- Memory size
- And so on
- ▶ Kernel configuration information (*for Linux only*):
 - Current kernel runtime settings
 - Boot command line used
 - Device and driver files
 - Memory information and statistics
 - Kernel modules
 - And so on
- ▶ Operating system configuration information:
 - Boot loader type and configuration (*Linux only*)
 - Password and group files
 - System-wide profile setting
 - PAM configuration
 - Inittab file
 - RC files
 - Default configuration files for various software components
 - Syslog configuration
 - Print queues
 - Cron settings
 - Packages and patches
 - And so on
- ▶ Disk configuration information:
 - Disk usage information
 - Mount point ownership and permission settings
 - File system table file
 - Swap devices
 - Physical disk information
 - Disk partitions
 - If SVM or LVM is used, display configuration information

- ▶ Network configuration information:
 - NIC settings
 - Host files
 - Routers
 - Name resolver configuration
 - Services
 - NFS exports
 - Remote mounts
 - Automounter configuration
 - And so on
- ▶ System configuration information:
 - For Solaris: **prtdiag**, **prtconf**, **eeeprom**
 - For Red Hat: **hwconf** file
 - For Novell SUSE: **hwinfo** output
- ▶ Other software configurations:
 - SSHD server configuration file
 - SSH client configuration file
 - sudo configuration
 - Sample listing of processes that are currently running

Not all available software components are included in the sample scripts. However, these samples should cover most of the common system components about which you might want to gather information. If you have other software components that you use and want to monitor using a tool such as this, you can adapt these samples as necessary.

These scripts were originally designed for two purposes:

- ▶ As a snapshot tool for documenting the server configuration
- ▶ As a source of information for emergency purposes (system recovery)

The scripts are designed to provide output in two types of file formats:

- ▶ Plain text format
- ▶ HTML format

Key command line parameters for the script are:

- ▶ **h**, **H**, **html**, or **HTML**: Output in HTML format only

- ▶ **t, T, text, or TEXT:** Output in text format only
- ▶ **b, B, both, or BOTH:** Output in HTML and text format
- ▶ **-h or -?:** Display usage information

The default output format is text. The output location is the present working directory. These scripts need to be run with root authority in order to be able to access certain portions of the operating system information.

The output file naming convention works as follows:

```
<hostname>.sysinfo.YYYYMMDD.<output-type>
```

An example set of output files is:

- ▶ fred.sysinfo.20051116.html
- ▶ fred.sysinfo.20051116.txt
- ▶ bart.testlab.com.sysinfo.20051116.html
- ▶ bart.testlab.com.sysinfo.20051116.txt

After these output files have been generated, you might want to copy them to another system for safe keeping in the event that the system that generated the files needs to be recovered (using information saved in these files as a guide).



Additional material

This redbook refers to additional material that can be downloaded from the Internet as described here.

Locating the Web material

The Web material associated with this redbook is available in softcopy on the Internet from the IBM Redbooks Web server. Point your Web browser to:

<ftp://www.redbooks.ibm.com/redbooks/SG247186>

Alternatively, you can go to the IBM Redbooks Web site at:

ibm.com/redbooks

Select **Additional materials** and open the directory that corresponds with the redbook form number, SG247186.

Using the Web material

The additional Web material that accompanies this redbook includes the following file:

<i>File name</i>	<i>Description</i>
SG247186-getsysinfo.zip	ZIP file containing getsysinfo sample system diagnostic scripts.

How to use the Web material

Create a subdirectory (folder) on your workstation, and unzip the contents of the Web material ZIP file into this folder.

We provide detailed information about how to use the sample getsysinfo scripts in Appendix D, “Example: System information gathering script” on page 397.

Related publications

The publications listed in this section are considered particularly suitable for a more detailed discussion of the topics covered in this redbook.

IBM Redbooks

For information about ordering these publications, see “How to get IBM Redbooks” on page 411. Note that some of the documents referenced here may be available in softcopy only.

- ▶ *Accounting and Monitoring for z/VM Linux guest machines*, REDP-3818
- ▶ *Advanced POWER Virtualization on IBM System p5*, SG24-7940
- ▶ *IBM @server BladeCenter, Linux, and Open Source: Blueprint for e-business on demand*, SG24-7034
- ▶ *IBM @server xSeries and BladeCenter Server Management*, SG24-6495
- ▶ *IBM @server zSeries 890 Technical Introduction*, SG24-6310
- ▶ *IBM @server zSeries 900 Technical Guide*, SG24-5975
- ▶ *IBM @server zSeries 990 Technical Guide*, SG24-6947
- ▶ *IBM System z9 109 Technical Guide*, SG24-7124
- ▶ *IBM System z9 and @server zSeries Connectivity Handbook*, SG24-5444
- ▶ *Linux Client Migration Cookbook A Practical Planning and Implementation Guide for Migrating to Desktop Linux*, SG24-6380
- ▶ *Linux for IBM System z9 and zSeries*, SG24-6694
- ▶ *Linux for S/390*, SG24-4987
- ▶ *Linux Handbook A Guide to IBM Linux Solutions and Resources*, SG24-7000
- ▶ *Linux on IBM @server i5 Implementation Guide*, SG24-6388
- ▶ *Linux on IBM @server zSeries and S/390: Best Security Practices*, SG24-7023
- ▶ *Linux on IBM @server zSeries and S/390: Building SuSE SLES8 Systems under z/VM*, REDP-3687
- ▶ *Linux on IBM @server zSeries and S/390: Cloning Linux Images in z/VM*, REDP-0301
- ▶ *Linux on IBM @server zSeries and S/390: Distributions*, SG24-6264

- ▶ *Linux on IBM @server zSeries and S/390: ISP/ASP Solutions*, SG24-6299
- ▶ *Linux on IBM @server zSeries and S/390: Large Scale Linux Deployment*, SG24-6824
- ▶ *Linux on IBM @server zSeries and S/390: Performance Measurement and Tuning*, SG24-6926
- ▶ *Linux on IBM @server zSeries and S/390: System Management*, SG24-6820
- ▶ *Linux on IBM zSeries and S/390: High Availability for z/VM and Linux*, REDP-0220
- ▶ *Linux on IBM zSeries and S/390: Securing Linux for zSeries with a Central z/OS LDAP Server (RACF)*, REDP-0221
- ▶ *Linux on IBM zSeries and S/390: Server Consolidation with Linux for zSeries*, REDP-0222
- ▶ *Linux System Administration and Backup Tools for IBM @server xSeries and Netfinity*, SG24-6228
- ▶ *Linux with zSeries and ESS: Essentials*, SG24-7025
- ▶ *Networking Overview for Linux on zSeries*, REDP-3901
- ▶ *Printing with Linux on IBM @server zSeries Using CUPS and Samba*, REDP-3864
- ▶ *Technical Introduction: IBM @server zSeries 800*, SG24-6515
- ▶ *The IBM @server BladeCenter JS20*, SG24-6342
- ▶ *Tuning Red Hat Enterprise Linux on IBM @server xSeries Servers*, REDP-3861
- ▶ *Tuning SUSE LINUX Enterprise Server on IBM @server xSeries Servers*, REDP-3862
- ▶ *Up and Running with DB2 for Linux*, SG24-6899
- ▶ *z/VM and Linux on zSeries: From LPAR to Virtual Servers in Two Days*, SG24-6695

Other publications

These publications are also relevant as further information sources:

- ▶ Mendoza, et al., *UNIX to Linux Porting: A Comprehensive Reference*, Prentice Hall, 2006, ISBN 0131871099
- ▶ Johnson, S., *Performance Tuning for Linux Servers*, IBM Press, 2005, ISBN 013144753X

- ▶ Kuo, P. and N. Rahimi, *SUSE LINUX Enterprise Server 9 Administrator's Handbook*, Novell Press, 2005, ISBN 067232735X
- ▶ Linux Standard Base Team, *Building Applications with the Linux Standard Base*, IBM Press, 2004, ISBN 0131456954
- ▶ Nemeth, E. and G. Snyder, *Linux Administration Handbook* (2nd Edition), Prentice-Hall PTR, 2006, ISBN 0131480049
- ▶ Petersen, R., *Red Hat Enterprise Linux & Fedora Core 4: The Complete Reference*, McGraw-Hill Osborne Media, 3 edition, 2005, ISBN 0072261544
- ▶ Siever, E., et al., *Linux in a Nutshell*, O'Reilly Media, Inc., 5 Edition, 2005, ISBN 0596009305

Online resources

These Web sites and URLs are also relevant as further information sources:

- ▶ IBM developerWorks open source section
<http://www.ibm.com/developerworks/opensource>
- ▶ IBM Tivoli Workload Scheduler
<http://www.ibm.com/software/tivoli/sw-bycategory/>
- ▶ What's new in Performance Toolkit for VM in Version 5.1.0
<http://www.vm.ibm.com/library/gm130637.pdf>
- ▶ *IBM @server BladeCenter Management Module User's Guide*
<http://www.ibm.com/pc/support/site.wss/MIGR-45153.html>
- ▶ *Cisco Systems Intelligent Gigabit Ethernet Switch Module for the IBM @server BladeCenter Installation Guide*
<http://www.ibm.com/pc/support/site.wss/document.do?lndocid=MIGR-57858>
- ▶ *IBM @server BladeCenter Management Module Command-Line Interface Reference Guide*
<http://www.ibm.com/support/docview.wss?uid=psg1MIGR-54667>
- ▶ RMF PM for Linux
<http://www.ibm.com/servers/eserver/zseries/zos/rmf/rmfhtmls/pmweb/pmlin.html>
- ▶ z/VM Security and Integrity Resources
<http://www.vm.ibm.com/security>
- ▶ *Linux virtualization on POWER5: A hands-on setup guide*
<http://www.ibm.com/developerworks/edu/1-dw-linux-pow-virtual.html>

- ▶ IBM @server xSeries server platforms
<http://www.ibm.com/servers/eserver/xseries>
- ▶ IBM BladeCenter server platform
<http://www.ibm.com/servers/eserver/bladecenter>
- ▶ IBM LPAR Validation Tool
<http://www.ibm.com/servers/eserver/series/lpar/>
- ▶ IBM Systems Hardware Information Center
<http://publib.boulder.ibm.com/infocenter/eserver/v1r3s/index.jsp>
- ▶ Management Processor Command Line Interface (MPCLI)
<http://www.ibm.com/pc/support/site.wss/MIGR-54216.html>
- ▶ IBM Director
<http://www.ibm.com/pc/support/site.wss/MIGR-57057.html>
- ▶ z/VM publications
<http://www.vm.ibm.com/pubs/>
- ▶ IBM @server zSeries and IBM System z
<http://www.ibm.com/servers/eserver/zseries/>
- ▶ Linux Documentation Project, a collection of Linux documentation and “How-To” guides about almost every topic related to installing, configuring, or using Linux
<http://www.tldp.org>
- ▶ Red Hat Enterprise Linux documentation
<http://www.redhat.com/docs/>
- ▶ SUSE Linux Enterprise Server documentation
<http://www.novell.com/documentation/sles9/>
- ▶ SUSE Linux Enterprise Server Support
<http://www.novell.com/support/products/linuxenterpriseserver/>
- ▶ Red Hat Package Manager (RPM)
<http://www.rpm.org>
- ▶ SUSE Linux portal
<https://portal.suse.com>
- ▶ Red Hat Network (RHN)
<https://rhn.redhat.com>

- ▶ Common UNIX Printing System (CUPS) information
<http://www.cups.org>
- ▶ Printer compatibility and other printing information
<http://www.linuxprinting.org>
- ▶ Linux Assigned Names And Numbers Authority (LANANA), which maintains the Linux Device List
<http://www.lanana.org>
- ▶ Web-Based Enterprise Management (WBEM) and the Common Information Model (CIM)
<http://www.dmtf.org>
- ▶ Simple Network Management Protocol (SNMP)
<http://www.net-snmp.org>
- ▶ Linux Assigned Names And Numbers Authority (LANANA), which maintains the Linux Device List
<http://www.lanana.org>
- ▶ GNU General Public License
<http://www.gnu.org/copyleft/gpl.html>
- ▶ Linux Standard Base
<http://www.linuxbase.org>
- ▶ Free Software Foundation
<http://www.fsf.org>
- ▶ Open Source Initiative
<http://www.opensource.org>
- ▶ Knoppix
<http://www.knoppix.com>
- ▶ *Red Hat SELinux Guide*
<http://www.redhat.com/docs/manuals/enterprise/RHEL-4-Manual/selinux-guide/>
- ▶ NSA SELinux information page
<http://www.nsa.gov/selinux>
- ▶ GRand Unified Bootloader
<http://www.gnu.org/software/grub>
- ▶ SYSLINUX
<http://syslinux.zytor.com>

- ▶ SUSE Linux Portal: How to Setup/Use Multipathing on SLES
http://portal.suse.com/sdb/en/2005/04/sles_multipathing.html
- ▶ Red Hat device-mapper-multipath enhancement update
<https://rhn.redhat.com/errata/RHEA-2005-280.html>
- ▶ Filesystem Hierarchy Standard
<http://www.pathname.com/fhs/>
- ▶ Linux LVM How-To
http://www.tldp.org/HOWTO/html_single/LVM-HOWTO/
- ▶ VERITAS Software (now Symantec)
<http://www.veritas.com>
- ▶ Information about the **cdrecord** command
<http://freshmeat.net/projects/cdrecord/>
- ▶ Mochel, P., "The sysfs file system" paper, presented in Volume 1 of the 2005 Ottawa Linux Symposium proceedings
<http://www.linuxsymposium.org/2005/>
- ▶ Quagga Routing Suite
<http://www.quagga.net>
- ▶ Distributed Management Task Force (DMTF)
<http://www.dmtf.org>
- ▶ Internet Engineering Task Force (IETF)
<http://www.ietf.org>
- ▶ Web-Based Enterprise Management (WBEM)
<http://www.dmtf.org/standards/wbem>
- ▶ OpenPegasus CIM Server
<http://www.openpegasus.org>
- ▶ OpenWBEM
<http://www.openwbem.org>
- ▶ Net-SNMP package
<http://www.net-snmp.org/>
- ▶ SNMP tutorials
<http://www.simpleweb.org/>
- ▶ Common UNIX Printing System (CUPS)
<http://www.cups.org/>

- ▶ LinuxPrinting.org Web site
<http://www.linuxprinting.org>
- ▶ OpenLDAP
<http://www.OpenLDAP.org/>
- ▶ “Rsync snapshot” article
<http://www.mikerubel.org>
- ▶ Advanced Maryland Automated Network Disk Archiver (AMANDA)
<http://www.amanda.org/>
- ▶ Sun Security Resources
<http://sunsolve.sun.com/pub-cgi/show.pl?target=security/sec>
- ▶ Red Hat Support Resources
<https://www.redhat.com/apps/support/>
- ▶ SUSE Online Security Support
<http://www.novell.com/linux/security/securitysupport.html>
- ▶ Yet Another Solaris Security Package (YASSP)
<http://www.yassp.org/>
- ▶ Bastille Linux
<http://www.bastille-linux.org>
- ▶ Security Enhanced Linux (SE-Linux)
<http://www.nsa.gov/selinux/>
- ▶ Very Secure FTP Daemon, or vsftpd
<http://www.vsftpdrocks.org>
- ▶ vsftpd FTP server
<http://vsftpd.beasts.org>
- ▶ *UNIX IP Stack Tuning Guide*
<http://www.cymru.com/Documents/ip-stack-tuning.html>
- ▶ OpenSSH
<http://www.openssh.com>
- ▶ Using DVDs with Linux
<http://dvd.sourceforge.net>
- ▶ Linux hotplugging
<http://linux-hotplug.sourceforge.net>

- ▶ Submount: Linux Removable Media Handling System
<http://submount.sourceforge.net>
- ▶ K3b application
<http://k3b.sourceforge.net>
- ▶ Linux Channel Bonding
<http://sourceforge.net/projects/bonding>
- ▶ Linux Class-based Kernel Resource Management
<http://ckrm.sourceforge.net/>
- ▶ Standards Based Linux Instrumentation for Manageability (SBLIM) project
<http://sblim.sourceforge.net>
- ▶ OProfile profiling suite
<http://oprofile.sourceforge.net>
- ▶ High Availability Linux project
<http://linux-ha.org>
- ▶ Open Cluster Framework (OCF)
<http://opencf.org>
- ▶ The Bash shell
<http://www.gnu.org/software/bash/>
- ▶ Linux Kernel Crash Dump utility
<http://lkcd.sourceforge.net>
- ▶ Memtest86 - A Stand-alone Memory Diagnostic
<http://www.memtest86.com>
- ▶ Latest BIOS and firmware code
<http://www.ibm.com/pc/support/site.wss/document.do?lnocid=MIGR-4JTS2T>
- ▶ Driver matrixes
<http://www.ibm.com/pc/support/site.wss/DRV-MATRIX.html>
- ▶ YaBoot
<http://penguinppc.org/bootloaders/yaboot/>
- ▶ Linux for S/390 project
<http://www.Linuxvm.org/>

How to get IBM Redbooks

You can search for, view, or download Redbooks, Redpapers, Hints and Tips, draft publications and Additional materials, as well as order hardcopy Redbooks or CD-ROMs, at this Web site:

ibm.com/redbooks

Help from IBM

IBM Support and downloads

ibm.com/support

IBM Global Services

ibm.com/services

Index

Symbols

#cp 327, 336
\$cp 336
/bin/ash.static 212
/dev/hvcs0 299
/etc/auto.master 63
/etc/auto_master 63
/etc/ftpd/ftpusers 192
/etc/ftpusers 192
/etc/group 160
/etc/inetd.conf 189
/etc/inittab 133
/etc/issue 202
/etc/ldap.conf 164
/etc/motd 202
/etc/nsswitch.conf 164
/etc/printcap 155
/etc/racoon/racoon.conf 199
/etc/securetty 31, 196
/etc/skel/* 163
/etc/sysctl.conf 192
/etc/xinetd.d/ 190
/home 163
/lib/libipsec.so 198
/media/cdrom 94
/media/floppy 94
/proc 57, 89
/proc/mdstat 227
/sbin/init 133
/sbin/racoon 198
/sbin/setkey 198
/sys 90

Numerics

8 mm tape 99

A

accept (print jobs) 154
access control list 60
access control list (ACL) 194
acl 60
addbadsec 55

adding a device 92
Advanced Intelligent Tape 99
Advanced Maryland Automated Network Disk Archiver 185
advanced system installation 29
Advanced System Management (ASM) 243
Advanced System Management Interface 259
agetty 104
AIT 99
AMANDA 185
Apache Web Server 5
arp 231
ASET 189
ASMI 259
AutoFS 62
AutoFSCK 62
automated restart 339
automated software management 81
automount 62–63
automountd 62–63
autoyast 37

B

backup 182
badblocks 225
Baseboard Management Controller 239
bash 212
bash parameters 335
basic system installation 24
Bastille Linux 188
best practice 309, 311, 321, 331, 333
bgpd 115
BIOS 127, 238
blade 239
BladeCenter 237, 244, 247
BladeCenter boot sequence 272
BladeCenter LAN Switch I/O Modules 268
BladeCenter Management Module 239, 244, 264
 configuring 265
BladeCenter naming 271
block device 89
BMC 240
bonding 117

- boot command 277, 326
- boot configuration files 132
- boot loader 128
- boot loaders 126
- boot source 127
 - cdrom / dvd 127
 - floppy and various USB devices 127
 - hard disk 127
 - network 127
- boot troubleshooting 216
- booting 126
 - default boot 126
 - emergency boot 127
 - interactive boot 126
 - interactive start of services 127
 - reconfiguration boot 126
 - recovery boot 127
 - single user 126
 - verbose boot 126
- Bootparams 136
- br0 298
- brctl 298
- bridge-down.sh 297
- bridge-up.sh 297
- bridge-utils 296–297
- bridging 117
- BSM 189
- bunzip2 182
- business intelligence 16
- bzcat 182
- bzip2 182
- bzip2recover 182
- bzless 183

C

- CacheFS 56–57
- cancel (print jobs) 154
- Capacity on Demand 249
- CD 96
- cda 98
- cdda2wav 98
- cdparanoia 98
- cdrecord 97
- CD-ROM 94
 - bootable 18
- cdrw 97
- central processing units (CPUs) 312
- central processor (CP) 312
- central storage 313
- cfgadm 93
- chacl 194
- channel 313, 319
- channel path 314
- channel-path identifier (CHPID) 314
- character device 88
- chccwdev 336
- chgrp 223
- chmod 223
- chown 223
- CHPID 314, 318
- chroot 230
- CIM object manager 147
- CIM server 147
- Cisco Systems Intelligent Gigabit Ethernet Switch Module 271, 274
- CKRM 141
- Class-based Kernel Resource Management 141
- client partition profile 291
 - configuring Linux 302
- cluster 206
- cmsfscat 336
- cmsfsck 336
- cmsfscp 336
- cmsfslst 336
- cmsfsvol 336
- commercial distributions 6
- Common UNIX Printing System (CUPS) 152
- Companion CD 183
- compiling patches 82
- compress 182
- compression tools 182
- console command 277
- console for the JS20 blade 273
- Control Program (CP) 320
- control unit 313
- core files 216
- cpint 336
- cpio 182
- crash 219
- crash dumps 218
- cryptographic hardware 341
- csh 212
- cupsd 155

D

- DASD 310, 334

- DASD Dump/Restore (DDR) Facility 340
- dasdfmt 337
- dasdview 337
- DAT 99
- database serving 15
- dbginfo.sh 335, 337
- dd 182
- Debian 6
- debugging tools 335
- Dedicated Service Tools 251
- dependencies 79
- dependency 80
- developerWorks 83
- devfsadm 52
- device access 89
- device configuration information 89
- device drivers 92
 - install 92
- DFSMSdss 340
- DHCP 119, 136
 - Linux 119
 - /etc/dhcpd.conf 119
 - /etc/init.d/dhcpd 119
 - /etc/sysconfig/dhcpd 119
 - /var/lib/dhcpd.leases 119
 - Solaris 119
 - /etc/inet/dhcpsvc.conf 119
 - dhcpconfig 119
 - dhcpcmgr 119
 - dhtadm 119
 - in.dhcpd 119
 - ptadm 119
- dhcpconfig 119
- dhcpcmgr 119
- dhtadm 119
- diagela 303
- dig 231
- Digital Audio Tape 99
- Digital Linear Tape 99
- disable (printers) 154
- disabling rc scripts 134
- disk drives 315
- disk partition 59
- disk partitioning 53
- disk recognition 52
- disk slice 59
- Disk-Based File System Management 55
- diskdump 219
- diskdumpctl 219

- diskdumpfmt 219
- diskettes 94
- disks 52
- disks and disk partitioning 52
- displaying tape drive status 101
- Distributed Data Server (DDS) 331–332
- Distributed Management Task Force 147
- distribution 4
- distributions 5
- DLT 99
- dmesg 91, 221, 232
- DMTF 147
- DNS 120
- DST 251
- dump 183, 185
- dumpadm 218
- DVD 94, 96
- dvd+rw-tools 97
- dvdrecord 98
- dynamic logical partitioning 253

E

- e2fsck 55–56
- eject 96
- enable (printers) 154
- Enterprise Resource Planning (ERP) 15
- env -T command 277
- Epson 157
- ESC/P printers 157
- ESCON 314
- ethereal 231
- ethtool 231
- expanded storage 313
- ext2 54, 62
- ext3 55–56, 59, 62
- extended user attributes 60

F

- FCON/ESA 332
- FCP 315
- fdasd 337
- fdformat 96
- fdisk 53–55
- Fibre Channel Protocol (FCP) 315
- FICON 314
- file system dump and restore 183
- file system journaling 59
- file system snapshot 185

- file systems 178
- file/print infrastructure 14
- Filesystem Hierarchy Standard 60
- find 222
- firewall 203
 - chains 203
 - policies 203
- firewalls 202
- firmware 238
- floppy diskette 96
- fmthard 53
- format 53
- free 332, 335
- free command 59
- Free Software Foundation 5
- fsck 55–56
 - fsck -f 225
 - fsck -y 225
 - fsck.ext3 55–56
 - fsck.reiserfs 56
- fssnap 184–185
- FTP 191
- fuser 232

G

- gdb 233
- General Public License (GPL) 4–5
- getfacl 194
- gfloppy 96
- gfxmenu 30
- GID 165
- GNOME desktop 5
- GNU 4–5
- group administration 160
- growfs 66–67
- growisofs 97
- GRUB 126
 - config file 128
- guest LAN 317
- guest operating system 320
- gunzip 182
- gzcat 182
- gzexe 182
- gzgrep 182
- gzip 182
- gznew 182

H

- halt 135
- Hardware Management Console 249
- hcp 327–329, 337
- Heartbeat 207, 340
- Helsinki University of Technology 197
- high availability 206, 338
- HiperSockets 316, 341
- HMC 249
- hosted installation 248
- hot-plug 52, 93, 328
- hotplug command 90, 93
- hot-plugging 93
- HP printers 157
- HSFS 55
- hsfs 56
- hvcs 299
- hwbrowser 92, 233
- hwconf 91
- hwinfo 92
- Hypervisor Virtual Console 299

I

- IA32 238
- IBM Director 240
- IBM Machine Reported Product Data database 303
- IBM VIO Server 285
- IBM virtual SCSI server module 300
- ibmvscsis 300
- IDE 52
- IEEE POSIX 10
- IEEE POSIX 1003.1-2001 7
- ifconfig 111, 335
- ifenslave 117
- IFL 310, 312
- IKE 198
- imap4 122
- inetd 117
 - /etc/default/inetd 118
 - /etc/inetd.conf 118
 - /etc/services 118
 - ENABLE_CONNECTION_LOGGING 118
 - ENABLE_TCPWRAPPERS 118
- info 8
- init 135
- initrd 128
- inittab 102
- insmod 86

- install
 - automation 29
 - Linux client from Solaris server 47
 - PXE 47
 - Red Hat Kickstart 32
 - remote display 31
 - serial console 29
 - software bundles 26
 - Solaris client from Linux server 50
 - Solaris Jumpstart 32
 - SUSE AutoYaST 37
 - vnc 31
- installation automation 29
- installation methods 320
- Integrated Virtualization Manager 249
- Internet Engineering Task Force 147
- Internet Printing Protocol (IPP) 152
- inter-user communication vehicle 317
- iostat 232, 332, 335
- ip 231
- IP masquerading 202
- IP redirects 193
- IP routing 114
 - Linux 115
 - /etc/sysctl.conf 115
 - bgpd 115
 - net.ipv4.ip_forward 115
 - ospf6d 115
 - ospfd 115
 - quagga 115
 - radvd 115
 - ripd 115
 - ripngd 115
 - Solaris 114
 - /etc/defaultrouter 115
 - /etc/gateways 115
 - /etc/notrouter 115
 - /usr/sbin/in.rdisc 115
 - ndpd 115
 - RDISC 114
 - RIP 114
 - ripngd 115
- IP stateful firewalling 123
- ipcs 232
- iPlanet Directory Server 163
- IPSEC 198
- IPSec 198–199
- IPSec and IKE 116
- iptables 203, 341
- IPv4 108
 - Linux 109
 - /etc/bootparams 109
 - /etc/hosts 109
 - /etc/nsswitch.conf 109
 - /etc/protocols 110
 - /etc/resolv.conf 109
 - /etc/services 110
 - ifcfg-interface-id-mac-address 109
 - ifcfg-interface-name 109
 - ifconfig 111
 - network profiles 110
 - sysctl 110
 - Solaris 108
 - /etc/bootparams 108
 - /etc/defaultdomain 108
 - /etc/defaultrouter 108
 - /etc/dhcp.interface 108
 - /etc/ethers 108
 - /etc/hostname.interface 108
 - /etc/inet/hosts 108
 - /etc/inet/protocols 108
 - /etc/inet/services 108
 - /etc/netmasks 108
 - /etc/nodename 108
 - /etc/nsswitch.conf 108
 - /etc/resolv.conf 108
 - dhcpcagent 108
 - ifconfig 111
 - ndd 110
- IPv6
 - Linux 113
 - ifcfg-interface-id-mac-address 114
 - ifcfg-interface-name 113
 - Solaris 113
 - /etc/hostname6.interface 113
 - /etc/inet/ipnodes 113
- iso9660 56
- ITEF 147
- IUCV 317

J

- Jaz drive 94
- JetDirect 153
- journal mode 60
- journaling 59
- JS20 247
- jumpstart 32

K

K3b 97
KDE desktop 5
Kerberos 199
kernel 4–5, 128
kernel.core_pattern 217
kernel.core_uses_pid 217
kickstart 32
kill 232
Knoppix 18
ksh 212
ksymoops 233
kudzu 105

L

LANANA 88
lcrash 220
LDAP 120, 163
ldapadd 164
ldapcompare 164
ldapdelete 164
ldapmodify 164
ldapmodrdn 164
ldappasswd 164
ldapsearch 164
ldapwhoami 164
ldd 232
Lexmark 157
librtas 303
LILO 126
Linus Torvalds 4–5
Linux Assigned Names And Numbers Authority 88
Linux Device List 88
Linux on POWER installation 278
 Novell SUSE 280
 Red Hat 279
Linux Standard Base (LSB) 9
Linux Virtual Server (LVS) 340
Linux-HA 206
live CD 18
LKCD 219
LOADLIN 126
locally attached printers 153
locate 222
logadm 221
logging 193
logical disk devices
 Linux 87

 Solaris 87
logical partition 54
logical partitioning 248
logical volume 68
logical volume groups 177
logical volume management 66
logical volumes 177
logs 220
LOM 30
loopback 57, 339
lp 154
lpadmin 154
LPAR 248, 309, 311, 318–319, 333, 341
lpc 154
LPD 152
lpfilter 154
lpforms 154
lpget 154
lpinfo 155
lpmove 154
lpoptions 155
lppasswd 155
lpq 154
lpr 154
lprm 154
lpsched 154
lpset 154
lpshut 154
lpstat 154
lpsystem 154
lpusers 154
lscss 337
lsdasd 337
lsmod 86
lsnf 232
lspci 91, 233
lsqeth 337
lsraid 67
lstape 337
lsusb 91
lsvpd 303
ltrace 232, 335
LUN number 52
lvchange 65, 67
lvcreate 65–66
lvdisplay 67, 72
lvextend 67
LVM 54
LVM2 66, 68

lvm2 66
lvmdiskscan 68
lvreduce 65–66
lvremove 65
lvscan 68

M

mail server 15
mail services 122
major numbers 87
MAKEDEV 86
man 8
Mandrake 6
Master Boot Record (MBR) 54
MBR 127
mdadm 65–67
mdb 217–218
memtest86 233
messaging 15
metaclear 65, 67
metadb 67
metadetach 65–67
metadevadm 67
metahs 67
metainit 65, 67
metaoffline 67
metaparam 67
metarecover 67
metarename 67
metareplace 67
metaroot 67
metaset 67
metastat 67
metasync 67
metattach 65, 67
mgetty 104
MIB 149
microcode 303, 312, 338–339
Micro-Partitioning 252
migration planning 11
mii-tool 231
mingetty 104
MINIX 4
minor number 87
mk2fs 55
mkdump 337
mke2fs 56
mkfs 55–56

mkfs.ext3 55–56
mkfs.reiserfs 55–56
mkinitrd 329
mkisofs 97
mkraid 65
mkreiserfs 56
mkswap 58–59
modem 103
modinfo 86, 93
modload 86
modprobe 93
module stacking 196
modunload 86
monitoring 175
monolithic installation 248
mount 57
mpstat 332
msdos 56
mt 101
mtools 96
mtx 101
multichip module (MCM) 312
multipath devices 53
mzip 96

N

NAT masquerading 202
National Security Agency (NSA) 189
Nautilus File Manage 95
ndd 192
ndpd 115
netdump 219
netdump-server 219
Netfilter 202
net-snmp 150
netstat 231, 335
NetWare 153
network boot services 136
network booting 135
Network File System (NFS) 58, 62
Network File Systems 58
network printing 153
network troubleshooting 230
network trunking 116
newfs 55
newsyslog 221
NFS 57, 121, 137
NIS 121

NIS (and NIS+) 164
NIS and NIS+ 121
NIS+ 121
non-commercial distributions 6
Novell 6
nslookup 231
NSS (name switch service) 164
NTP 120

O

OpenFirmware 283
OpenLDAP 163
OpenPegasus 148
OpenPROM 126
OpenSSH 198
OpenWBEM 149
operating system 4
opreport 233
OSA-2 317
OSA-Express 317
ospf6d 115
ospfd 115

P

pack 182
package 76
package distribution 80
package management 76
Packages 76
paging 313
PAM module types 195
pam_login.so 197
pam_nologin.so 197
pam_securetty.so 196
parted 53–55
partition 53
partitions 52
passwd 230
patch 78
Patch activation 82
patch cluster 80
Patch Manager 81
patchadd 79
patches 78
patching 79
PatchPro Expert 81
PatchPro Interactive 81
patchrm 79

pax 182
pcat 182
PCFS 55
pcfs 56
PCL printers 157
pcsf 56
pdksh 213
performance 178
Performance Toolkit for VM 331, 335
pgrep 232
ping 231, 335
pkgadd 76
pkgchk 76
pkginfo 76
pkgparam 76
pkgrm 76
pkill 232
Pluggable Authentication Modules (PAM) 194
pmadm 105
pntadm 119
pop3 122
port initialization 104
portmap 226
POSIX threads 7
POST 128
postfix 122
PostScript printing 157
power command 277
poweroff 135
ppc64-utils 303
PPD (PostScript Printer Description) 157
PReP boot partition 279, 281
primary partition 54
print server 319
printer 319
printer types 156
printing subsystem 152
process accounting 146
processes 179
processing units (PU) 312
proof of concept 331
proof of technology 331
prtvtoc 53
ps 231, 335
Public Domain Korn shell 213
pvcreate 65, 67
pvdisplay 68, 71
pvs 69
pvscan 69

PXE 47

Q

QDIO 310
qetharp 337
qethconf 337
quorum 68
quota 145

R

racocon keying daemon 198
radvd 115
RAID 54, 66, 68
raidhotadd 67, 227
raidhotremove 67
raidreconf 66–67
raidstart 65
raidstop 65
raidtools 66
ramfs 58
RARP 136
raw device 89
raw disk slices 52
RDISC 114
reboot 135
Red Hat 6, 198
Red Hat Network (RHN) 80–81
Redbooks Web site 411
 Contact us xix
ReiserFS 60
Reiserfs 54
reiserfs 55–56, 59
reiserfsck 56
reject (print jobs) 154
remote display installation 31
Remote Supervisor Adapter II 239
remote system management 147
removable media 94
 access 95
 formatting 96
 supported types 94
reset command 277
resize_reiserfs 66–67
resize2fs 66–67
resources 139
restore 183
REXX 340
RHEL install boot options 280

RIP 114
ripd 115
ripngd 115
rksh 212
rmformat 96
rmmod 86
Road Warrior server 199
 IPSEC 199
root password recovery 229
route 231, 335
rpcbind 226
rpm 76
RSA II 241
rsh 212
run control files 133
run levels 131
 0 132
 1 132
 2 132
 3 132
 4 132
 5 132
 6 132
 emergency 132
 single user 132

S

S/390 336–337
s390-tools 336
s390-utils 336
s390x 337
SAN 52
sar 232, 332
savecore 218–219
scheduling 144
scp 341
SCSI 52, 316
sd.conf 52
SDS 65
Secure Shell (SSH) 197
securing inetd/xinetd 189
Security Enhanced Linux (SE-Linux) 189
security/directory infrastructure 15
sendmail 122
serial console 104
serial interface as console device 30
serial interface as input/output device for GRUB 30
serial interface as input/output device for install 29

- Serial over LAN 273
 - configuring 274
 - using 276
- serial port devices 102
- Serial Ports Tool 102
- ServeRAID 209
- setfacl 194
- setserial command 103
- setup 8
- sftp 341
- sh 212
- shared processing 252, 284
- showrev 79
- shutdown 135
- shutting down 135
- sigar 233, 335
- Simple Network Management Protocol 147
- Single UNIX Specification (SUS) 9
- SLES 199
- SLES install boot options 282
 - AddSwap 283
 - Gateway 282
 - HostIP 282
 - Insmode 283
 - Install 282
 - Netdevice 282
 - Proxy 282
 - ProxyPort 283
 - SSHPasswd 283
 - Textmode 283
 - UseSSH 283
 - VNC 283
 - VNCPassword 283
- slice 53
- slices 52
- Small Computer System Interface 316
- SMB 185
- SMS 261
- snapshots 184
- SNMP 147
- snoop 231
- soft partitions 68
- software bundles 26
- software RAID 176
- SoL 273
- Solaris
 - Volume Manager 65
- Solaris Volume Manager (SVM) 65
- Solstice DiskSuite (SDS) 65
- Solstice DiskSuite / Solaris Volume Manager to Linux LVM 65
- Solution Assurance Review 309
- sound-juicer 98
- splashimage 30
- squid 122
- SSH 197
- SSH1 197
- SSH2 198
- SST 251
- startup scripts 142
- stat 223
- statistics 173
- statserial 233
- strace 232–233, 335
- subchannel 314
- subfs 95
- submount file system 95
- SunScreen 202
- SunSolve 80
- Super Digital Linear Tape 99
- Super DLT 99
- superuser access 196
- support 10
- SUSE 6
- SUSE Linux portal 80
- SVM 59, 65, 68
- swap 58–59
- swapoff 58–59
- swapon 58–59
- SYN attacks 192
- sysctl 192
- sysfs 90
- syslinux 47
- syslogd 221
- sysreport 233, 335
- system information 140
- System Management Services 261
- System Service Tools 251
- system services 141
- system-config-netboot 136

T

- tables 203
- tape device name format 99
- tape drives 98, 315
- tape390_display 337
- tapeinfo command 101

tar 182–183
tar and cpio 182
Tatu Ylönen 197
TCP wrappers 123, 190
tcpdump 231
tcsh 212
telinit 105, 135
TFTP 136
TMPFS 57
tmpfs 58
top 231–232, 332, 335
traceroute 335
Travan 98
trunking 117
truss 232
ttymon 104
ttyS0 104
tune2fs 62
tunedasd 337

U

udev 52, 87
udevd 52
UDF 55
udf 56
UFS 56, 59
ufs 56
UFS snapshot 184
ufsdump 183
UID 165
ulimit 232
uname 92
unpack 182
up2date 80–81
updatedb 222
user administration 160
user_xattr 60
useradd 160
userdel 160
usermod 160
usfrestore 183

V

VCTC 316
VDISK 313, 333
VERITAS 73
VERITAS VxVM 73
VERITAS VxVM and VxFS 73

Very Secure FTP (vsftp) 191
vfat 56
vgcfgbackup 67
vgcfgrestore 67
vgchange 65, 67
vgck 69
vgcreate 65
vgdisplay 67, 69
vgexport 69
vgextend 65, 67
vgimport 69
vgreduce 65
vgremove 65
vgscan 69
VIPA 339
virtual channel-to-channel 316
virtual consoles 284
virtual Ethernet 284
virtual file system 57
virtual file systems 56–57
virtual I/O 253
Virtual I/O Server 250
virtual I/O server partition profile 285
 configuring Linux 295
virtual machine 320
virtual memory 175
virtual network bridging 296
Virtual Partition Manager 251
virtual SCSI 284
virtual terminals 102
virtualization 284
vmconvert 337
vmstat 232, 332, 335
vold 95
volume group 68
volumes
 Solaris Volume Manager 65
VPN 116
vsftpd 191
VSWITCH 317

W

warning banners 202
WBEM 147
Web and application serving 15
Web proxy and cache servers 122
Web Proxy Server 122
Web-Based Enterprise Management 147

WholeDisk slice 53
writeback 60
writeback mode 60
wu-ftp 191

X

X Window System 7
xedit 325
xinet
 /etc/xinetd.conf 118
 /etc/xinetd.d/ 118
xinetd 117–118
xmcd 98
xSeries server 237

Y

YaBoot 283
yaboot.conf 283
YASSP (Yet Another Solaris Security Package)
188
YaST 8
YaST online update 81
YaST2 8
yast2 105
yast2-s390 336
you 80–81

Z

zgetdump 337
zip 182
zip drive 94
zipcloak 182
ZIPL 327
zipl 330, 337
zipnote 182
zipsplit 182
zsh 212



Redbooks

Solaris to Linux Migration: A Guide for System Administrators

(0.5" spine)
0.475" <-> 0.875"
250 <-> 459 pages



Redbooks

Solaris to Linux Migration: A Guide for System Administrators

A comprehensive reference for a quick transition

Presents a task-based grouping of differences between the operating system environments

Additional content about how to optimize Linux on IBM @server platforms

The goal of this IBM Redbook is to provide a technical reference for IT systems administrators in organizations that are considering a migration from Solaris to Linux-based systems. We present a systems administrator view of the technical differences and methods necessary to complete a successful migration to Linux-based systems, including coverage of how those differences translate to two major Linux distributions: Red Hat Enterprise Linux and SUSE Linux Enterprise Server.

The book is designed primarily to be a reference work for the experienced Solaris 8 or 9 system administrator who will need to begin working with Linux. It should serve as a guide for system administrators that need a concise technical reference for facilitating the transition to Linux.

The book also provides details about how to leverage the additional industry-leading technologies in IBM @server xSeries servers, IBM POWER technology-based systems (iSeries/pSeries), and IBM @server zSeries systems that make them very powerful and flexible platforms for hosting Linux-based solutions.

**INTERNATIONAL
TECHNICAL
SUPPORT
ORGANIZATION**

**BUILDING TECHNICAL
INFORMATION BASED ON
PRACTICAL EXPERIENCE**

IBM Redbooks are developed by the IBM International Technical Support Organization. Experts from IBM, Customers and Partners from around the world create timely technical information based on realistic scenarios. Specific recommendations are provided to help you implement IT solutions more effectively in your environment.

**For more information:
ibm.com/redbooks**