IBM

Advanced POWER Virtualization on IBM @server p5 Servers: Introduction and Basic Configuration

- Includes basic configuration of clients, Virtual I/O Server, and PLM
- Useful worksheets included for documentation and reference
- AIX partition, Virtual I/O Server, and HMC command examples

Bill Adra
Annika Blank
Mariusz Gieparda
Joachim Haust
Oliver Stadler
Doug Szerdi

Redbooks

ibm.com/redbooks

International Technical Support Organization

Advanced POWER Virtualization on IBM @server p5 Servers:

Introduction and Basic Configuration

October 2004

SG24-7940-00

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

First Edition (October 2004)

This edition applies to IBM AIX Version 5.3 (Build 0427D), HMC Version 4, Release 2.0, Build 20040628.1, Hardware type 9111-520, firmware level SF220_006.

Note: This book is based on a pre-GA version of a product and may not apply when the product becomes generally available. We recommend that you consult the product documentation or follow-on versions of this redbook for more current information.

Contents

Figures

Tables

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.

Any performance data contained herein was determined in a controlled environment. Therefore, the results obtained in other operating environments may vary significantly. Some measurements may have been made on development-level systems and there is no guarantee that these measurements will be the same on generally available systems. Furthermore, some measurement may have been estimated through extrapolation. Actual results may vary. Users of this document should verify the applicable data for their specific environment.

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:

@server®	Enterprise Storage Server®	PR/SM™
@server®	HiperSockets™	PTX®
Redbooks (logo)  ™	Hypervisor™	QMF™
ibm.com®	HACMP™	Redbooks™
iSeries™	IBM®	RDN™
i5/OS™	Micro Channel®	RS/6000®
pSeries®	Micro-Partitioning™	Tivoli®
zSeries®	PowerPC®	TotalStorage®
AFS®	POWER™	Versatile Storage Server™
AIX 5L™	POWER2™	Virtualization Engine™
AIX®	POWER4™	
DFS™	POWER5™	

The following terms are trademarks of other companies:

Java and all Java-based trademarks and logos are trademarks or registered trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.

Microsoft and the Windows logo are trademarks of Microsoft Corporation 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, and service names may be trademarks or service marks of others.

Preface

This IBM® Redbook provides an introduction to Advanced POWER™ Virtualization on IBM eServer p5 servers.

The Advanced POWER Virtualization feature is a combination of hardware and software that supports and manages the virtual I/O environment on POWER5™ systems. The main technologies are:

- Virtual Ethernet
- Shared Ethernet Adapter
- Virtual SCSI Server
- Micro-Partitioning™ technology
- Partition Load Manager

The primary benefit of Advanced POWER Virtualization is to increase overall utilization of system resources by allowing only the required amount of processor and I/O resource needed by each partition to be used.

This redbook is also designed to be used as a reference tool for System Administrators who manage IBM eServer p5 servers. It provides detailed instructions for:

- Configuring and creating partitions using the HMC
- Installing and configuring the Virtual I/O Server
- Creating virtual resources for partitions
- Installing partitions with virtual resources

While the discussion in this publication is focused on p5 hardware and the AIX® 5L™ operating system, the basic concepts can be extended to the i5/OS™ and Linux® operating systems as well as the i5 platform.

A basic understanding of logical partitioning is required.

The team that wrote this redbook

This redbook was produced by a team of specialists from around the world working at the International Technical Support Organization, Austin Center.

Bill Adra is an IT specialist in the IBM Systems and Technology group (STG) in Sydney, Australia with over five years of experience in IBM pSeries® and TotalStorage® solutions. He provides pre-sales technical support and post-sales implementation to IBM Business Partners and customers across Australia and New Zealand. Bill is a Certified Specialist for pSeries Solutions Sales and an IBM Certified pSeries AIX Systems Administrator.

Annika Blank is an IT specialist for pre-sales technical support in the IBM System Sales organization group in Hamburg, Germany. She has five years of experience in AIX, RS/6000® and @Server pSeries. She has worked at IBM for 11 years. Her areas of expertise include Web technologies on pSeries and high-end server systems, supporting IBM sales, IBM Business Partners, and customers with pre-sales consultation and implementation of client/server environments.

Mariusz Gieparda is a System Analyst for ComputerLand S.A., an IBM Business Partner in Poland. He has five years of experience in RS/6000, AIX, HACMP™, and more than ten years of experience in IT. His areas of expertise include UNIX®, SAP Basis, and system security. He is an IBM Certified Advanced Technical Expert - RS/6000 AIX and a Microsoft® Certified System Engineer.

Joachim Haust is an IT specialist for T-Systems in Germany. He has fifteen years of experience in UNIX. For the last five years he has specialized in planning, installing, and managing AIX environments, including HACMP for SAP. T-Systems - in addition to T-Mobile, T-Online and T-Com - is a division of Deutsche Telekom AG and is one of the leading information and communication (ICT) service providers in Europe, with approximately 41,000 employees in more than 20 countries.

Oliver Stadler is an Advisory IT specialist working for IBM Global Services in Switzerland. Overall he has 15 years of experience in IT, mainly focusing on systems management and system administration. His areas of expertise include Tivoli®-based system management solutions and zOS system programming. He has extensive programming experience using scripting languages like Shell, Perl and REXX. For the last two and half years he has worked for IGS Strategic Outsourcing as an AIX system engineer designing and implementing pSeries-based environments for customers. Oliver is a certified specialist for pSeries system administration and HACMP.

Doug Szerdi is a Senior Technical Staff Member at IBM Systems and Technology Group in Austin, TX. He has been with IBM for 22 years, working in various hardware development and system test roles across IBM pSeries, iSeries™, and zSeries® products. Doug is currently a lead engineer on the @server Global System Integration team, working on delivery of the POWER5 product family.

The project that produced this publication was managed by:

Scott Vetter
IBM Austin

Thanks to the following people for their contributions to this project:

Luke Browning, Bob Kovacs, Vinit Jain, Joel Tendler, Rafael Nogueras, Jessie Haug, Michael Cyr, Sertac Cakici, Geoff Gilbert
IBM Austin

Naresh Nayar
IBM Rochester

Matt Trzyna
IBM Raleigh

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 Internet note to:

  redbook@us.ibm.com

- Mail your comments to:

  IBM Corporation, International Technical Support Organization
  Dept. JN9B Building 905 Internal Zip 9053D004
  11501 Burnet Road
  Austin, Texas 78758-3493

1

Introduction

Enterprises continue to look for ways to overcome new and existing challenges. Whether driving to increase productivity, developing new strategies, looking for ways to innovate, or striving to create more flexibility and efficiency in their IT infrastructure, businesses require technology that supports their on demand initiatives. Building on IBM's mainframe leadership and strong heritage of virtualization across IBM systems, IBM Virtualization Engine™ enriches these industry-leading and innovative competencies across the entire IBM ℮server and Total Storage portfolio. It helps to further simplify and optimize the management of a heterogeneous IT infrastructure. This chapter introduces the basic concepts of IBM's On-Demand Infrastructure and IBM Virtualization Engine, and how the IBM ℮server p5 systems incorporate these technologies.

The remainder of this publication focuses on explaining the latest p5 virtualization technologies and walking you through various configurations you may find useful to better your understanding.

1.1 IBM's Virtualization Engine and on demand

The IBM on demand computing model applies to all levels of a business's IT stack. At the system level, the components are system objects (for example, computing capacity, storage, and files). At the application level, components are dynamically integrated application modules that constitute sophisticated, yet much more flexible applications. At the business level, the components are business objects, defined for particular vertical industries or more generally, as they apply horizontally across industries.

IBM Virtualization Engine, as shown in Figure 1-1, is comprised of a suite of system services and technologies that form key elements of IBM's on demand computing model. It treats resources of individual servers, storage, and networking products as if in a single pool, allowing access and management of resources across an organization more efficiently. Virtualization is a critical component in the on demand operating environment, and the system technologies implemented in the POWER5 processor based IBM @server p5 servers provide a significant advancement in the enablement of functions required for operating in this environment.

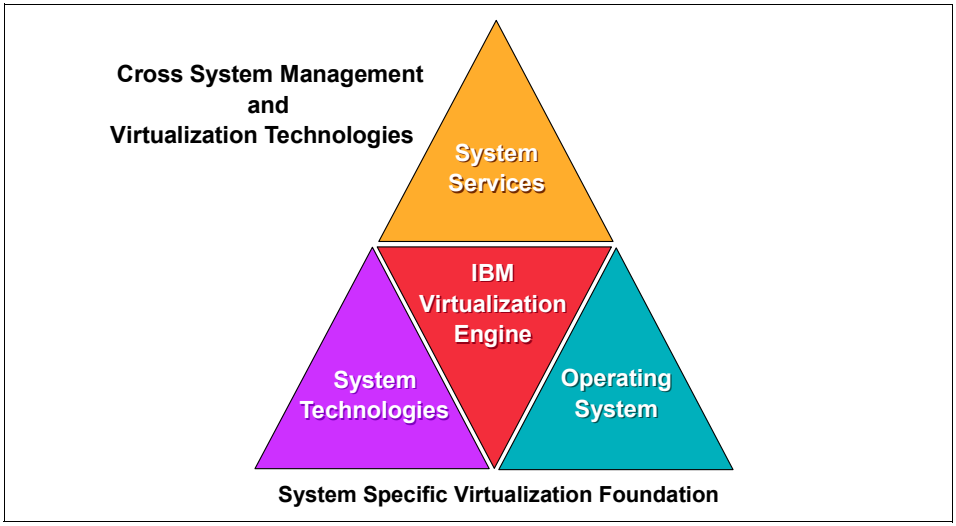

Figure 1-1 IBM Virtualization Engine components

The following sections explain the virtualization engine system technologies that are integrated into p5 system hardware and operating systems, including:

POWER Hypervisor™ Supports partitioning and dynamic resource movement across multiple operating system environments

Micro-Partitioning Enables you to allocate less that a full physical processor to a logical partition

Virtual LAN Provides network virtualization capabilities that allow you to prioritize traffic on shared networks

Virtual I/O Provides the ability to dedicate I/O adapters and devices to a virtual server, allowing the on demand allocation and management of I/O devices

Capacity on Demand Allows system resources such as processors and memory to be activated on an as-needed basis

Simultaneous multi-threading
Allows applications to increase overall resource utilization by virtualizing multiple physical CPUs through the use of multi-threading

Multiple operating system support
Logical partitioning allows a single server to run multiple operating system images concurrently

Figure 1-2 shows how several of these technologies combine to provide you the flexibility to help meet your computing requirements.

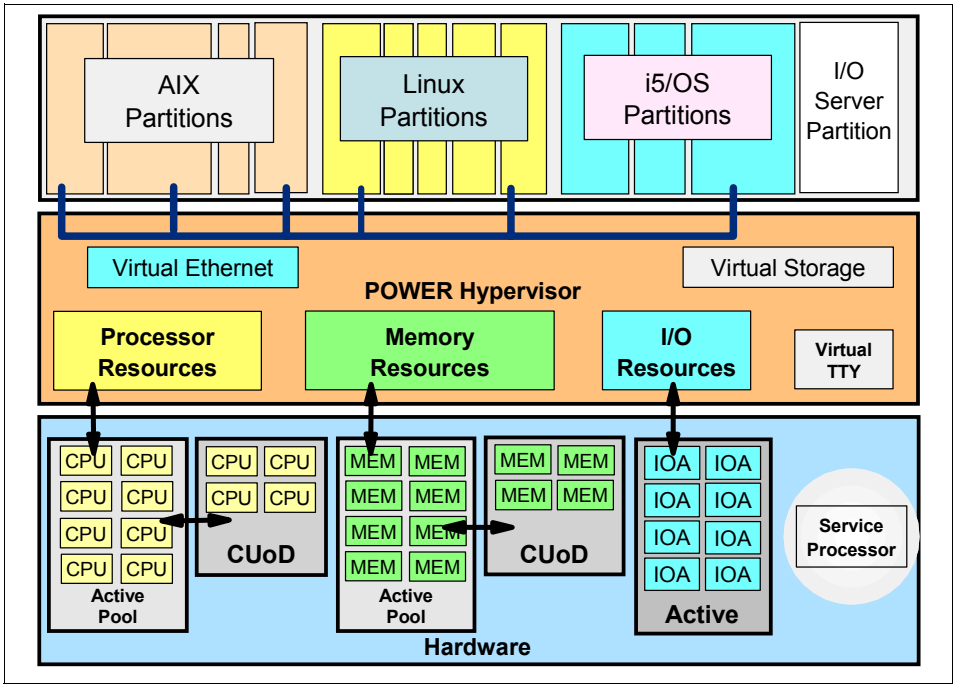

Figure 1-2 Virtualization technologies implemented on POWER5 servers

Along with these technologies, services integrated and tested in the IBM Virtualization Engine Suite for Servers include:

- IBM Director Multiplatform
- System Provisioning
- IBM Enterprise Workload Manager
- IBM Grid Toolbox V3 for Multiplatforms
- Business Resource Services

Completing the virtualization portfolio are the IBM Virtualization Engine Suite for Storage products, including:

- TotalStorage SAN Volume Controller
- TotalStorage SAN File System
- TotalStorage Productivity Center

1.2 Virtualization engine on IBM @server p5

The IBM @server Virtualization Engine system technologies added or enhanced on the POWER5 processor based p5 models 520, 550, and 570 are described in this section.

1.2.1 POWER Hypervisor

Always required on POWER5-based servers, the POWER Hypervisor is responsible for time slicing and dispatching the logical partition workload across the physical processors. The POWER Hypervisor also enforces partition security, and can provide virtual LAN channels between physical partitions, reducing the need for physical Ethernet adapters and freeing I/O adapter slots. See 3.2.2, “POWER Hypervisor processor dispatch” on page 46 for more information.

1.2.2 Simultaneous multi-threading (SMT)

Enhancements in POWER5 processor design allow for improved overall hardware resource utilization. SMT technology allows two separate instruction streams (threads) to run concurrently on the same physical processor, improving overall throughput. To the operating system, each hardware thread is treated as an independent logical processor. For any partition, it can be booted either in single thread (ST) mode or simultaneous multi-threading (SMT) mode. See 2.3.4, “SMT control” on page 31 for more information.

1.2.3 LPAR and Micro-Partitioning

Partitioning capabilities are enhanced to include sub-processor partitioning, or Micro-Partitioning. Partitions are not constrained to physical processor boundaries, and may share resources from a processor pool. It is possible to dynamically move system resources between partitions without rebooting. See 3.3, “Micro-Partitioning introduction” on page 54 for more information.

1.2.4 Virtual LAN

A function of the POWER Hypervisor, Virtual LAN allows secure communication between logical partitions without the need for a physical I/O adapter. See 3.4, “Virtual Ethernet introduction” on page 73 for more information.

1.2.5 Virtual I/O

Virtual I/O provides the capability for a single physical I/O adapter to be used by multiple logical partitions of the same server, allowing consolidation of I/O resources and minimizing the number of I/O adapters required. See 3.6, “Virtual SCSI introduction” on page 87 for more information.

1.2.6 Capacity on Demand

There are multiple Capacity on Demand (CoD) possibilities offered, including:

- Permanent Capacity on Demand:
  - Enables system upgrades by activating processors and/or memory.
  - No special contracts and no monitoring are required (no ability to turn off the capacity.)
  - Purchase agreement is activated using license keys.
- On/Off Capacity on Demand:
  - Enables the temporary use of a requested number of processors or amount of memory.
  - On a registered system, the customer selects the capacity and activates the resource.
  - Capacity can be turned ON and OFF by the customer; usage information is reported to IBM.
  - This option is post-pay. You are charged at activation.

- Reserve Capacity on Demand:
  - Used for processors only.
  - Prepaid debit temporary agreement, activated using license keys.
  - Adds reserve processor capacity to the shared processor pool, used if the base shared pool capacity is exceeded.
  - Requires AIX 5L Version5.3 and the Advanced POWER Virtualization feature.
- Trial Capacity on Demand:
  - Tests the effects of additional processors and memory.
  - Partial or total activation of installed processors and/or memory.
  - Resources are available for a fixed time, and must be returned after trial period.
  - No formal commitment required.

1.2.7 Multiple operating system support

The POWER5 processor based @server p5 products support IBM AIX 5L Version 5.2, IBM AIX 5L Version 5.3, SUSE Linux Enterprise Server 8 (SLES8), and Red Hat Enterprise Linux 3 (RHEL3).

IBM plans to extend the capabilities of the IBM @server p5 product line by introducing support for the i5/OS operating system. At the time of writing, support is planned for selected @server POWER5 570 models and future high-end @server POWER5 models.

1.3 Virtualization engine on IBM @server i5

Virtualization Engine system technologies added or enhanced in the POWER5 processor based i5 models 520, 550, and 570 are functionally very similar to what is available on the p5 products. Technologies such as Micro-Partitioning, SMT, POWER Hypervisor, and virtual LAN have the same implementation on both i5 and p5 @servers. Although functionally similar, the virtual I/O functions utilize different software mechanisms, and the rules for operating system support are not the same.

The topics covered in the remaining chapters of this redbook are described from the perspective of the AIX operating system running on a p5. AIX 5L is now supported on the i5 products, so in many cases this information will also apply to i5 models.

1.4 RAS and security

The POWER5 processor based @server products continue to build on IBM's industry leading RAS and security features available in previous IBM @server products. As individual servers become capable of hosting more system images, the importance of isolating and handling failures that might occur becomes greater. Hardware and operating system functions have been integrated into the system design to monitor system operation, predict where failures may occur, isolate failure conditions that do occur, then handle the failure condition, and when possible, continue operation.

1.4.1 Reliability, availability, and serviceability

Autonomic system capabilities are a critical building block of on demand computing, and reliability, availability, and serviceability (RAS) are key elements of these autonomic functions. The primary objective of the IBM @server RAS implementation is to minimize outages. This section highlights specific RAS capabilities introduced or enhanced in the @server products featuring the POWER5 processor.

There are four main categories to consider in discussing enhancements made in the hardware design of POWER5 systems:

- Reducing and avoiding outages
- Recovering from system failures
- Diagnostics and service
- Concurrency in maintenance

Reducing and avoiding outages

To reduce the probability of a system outage, and to minimize the impact of an outage if a hardware failure does occur, the following RAS capabilities exist:

- ECC is extended to the inter-processor fabric bus.
- PCI Enhanced Error Handling (EEH) is extended to the PCI host bridge (PHB) on the I/O planar.
- Containment of uncorrectable errors to code that uses the bad data is extended to errors originating on the fabric bus.
- Dynamic power management logic designed for the POWER5 processor allows it to dynamically turn off clocks to portions of logic, based on usage, resulting in a cooler operating environment and increased reliability.
- A failing processor can be dynamically swapped out if an available replacement processor exists in the CUoD pool.

Recovering from system failures

Design enhancements in both system hardware and firmware enable the following system recovery actions:

- Self-healing capabilities are extended by adding more redundancy in the L3 cache, with the ability to delete up to 10 L3 cache lines per module.
- More fine-grained deconfiguration is possible for L2 cache errors, resulting in a fraction of the L2 cache being garded out instead of the entire cache.
- An entire CEC node can be blocked after reboot for certain fabric errors.

Diagnostics and service

Problem determination and repair actions have been simplified with the addition of:

- Shared processor LPAR, allowing a failed physical processor to be blocked from use and diagnosed without system disruption.
- Compute-guided repair of components using HMC and lightpath functions.

Concurrency in maintenance

Enhancements made to allow service operations without the need to shut down the system include:

- The ability to apply some microcode updates concurrently with normal system operation.
- Ability to add I/O drawers concurrently.

All of the RAS capabilities implemented in POWER4™ processor based systems have been carried forward, including:

- First Failure Data Capture capabilities provided by system-wide error detection and fault handling.
- Memory RAS features include ECC, chipkill, bit steering to redundant memory bits, and soft error scrubbing.
- Hot-swap disk drives, fans, power supplies, and PCI and PCI-X adapter cards.

1.4.2 Security

The logical partitioning characteristics that were designed into POWER4/4+ processor based systems were based on maintaining the highest levels of security. Resources were isolated between partitions, and almost no sharing or communication between partitions existed. New features introduced in the

virtualization engine system technologies of POWER5 allow some of these barriers to *open up*, while maintaining a high degree of security.

Architectural enhancements made in the POWER5 @server system design make possible the cross-partition sharing and communication. Functions such as dynamic LPAR, shared processors, virtual LAN, virtual storage, and workload management all require architected facilities to ensure system security requirements are met. The basis for these enhancements is that cross-partition features are designed to not introduce any security exposure beyond what is implied by the function. For example, a virtual LAN connection would have the same security considerations as a physical network connection.

High security environments should carefully consider how they want to utilize cross-partition virtualization features. Any visibility between partitions must be consciously enabled through administrative system configuration choices.

1.5 Operating system support details

This section shows some more detailed information about the various operating system which are supported by the IBM Virtualization Engine. These are:

- AIX
- Linux
- i5/OS

1.5.1 AIX support

If you plan to use virtualization on AIX you need to consider the following details:

- Supported levels
- Licensing
- HACMP considerations

Supported levels

Although the IBM @server p5 servers support AIX 5L Version 5.2, it is not possible to run an IBM AIX 5L Version 5.2 partition with Micro-Partitioning, virtual SCSI, virtual Ethernet, or shared Ethernet adapters. A mixed environment between AIX 5L Version 5.2 and AIX 5L Version 5.3 partitions on p5 servers is supported.

Figure 1-3 on page 10 shows a sample configuration with mixed operating systems and mixed AIX 5L versions. The first five partitions are using dedicated processors. The AIX 5L Version 5.2 partition is not able to join the virtual I/O paths, but it provides all the known LPAR and DLPAR features. It has to be

configured with dedicated I/O adapters. The AIX 5L Version 5.3 partitions using shared processors likewise may use dedicated storage and dedicated networking.

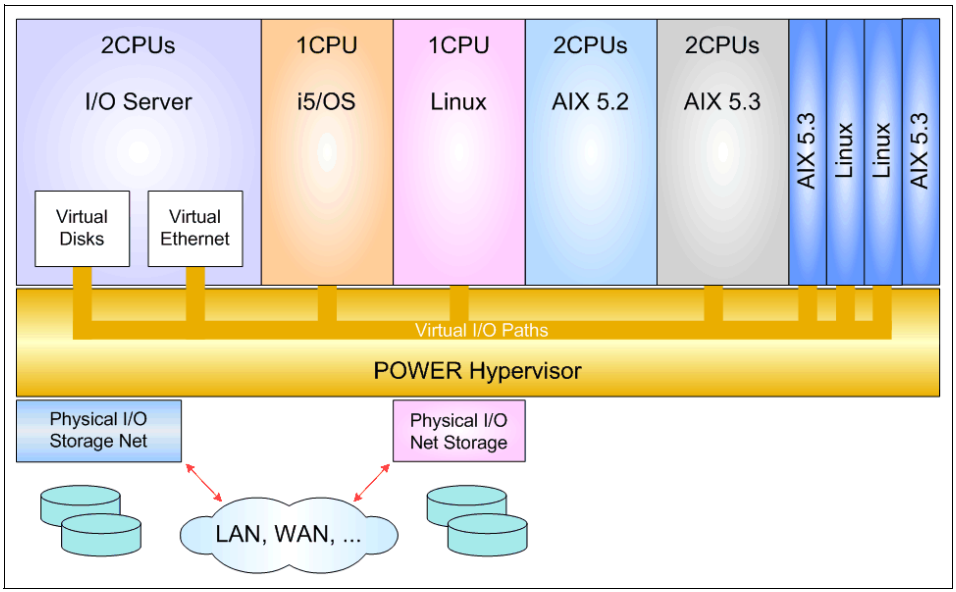

Figure 1-3 Mixed operating system environments

An overview of AIX 5L Version 5.2 and AIX 5L Version 5.3 and their supported features is provided in Table 1-1.

Table 1-1 AIX 5L features and versions

Feature	AIX 5L v5.2 ML4	AIX 5L v5.3
POWER4 support	Yes	Yes
POWER5 support	Yes	Yes
Dynamic LPAR -CPU	Yes	Yes
Dynamic LPAR - Memory	Yes	Yes
Dynamic LPAR -I/O	Yes	Yes
Micro-Partitioning	No	Yes
Virtual Ethernet	No	Yes
Virtual SCSI Client	No	Yes
Support for 254 partitions	No	Yes

Note: Partition Load Manager supports both IBM AIX 5L Version 5.2 and IBM AIX 5L Version 5.3 partitions running on one POWER5 system.

Licensing

AIX 5L Version 5.3 licenses are not bundled on the hardware; they have to be ordered separately. Because of the possibility to run other operating systems, such as LINUX, there is an agreement to pay only for the processors which are running IBM AIX. For shared processor (micro-partition) pools, the number of IBM AIX licenses required will be the lesser of the number of processors assigned to the pool, or the sum of the virtual processors of each uncapped partition, plus the processing units in each capped partition.

In the configuration shown in Table 1-2, 10 AIX licenses would be required: 2 (physical) for partition 1, and 8 (physical) for the shared processor pool.

The number of physical processors (8) is used because the number (9) of virtual processors used for AIX in the shared processor pool exceeds the number of physical processors, so the smaller number is used. This is added to the number of real processors (2) used for AIX in dedicated partitions.

The calculations of the number of virtual processors in the Micro-Partitions used in this example are as follows:

```
5 (Virtual) for partition #4+ (uncapped, so 100% of 5 virtual CPUs)
4 (Virtual) for partition #6 + (3.75 processors rounded up to 4)
```

Since the number of virtual processor licenses (9) is greater than the number of real processors (8) assigned to the shared processor pool, the smaller number (8) is the number of licenses required.

Table 1-2 AIX licensing example one

Partition number	1	2	4	5	6
Partition type	Dedicated	Dedicated	Micro-Partition		
Number of real processors	2	2	8		
Operating system	AIX	Linux	AIX	Linux	AIX
Number of virtual processors	2	2	5	4	5
Capped / Uncapped	N/A	N/A	Uncapped	Uncapped	Capped
Entitled capacity	N/A	N/A	5	4	3.75
Number of AIX licenses required	2		8		

In the configuration shown in Table 1-3, 6 AIX licenses would be required, as follows:

```
2 (physical) for partition #1
3 (Virtual) for partition   #4
1 (entitled capacity) for partition #6 + (+1 CPU of capacity)
```

The number of Micro-Partition virtual processors used for AIX (4) is less than the total number of real processors (8) in the pool, so only 4 AIX processor licenses are needed for the Micro-Partition AIX images. This is added to the number of real processors (2) used for AIX in dedicated partitions.

Customers will be required to purchase sufficient AIX licenses as part of the Terms and Conditions of AIX.

Table 1-3 AIX licensing example two

Partition number	1	2	4	5	6
Partition type	Dedicated	Dedicated	Micro-Partition		
Number of real processors	2	2	8		
Operating system	AIX	Linux	AIX	Linux	AIX
Number of virtual processors	2	2	3	4	5
Capped/Uncapped	N/A	N/A	Uncapped	Uncapped	Capped
Entitled capacity	N/A	N/A	3	4	1
Number of AIX licenses required	2		4		

Uncapped partitions are those that can consume up to all the available resources, while *capped partitions* have set limits. *Entitled capacity* is the maximum limit a capped partition can be set to consume if required by applications.

For application-licensing the vendors determine how they handle licensing when working with Micro-Partitioning technology.

See 3.3, “Micro-Partitioning introduction” on page 54 for more information on Micro-Partitioning.

HACMP considerations

At the time of writing, HACMP does not support virtual SCSI, virtual Ethernet, or Micro-Partitioning technology.

1.5.2 Linux support

The initial launch of POWER5 processor based @servers supports logical partitions running the Linux operating system. Both the SUSE Linux Enterprise Server 8 (SLES8) and Red Hat Enterprise Linux 3 (RHEL3) versions of Linux are supported. Linux partitions support many of the virtualization engine system technologies, such as Micro-Partitioning, SMT, Virtual LAN, and Virtual SCSI client.

For the latest updates to the SUSE or Red Hat Linux offering for POWER5 processor based @servers, refer to:

http://www.redhat.com/software/
http://www.suse.com

Not all devices and features supported by the AIX operating system are supported in logical partitions running the Linux operating system. Information on external devices and features supported on POWER5 @server products can be found at:

http://www-1.ibm.com/servers/eserver/pseries/linux

1.5.3 i5/OS support

IBM plans to extend the capabilities of the IBM @server p5 product line by introducing support for the i5/OS operating system. At the time of writing this redbook, support is planned for selected @server POWER5 570 models and future high-end @server POWER5 models, but is not yet available. This support will provide additional flexibility for large-scale server consolidation where IBM AIX 5L or Linux is the primary operating system.

1.6 Comparison with zSeries virtualization engine

The IBM zSeries products have a long history in the implementation of partitioning and virtualization technologies that are relatively new in UNIX. Table 1-4 on page 14 gives a brief overview of the virtualization capabilities available on zSeries and p5 servers. The subsequent paragraphs describe each of these capabilities in more detail.

Table 1-4 Virtualization capabilities on zSeries and p5 servers

Function / capability	zSeries	p5
Enabling software	PR/SM™	POWER Hypervisor
Sharing processor resources	Physical processor resources assigned as weighted CP shares to LPARs	Physical processor resources assigned as virtual processors to LPARs
Sharing memory resources	Memory can be dynamically assigned to LPARs	Memory can be dynamically assigned to LPARs
Sharing network resources	Interpartition communication through HiperSockets™	Interpartition communication through virtual Ethernet
	OSA adapter allows connection to external networks	Shared Ethernet Adapter connects virtual Ethernet to external networks
Sharing I/O resources	Access to I/O resources by shared devices and paths	Virtualized disks accessed through virtual SCSI adapters

1.6.1 Enabling software

zSeries provides logical partitioning capabilities through the PR/SM software.

On p5 servers, these capabilities are provided by the POWER Hypervisor, which is running as part of the firmware. See 3.2, “Introduction to the POWER Hypervisor” on page 45 for more information.

1.6.2 Sharing processor resources

On zSeries, processor resources are delivered to a partition by assigning logical processors and specifying weights for each partition. As many logical processors as physical processors can be assigned to any partition.

On p5 servers, each partition is assigned an entitled processor capacity represented by one or more virtual processors. The minimum assignable capacity to a virtual processor can be from as little as 1/10 to a whole physical processor. See 3.3, “Micro-Partitioning introduction” on page 54 for more information.

1.6.3 Sharing memory resources

Sharing of memory resources is fairly similar on p5 and zSeries servers. Each partition can be assigned a certain amount of memory, which gets allocated when to partition is activated. Once a partition is active allocations can be

dynamically reconfigured. On p5 servers, memory is assigned in increments of 16 MB.

1.6.4 Sharing I/O resources

On zSeries, I/O resources are shared by defining devices and paths that can be assigned to several partitions.

On p5 servers, I/O resources (disks and adapters) are shared through Virtual I/O. The physical resources are owned by the virtual I/O server (see Chapter 4, “Virtual I/O Server configuration” on page 99 for more information). A physical disk can be split into several logical volumes. These logical volumes are then exported to the client partitions which see them as normal disks through virtual SCSI adapters.

1.6.5 Sharing network resources

On zSeries, communication between partitions is done through HiperSockets. OSA adapters are used to connect to external networks.

On p5 servers, inter-partition communication is done through a Virtual Ethernet. The partitions connect to virtual Ethernet through a Virtual Ethernet Adapter (see 3.4, “Virtual Ethernet introduction” on page 73 for more information).

A shared Ethernet adapter (SEA) acts as a layer 2 switch to route traffic to external networks. The SEA is hosted by a virtual I/O server partition (see 4.2.1, “Ethernet adapter sharing” on page 109 for more information).

2

POWER5 processor introduction

Several years ago, IBM set out to design a new microprocessor that would leverage IBM strengths across many different disciplines to deliver a server that would redefine what was meant by the term *server*. POWER4 was the result. It was developed by over 300 engineers in several IBM development laboratories.

The POWER5 system is IBM's next generation of POWER-based microprocessors. It builds on the POWER4 architecture, providing new and improved functional support designed to meet a variety of customer needs and requirements. This chapter provides an overview of the POWER5 design and discusses various aspects of the functional enhancements which the POWER5 system is designed to support.

2.1 General description

The POWER4 microprocessor, which was introduced in 2001, was a result of advanced research technologies developed by IBM to create a high-performance, high-scalability chip design to power future IBM @server systems. The POWER4 design integrates two processor cores on a single chip, a shared second-level cache, a directory for an off-chip third-level cache, and the necessary circuitry to connect it to other POWER4 chips to form a system. The dual-processor chip provides natural thread-level parallelism at the chip level.

POWER5 is IBM's second generation dual-core microprocessor chip. It extends the POWER4 design by introducing enhanced performance and support for a more granular approach to computing. The POWER5 chip features single- and multi-threaded execution, and higher performance in the single-threaded mode than its POWER4 predecessor at equivalent frequencies.

The primary design objectives of POWER5 technology are:

- Maintain binary and structural compatibility with existing POWER4 systems
- Enhance and extend SMP scalability
- Continue superior performance
- Provide additional server flexibility
- Deliver power efficient design
- Enhance reliability, availability, and serviceability

2.1.1 POWER5 system structure

Key enhancements introduced into the POWER5 processor and system design include:

- Simultaneous multi-threading
- Dynamic resource balancing to efficiently allocate system resources to each thread
- Software-controlled thread prioritization
- Dynamic power management to reduce power consumption without affecting performance
- Micro-Partitioning technology
- Virtual storage, virtual Ethernet
- Enhanced scalability, parallelism
- Enhanced memory subsystem

Figure 2-1 shows the high-level structure of POWER5 processor-based systems. Moving the level-three (L3) cache from the memory side to the processor side of the fabric lets the POWER5 more quickly satisfy level-two (L2) cache misses with hits in the 36-Mbyte off-chip L3 cache, avoiding traffic on the interchip fabric. References to data not resident in the on-chip L2 cache cause the system to check the L3 cache before sending requests onto the interconnection fabric. Moving the L3 cache provides significantly more cache on the processor side than previously available, thus reducing traffic on the fabric and allowing POWER5-based systems to scale to higher levels of symmetric multiprocessing.

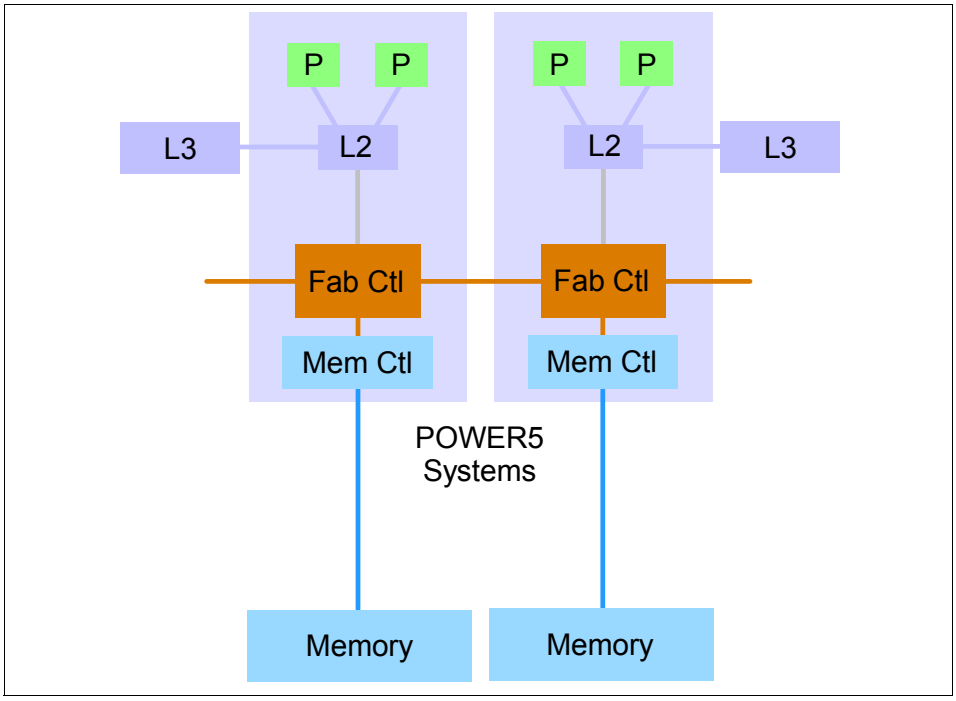

Figure 2-1 POWER5 system structure

POWER5 chip overview

Figure 2-2 on page 20 shows the POWER5 chip, which is fabricated by IBM using silicon-on-insulator (SOI) devices and copper interconnect. SOI technology reduces device capacitance to increase transistor performance. Copper interconnect decreases wire resistance and reduces delays in wire-dominated chip-timing paths. In 130 nm lithography, the chip contains 276 million transistors, uses eight metal levels, and measures 389 mm^2.

The POWER5 processor supports the 64-bit PowerPC® architecture. A single die contains two identical processor cores, each supporting two logical threads.

This architecture makes the chip appear as a four-way symmetric multiprocessor to the operating system.

The two cores share a 1.92 MB L2 cache. The L2 cache is implemented as three identical slices with separate controllers for each. The L2 slices are 10-way set-associative with 512 congruence classes of 128-byte lines. The data's real address determines which L2 slice the data is cached in. Either processor core can independently access each L2 controller. The directory is also integrated for an offchip 36 MB L3 cache on the POWER5 chip. Having the L3 cache directory on chip allows the processor to check the directory after an L2 miss without experiencing off-chip delays.

To reduce memory latencies, the memory controller is integrated on the chip, which eliminates driver and receiver delays to an external controller.

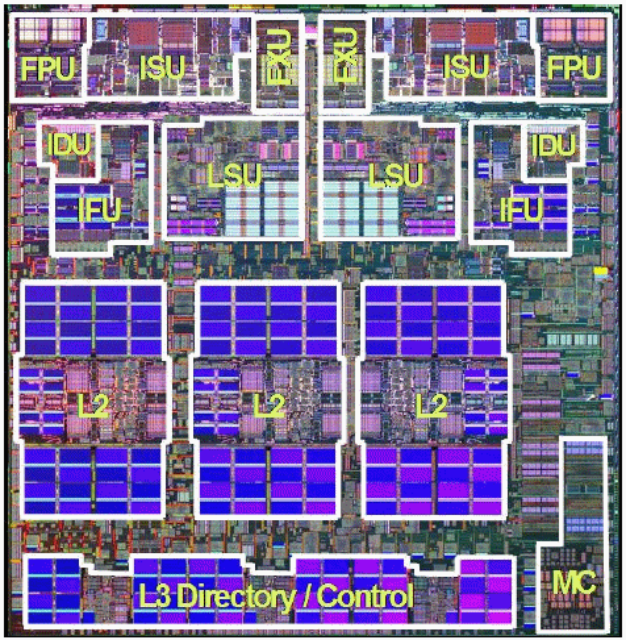

Figure 2-2 POWER5 processor chip

POWER5 processor core overview

The POWER5 processor core supports both enhanced simultaneous multithreading (SMT) and singlethreaded (ST) operation modes. Each core has 120 rename registers for integer and for FP, eight execution units and up to five instructions per cycle and four FLOPs per cycle issuing throughput, as well as I-cache of 64 KB (2-way set associative) and D-cache (32 KB, 4-way set associative). The cores share a three-bank L2 Cache of 1.92 MB (3 x 640 KB

caches with independent buses, 10-way set associative) and also have enhanced data stream prefetching to make better use of that bandwidth.

POWER5 packaging, MCM and DCM

POWER5 chips can be packaged in several ways such as MCM, DCM, or mounted on a system planar.

When several POWER5 chips are packaged on a single module to form an 8-way SMP, they are known as a Multichip module (MCM).

Each MCM houses four POWER5 chips, (eight processor cores) that are connected through chip-to-chip ports, with their associated L3 chips. The POWER5 chips are mounted on the MCM such that they are all rotated 90 degrees from one another. This arrangement minimizes the interconnect distances, which improves the speed of the inter-chip communication. There are separate communication buses between processors in the same MCM, and processors in different MCMs, as shown in Figure 2-3.

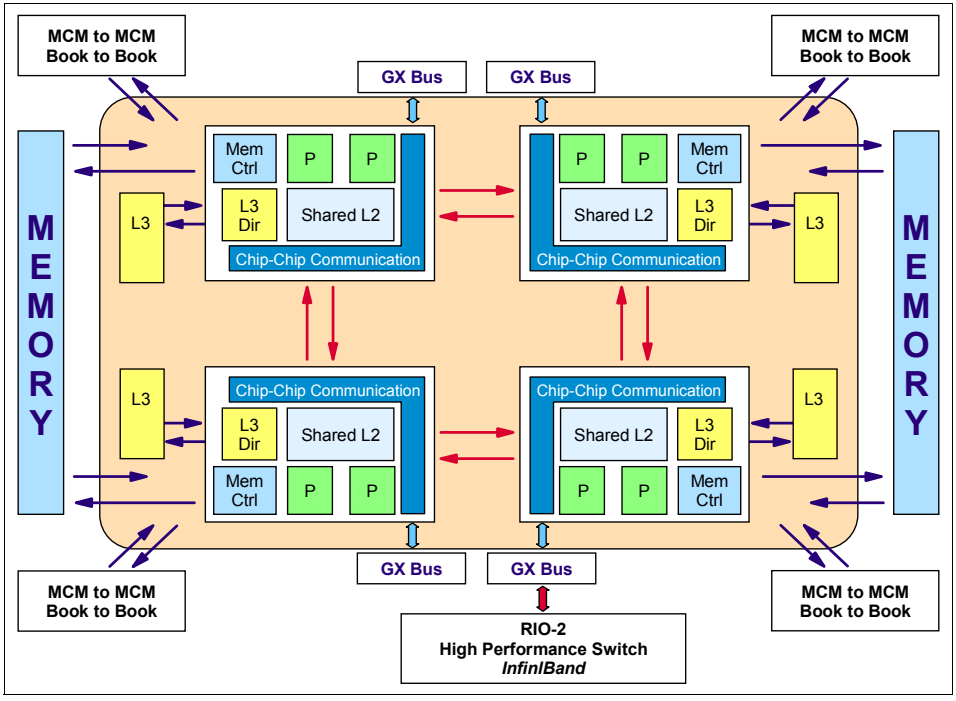

Figure 2-3 Logical view of the POWER5 multichip module

Two of the MCMs are then tightly coupled into a *book* as shown in Figure 2-4 on page 22.

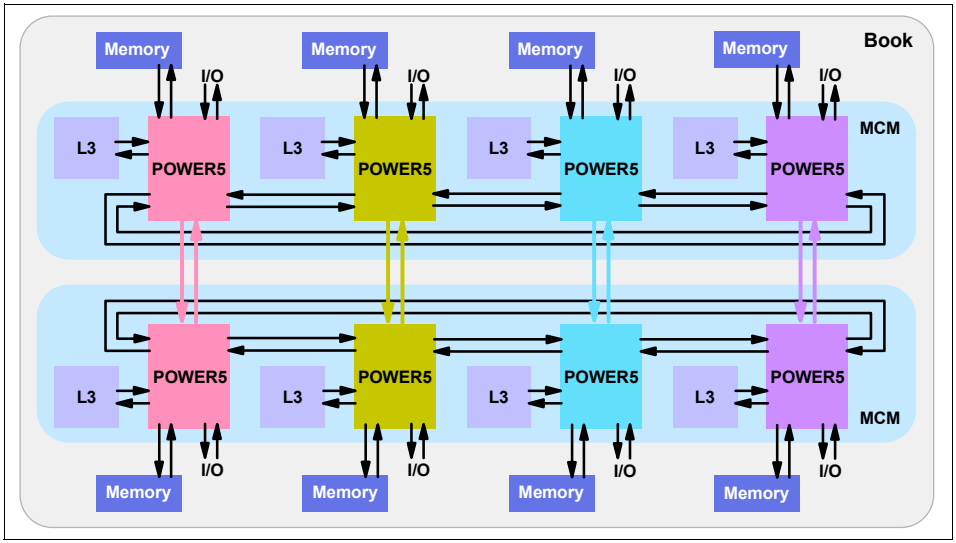

Figure 2-4 16-way POWER5 building block

Figure 2-5 shows four of these *books,* which together form a single SMP machine providing a 64-way POWER5 system.

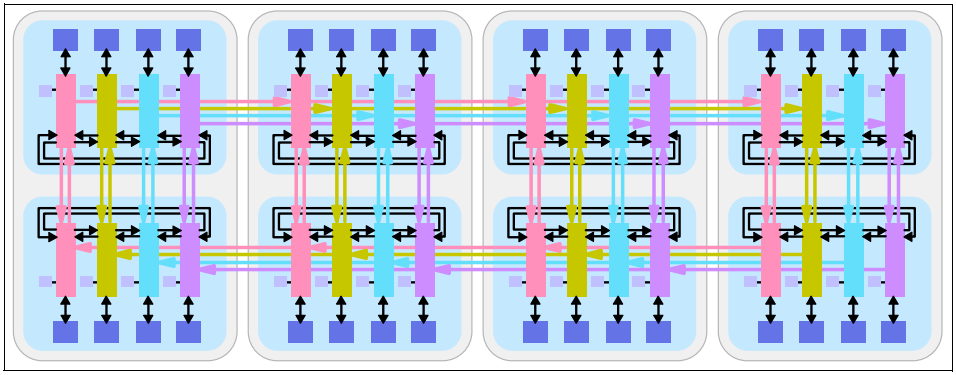

Figure 2-5 64-way POWER5 SMP interconnection

The POWER5 processor can also be packaged on a processor card containing one Dual-chip Module (DCM) and memory, or as a DCM surface mounted directly to a system planar. As Figure 2-6 on page 23 shows, a DCM contains only one POWER5 chip and one L3 chip. This design is available on the entry to mid-range POWER5-based servers.

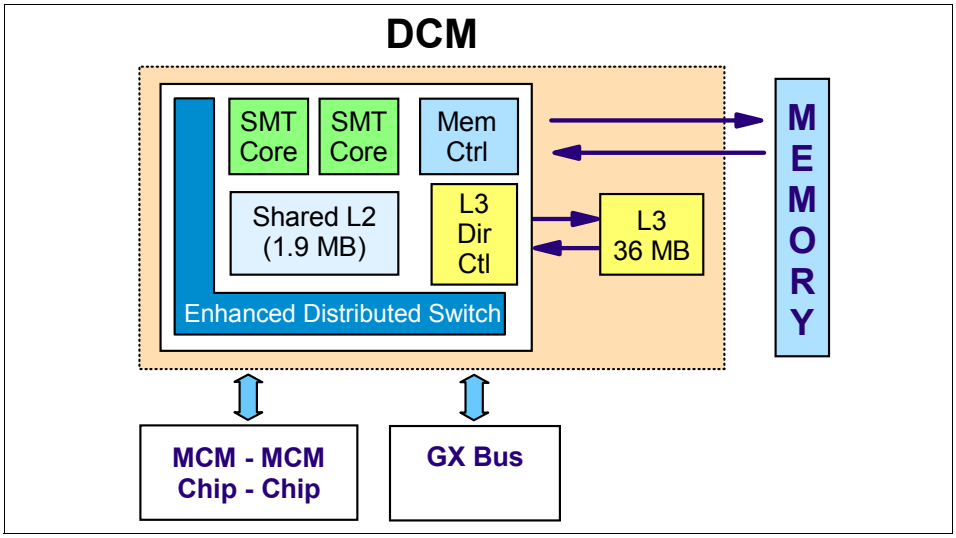

Figure 2-6 POWER5 dual-chip module

Summary

POWER5-based systems provide unmatched flexibility and performance. Many of the features that enable flexibility and performance challenge existing notions of how systems *look and feel*. IBM has already invested in ensuring that software can exploit the increased performance levels POWER5 systems will be offering, and is continuing in its pursuit to produce system-level enhancements to provide even greater performance increases over time.

2.2 POWER4 to POWER5 comparison

The POWER4 processor introduced two processors embedded in a single chip, sharing a common L2 level cache to increase processing efficiency. The POWER5 processor builds on this topology with a significant enhancement called Simultaneous Multithreading (SMT), a concept where multiple threads of execution can execute on the same processor at the same time. Multiprocessing within POWER4 took a different approach by providing double the amount of processors on a chip (one thread per processor), double the number of resources completely available to work on the user's problem set, each having its own L1 cache, but sharing a common L2 cache. The POWER5 series improves on the multithreading capabilities within a processor topology similar to that of POWER4, so that each processor can execute two instruction streams at the same time, without switching between them.

When the operating system's task dispatcher looks at this chip, it *sees* four available processors, four places onto which to dispatch a task: two physical processors, then, with the addition of multithreading, two logical processors per physical processor. (This depends on the operating system, specifically if the SMT control option is on or off. See the **smtctl** command description in 2.3.4, "SMT control" on page 31 for details.)

The detailed description of POWER5 architecture is not covered in this publication. Instead, we mention only the highlights and main changes that have an influence on virtualization.

There are several major differences between POWER4 and POWER5 chip designs, including in the following areas (detailed in Table 2-1):

- L1 cache
- L2 cache
- L3 cache
- Memory bandwidth
- Simultaneous multi-threading (SMT)
- Processor addressing
- Physical size

Table 2-1 POWER4 to POWER5 comparison

	POWER4 design	**POWER5 design**
L1 data cache	2-way associative FIFO	4-way associative LRU
L2 cache	8-way associative 1.44 MB	10-way associative 1.9 MB
L3 cache	32 MB 118 clock cycles	36 MB ~80 clock cycles
Memory bandwidth	4 GB/sec. / chip	~16 GB/sec. / chip
Simultaneous multi-threading	No	Yes
Processor addressing	1 processor	1/10 of processor
Dynamic power management	No	Yes
Size	412 mm	389 mm

The POWER5 processor can support both enhanced SMT mode and single threaded (ST) operation modes. All pipeline latencies in the POWER5, including

the branch misprediction penalty and load-to-use latency with an L1 data cache hit, are the same as in the POWER4. The identical pipeline structure allows optimizations designed for POWER4-based systems to perform equally as well on POWER5-based systems.

Dynamic power management

Chip power has become one of the most important design considerations using current CMOS technologies. The POWER5 design adds logic to reduce chip power, which enhances the reliability characteristics of the chip.

Dynamic power management logic designed for POWER5 can reduce the chip switching power, allowing it to run cooler. It uses a fine-grained clock gating mechanism to turn off the clocks to a local clock buffer if the logic detects that the set of latches driven by the buffer will not be used on the next clock cycle. For example, if a set of registers is guaranteed not to be read in a given cycle, the power management logic detects this, then turns off the clock to those registers' read ports.

In addition to the dynamic power management logic, POWER5 also has a low-power mode of operation. This mode is set by the operating system, and causes both processor threads to be set at the lowest priority level. In low-power mode, instructions are dispatched once every 32 cycles at most, reducing switching power even further. This mode is used only when there is no work to dispatch on either processor thread.

2.3 Introduction to simultaneous multi-threading

With increased processor speeds, memory can appear farther away, and longer instruction execution *stalls* can occur while waiting on memory accesses to be serviced by the memory subsystem. Conventional processors execute instructions from a single instruction stream, and despite micro architectural advances, execution unit utilization remains low in today's microprocessors. It is not unusual to see average execution unit utilization rates of approximately 25 percent across a broad spectrum of environments.

Simultaneous multi-threading (SMT) is a hardware design enhancement in POWER5 that allows two separate instruction streams (threads) to execute simultaneously on the processor. It combines the wide-issue capabilities of superscaler processors with the latency hiding abilities of hardware multi-threading.

2.3.1 Multi-threading techniques

The IBM STAR series of processors used in S85 servers used a hardware multi-threading (HMT) technique called *course grain multi-threading* that, when enabled, would allow multiple threads to run in parallel. When one or more threads were stalled on a long latency event, such as waiting for data from a cache miss, another thread would be dispatched to use the processor resource.

As illustrated in Figure 2-7 on page 27, there could be several idle cycles in the processor as thread contexts are swapped, since both threads share many system resources.

Some processors also implemented an HMT technique referred to as *fine grain multi-threading*. Multiple process threads in this implementation would use processor resources on a cycle-by-cycle basis, alternating between threads on each cycle with no latency. Still, only one thread would be active on a given processor at any point in time.

POWER4 introduced an SMP on a single chip.

Finally, in *simultaneous multi-threading* (SMT) as implemented by POWER5, and as in other multi-threaded implementations, the processor fetches instructions from more than one thread. What differentiates this implementation is its ability to schedule instructions for execution from all threads concurrently. With SMT, the system dynamically adjusts to the environment, allowing instructions to execute from each thread if possible, and allowing instructions from one thread to utilize all the execution units if the other thread encounters a long latency event.

The acronyms provided on the left of each block represent the fixed-point execution (FX) units, the load store (LS) units, the float point (FP) units, the branch execution (BRX) units, and the condition register logical execution unit (CRL).

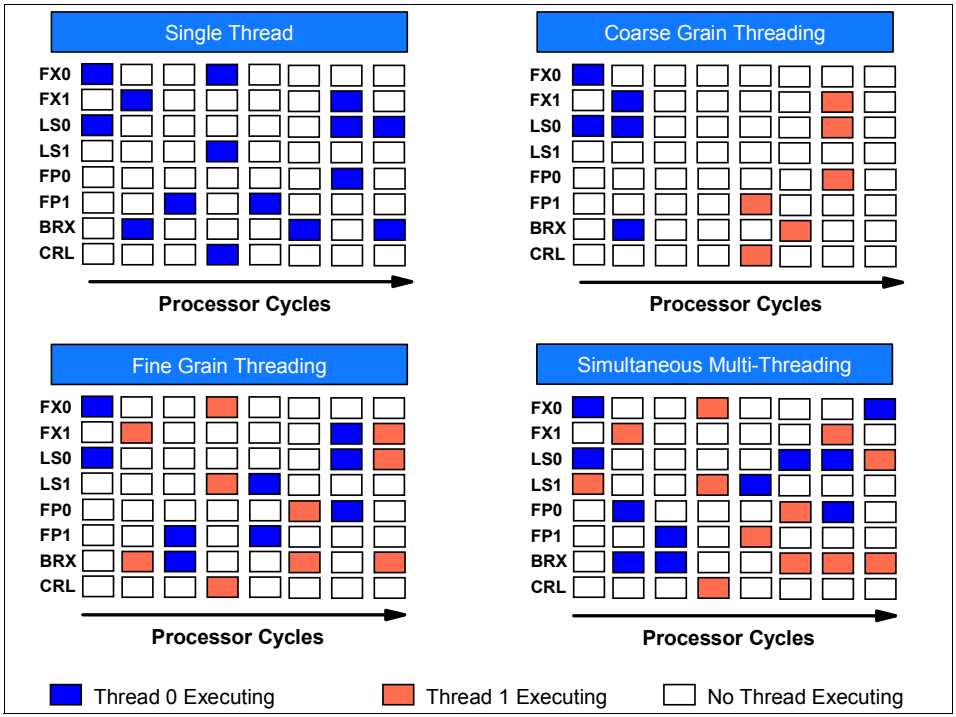

Figure 2-7 Multi-threading techniques

2.3.2 POWER5 SMT

Using multiple on-chip thread contexts, the SMT processor executes instructions from multiple threads each cycle. By duplicating portions of logic in the instruction pipeline and increasing the capacity of the register rename pool, the POWER5 processor can execute two instruction streams, or threads, concurrently. Through hardware and software thread prioritization, greater utilization of the hardware resources can be realized without an impact to application performance. Figure 2-8 on page 28 illustrates the increased processor resource utilization using SMT in POWER5 compared with POWER4.

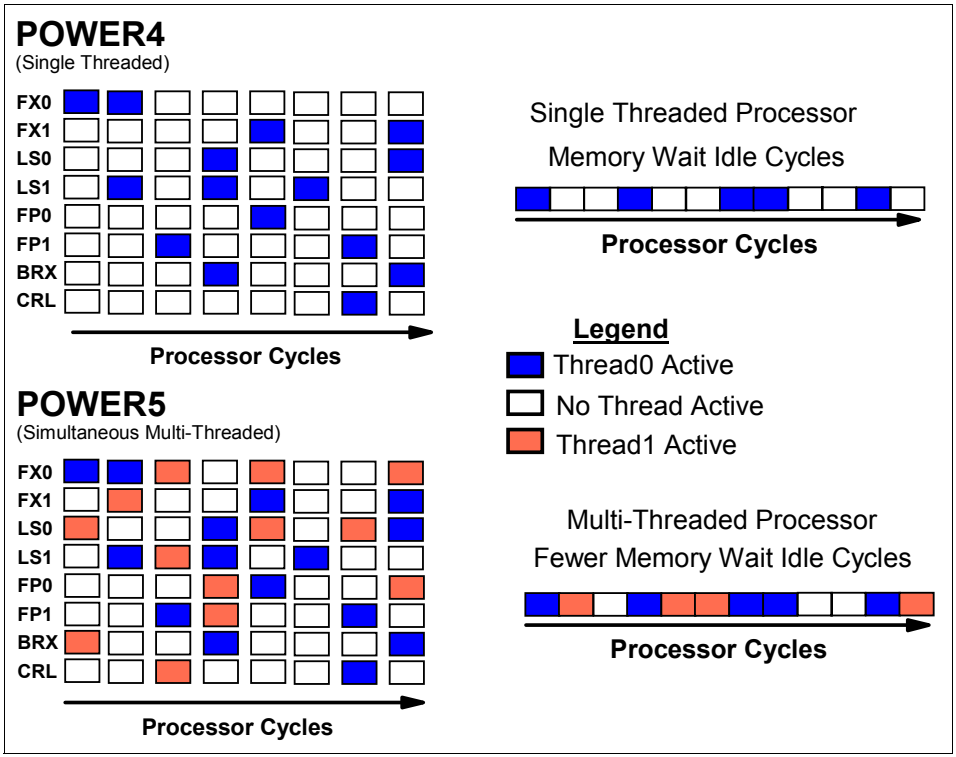

Figure 2-8 Increased processor resource utilization using SMT

Characteristics of POWER5 SMT implementation are as follows:

- Eight priority levels exist for each thread, and they can be raised or lowered by the hypervisor, operating system, or application.
- Processor resources optimized for best SMT performance, providing the ability to reduce priority of a thread that is consuming maximum resources or hold decode of a thread with long latency events.
- Dynamic feedback of shared resources allows for balanced thread execution.
- Software-controlled thread priority.
- Dynamic thread switching capabilities.

In SMT mode, the POWER5 processor uses two separate instruction fetch address registers to store the program counters for the two threads. Instruction fetches (IF stage) alternate between the two threads. It can fetch up to eight instructions from the instruction cache (IC stage) every cycle. The two threads share the instruction cache and the instruction translation facility. In a given cycle, all fetched instructions come from the same thread.

Not all applications benefit from SMT. Having two threads executing on the same processor will not increase the performance of applications with execution-unit-limited performance or applications that consume all the chip's memory bandwidth. For this reason, the POWER5 supports *single-threaded* (ST) execution mode. In this mode, the POWER5 gives all the physical resources to the active thread, allowing it to achieve higher performance than a POWER4 system at equivalent frequencies. In ST mode, the POWER5 uses only one program counter and can fetch instructions for that thread every cycle.

When the POWER5 processor is operating in ST mode, the inactive thread will be in one of two possible states, *dormant* or *null*, as shown in Figure 2-9. From a hardware perspective, the only difference between these states is whether or not the thread awakens on an external or decrementer interrupt.

The dormant state is entered when the operating system boots up in SMT mode, then instructs the hardware to put the thread into the dormant state when there is no work for that thread. This allows the active thread to use all of the physical processor resources. To make a dormant thread active, either the active thread executes a special instruction, or an external or decrementer interrupt targets the dormant thread. The hardware detects these scenarios and changes the dormant thread to the active state. It is software's responsibility to restore the architected state of a thread transitioning from the dormant to the active state.

When the system is set to operate in ST mode, by use of the **`smtctl`** command, the inactive thread is put into the null state, and the operating system is unaware of the thread's existence. No system resources are allocated to a null thread. This mode is advantageous if all the system's executing tasks perform better in ST mode.

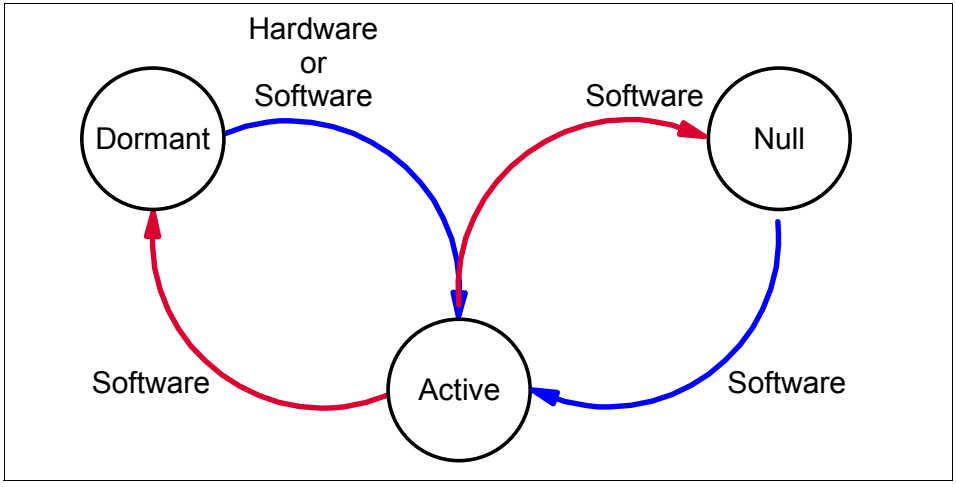

Figure 2-9 POWER5 thread states

Benefit of SMT

The benefit of SMT will be realized more in commercial environments, since the number of transactions performed outweighs the actual speed of the transaction. For example, the SMT environment would be much better suited for a Web server or database server than it would be for data-intensive high performance computing workloads. Some applications that are tuned to optimize the use of processor resources may actually see a decrease in performance due to increased contention to cache and memory. SMT may be disabled for these cases.

SMT is not a pure substitute for Symmetric Multi-Processing (SMP) environments, and is not a way to claim an SMP system larger than the number of physical processors. This is primarily due to the variability in performance caused by the workload sensitive nature of one SMT thread on another. Workload variability will also increase in shared partitions, due to latencies associated with scheduling virtual processors and interrupts.

If an application or workload environment is cache sensitive or cannot tolerate the variability introduced with resource sharing, it should be deployed in a dedicated processor partition with SMT disabled. In dedicated partitions, the entire physical processor is assigned solely to the partition.

Shared environments are discussed in 3.3.1, “Shared processor partitions” on page 55.

2.3.3 AIX 5L and SMT

Each hardware thread is supported as a separate logical processor by AIX 5L Version 5.3, independent of the type of partition. A partition with one dedicated processor would be configured as a logical 2-way by default. A shared partition with two virtual processors is configured as a logical 4-way by default. Both active threads running on an SMT processor are always from the same partition.

The POWER hypervisor saves and restores all necessary processor state information when preempting or dispatching virtual processor threads. This dispatch mechanism is discussed in 3.2.2, “POWER Hypervisor processor dispatch” on page 46.

Since shared processor capacity is delivered in terms of whole physical processors, differences exist in the way processor entitlement is applied when considering SMT mode.

Shared processor capacity and processor entitlement are new terms here that are further explained in the next chapter. For this discussion, consider *entitlement* as the processing resource defined for a partition out of a larger pool

of available resources. This capacity is from a shared processor pool. A partition utilizes this capacity through the logical processors, much the way an SMP system would provide real processors dedicated to a partition.

Figure 2-10 shows a 4-way shared processor partition with 200 processing units of entitlement. With SMT disabled, the partition is configured with 4 logical processors each having 50% of a physical processor. With SMT enabled, the partitions become an 8-way, where each logical processor has the power of 25% of a physical processor. Both threads of an SMT-enabled processor are dispatched together, so they are both active for the full duration of the 50% dispatch window, allowing them more than their individual capacities would normally allow for interrupt handling or other latency concerns.

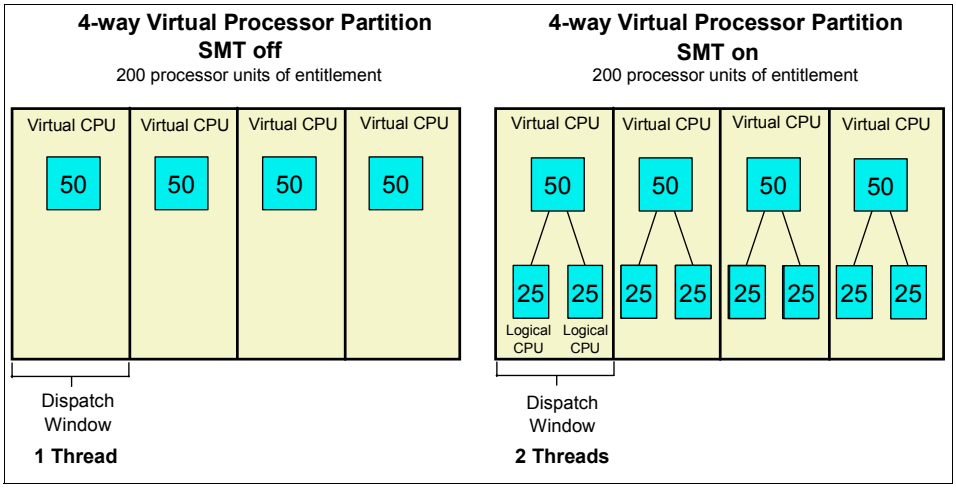

Figure 2-10 CPU entitlement in a shared processor partition

The POWER5 processor allows priorities to be assigned to hardware threads. Normally AIX 5L will maintain threads at the same priority, but has the ability to raise or lower the priority to optimize performance. For example, if a thread is spinning in an idle loop or waiting for a lock to free, thread priority will be lowered. Thread priority would be raised if it were holding a critical resource or lock. The ability to adjust thread priority does not persist into user mode, and AIX 5L does not consider an application software thread's dispatching priority when choosing its hardware thread priority.

2.3.4 SMT control

The SMT policy is controlled by the operating system, thus it is partition specific and can be enabled or disabled dynamically on a logical partition boundary. SMT mode can be changed dynamically during runtime, or on a subsequent reboot. It

can be changed using the AIX 5L Version 5.3 **smtctl** command, or with the system management interface tool (SMIT), which are the topics of the following sections.

Setting SMT mode using the command line

AIX 5L Version 5.3 adds the **smtctl** command, which controls the enabling and disabling of SMT mode. It provides privileged users and applications a means to enable or disable SMT for all processors in a partition either immediately or on a subsequent boot of the system.

The two flags associated with **smtctl** are **-m** and **-w**; they are defined as follows:

-m off Will set SMT mode to disabled

-m on Will set SMT mode to enabled

-w boot Makes the SMT mode change effective on the next and subsequent reboots

-w now Makes the mode change effective immediately, but will not persist across reboot

The **smtctl** command does not rebuild the boot image. If you want to change the default SMT mode of AIX, the **bosboot** command must be used to rebuild the boot image. The boot image in AIX 5L Version 5.3 has been extended to include an indicator that controls the default SMT mode.

Note: If neither the **-w boot** nor the **-w now** flags are entered, the mode change is made immediately and will persist across reboots. The boot image must be remade with the **bosboot** command in order for a mode change to persist across subsequent boots, regardless of **-w** flag usage.

The **smtctl** command entered without a flag will show the current state of SMT in the partition. An example of the **smtctl** command follows.

```
# smtctl

This system is SMT capable.

SMT is currently enabled.

SMT boot mode is set to enabled.
Processor 0 has 2 SMT threads
SMT thread 0 is bound with processor 0
SMT thread 1 is bound with processor 0

Processor 2 has 2 SMT threads
```

```
SMT thread 2 is bound with processor 2
SMT thread 3 is bound with processor 2
```

To dynamically turn SMT off for the current operating environment using **smtctl** with the **-m** and **-w** flag:

```
# smtctl -m off -w now
smtctl: SMT is now disabled.
```

Entering **smtctl** without an option to verify the current state shows:

```
# smtctl

This system is SMT capable.

SMT is currently disabled.

SMT boot mode is set to enabled.

Processor 0 has 1 SMT threads
SMT thread 0 is bound with processor 0

Processor 2 has 1 SMT threads
SMT thread 1 is bound with processor 2
```

If the logical partition was rebooted at this point, it would revert back to the default mode of SMT enabled as shown in the command output line indicating SMT boot mode status. To disable SMT across system reboots, the **-w boot** option must be used. After entering **smtctl** with the **-w boot** option and before rebooting, the boot record must be updated to reflect the change using the **bosboot -a** command.

```
# smtctl -m off -w boot
smtctl: SMT will be disabled on the next reboot.
        Note that the boot image must be remade with the bosboot
        command before the next reboot.
```

Setting SMT mode using SMIT

SMT mode can also be viewed or set from the SMIT menus as shown starting at Figure 2-11 on page 34. The selection sequence from the main System Management SMIT panel to enable or disable SMT for the operating system is:

1. Performance & Resource Scheduling

2. Simultaneous Multi-Threading Mode
3. Change SMT Mode

As shown in Figure 2-15 on page 36, and as discussed in the **smtctl** command description, the SMT mode can be set to change immediately without carrying over to a subsequent reboot, immediately and carrying over on a subsequent reboot, or on the next and all subsequent reboots.

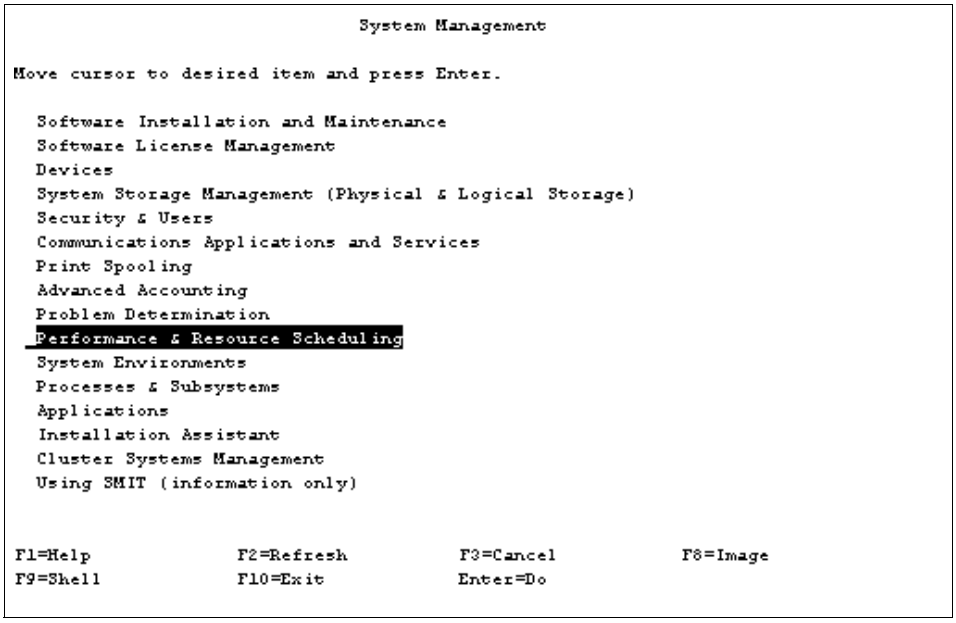

Figure 2-11 System Management primary screen

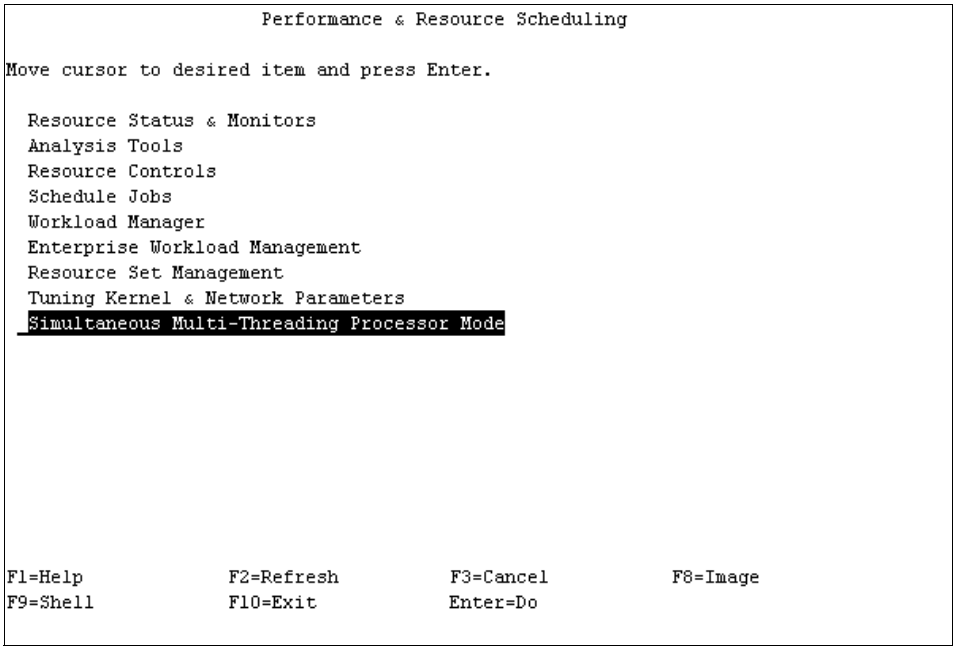

Figure 2-12 Performance & Resource Scheduling SMIT screen

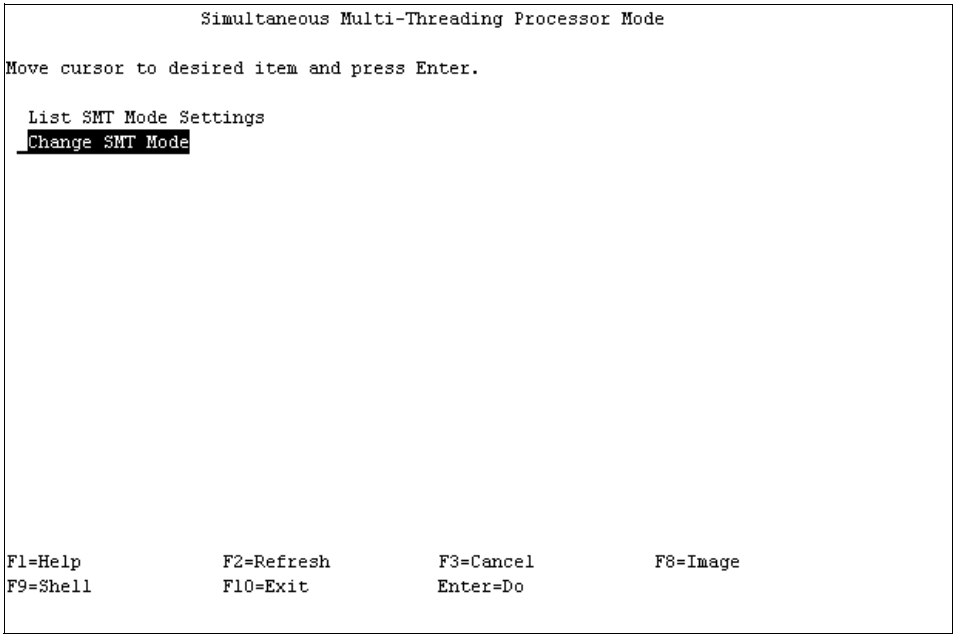

Figure 2-13 Simultaneous Multi-Threading Processor Mode screen

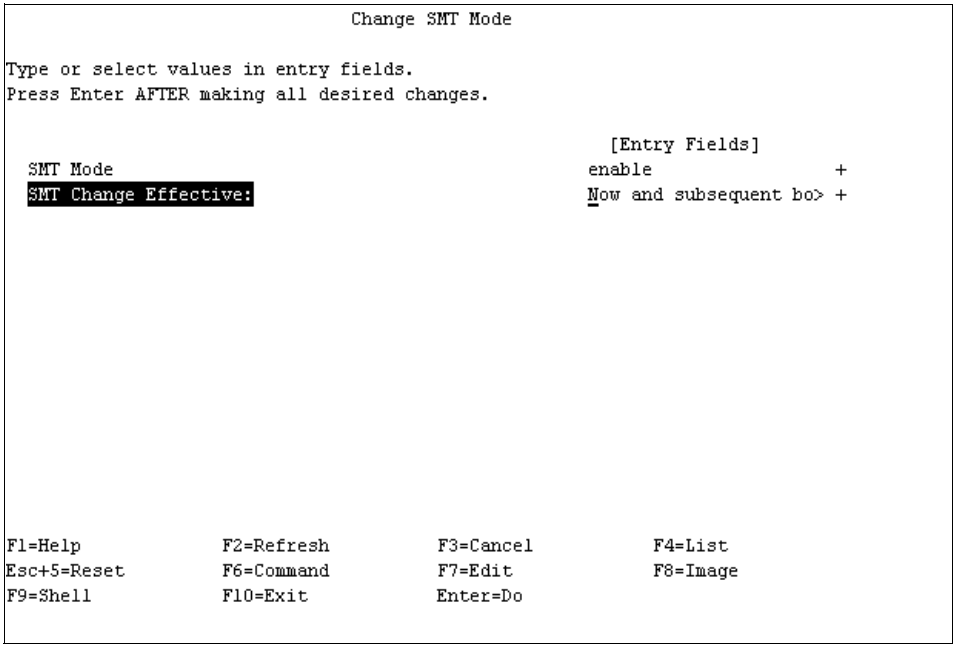

Figure 2-14 Change SMT Mode screen

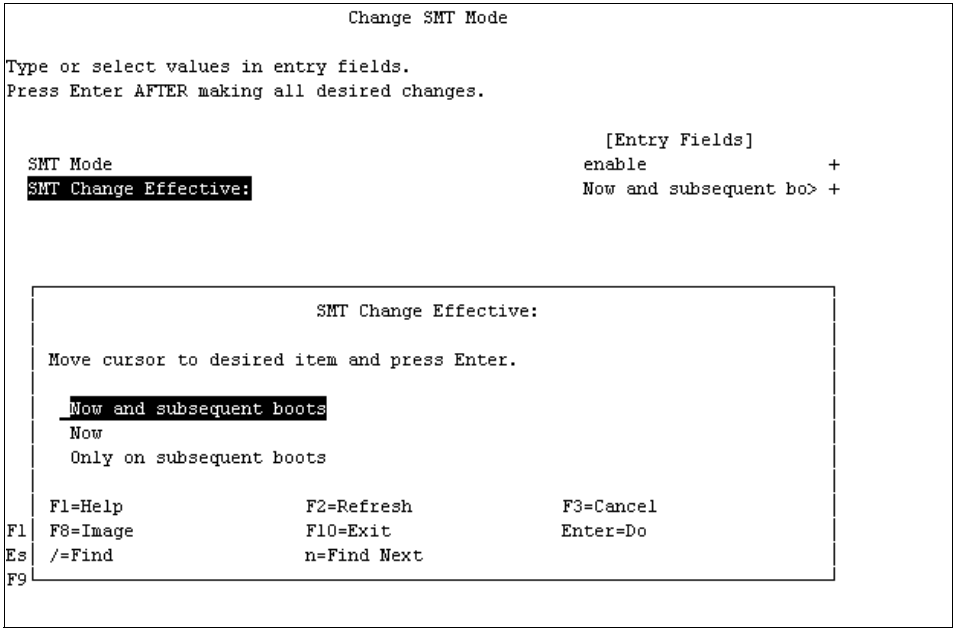

Figure 2-15 Options to select when SMT mode change become effective

2.3.5 SMT performance monitor and tuning

AIX 5L Version 5.3 includes commands or extended options to existing commands for the monitoring and tuning of system parameters in SMT mode.

SMT monitoring

The SMT behavior requires the operating system to provide statistics on the use of the particular logical processors. The **mpstat** command in AIX 5L Version 5.3 collects and displays performance statistics for all logical processors operating in the logical partition. When the **-s** flag is specified, the **mpstat** command reports SMT utilization if SMT is enabled. The report displays the virtual processor utilization, with the logical processor (thread) data associated with each virtual processor. For a micro-partition environment, the number of logical processors is also displayed with the entitled processor capacity for the partition.

Following is an example of the **mpstat** command using the **-s** flag.

```
# mpstat -s 1

System configuration: lcpu=4 ent=0.5

     Proc1            Proc0
      0.27%           49.63%
 cpu0    cpu2    cpu1    cpu3
  0.17%   0.10%   3.14%  46.49%
```

The **mpstat -s 1** command is run to get SMT statistics for a one second sample duration. First, the output provides the system information about the number of logical processors and the entitlement assigned to the logical partition. The statistics are organized into columns showing each virtual processor and the logical processors (threads) assigned to them for the partition.

In this example, the total assigned entitlement is 0.5, or 50% of a physical processor. This entitlement is distributed across the physical processors as 0.27% + 49.63% = 49.90%, which is approximately the total entitlement. Each physical processor's consumed entitlement is then redistributed again to logical processors, for example Proc0 = cpu1 + cpu3, that is 3.14% + 46.49% = 49.63%.

SMT tuning

A new option has been added to the **schedo** command in AIX 5L Version 5.3 for tuning SMT. The *smt_snooze_delay* parameter may be used to specify the amount of time a thread will spin in an idle loop before sending a cede call to the POWER hypervisor.

A cede call is a special hypervisor call that allows an operating system to give back processor resource to the hardware when it no longer has a demand for it. It allows shared processors the opportunity to do other work instead of servicing, for example, a wait loop.

In *dedicated partitions*, the processor will transition to ST mode if, while two threads are active, the hypervisor receives a cede hcall (see 3.2.2, “POWER Hypervisor processor dispatch” on page 46 for hcall information) from the operating system for one of the threads. The inactive thread will be put into a dormant state, eliminating any processing overhead introduced with SMT and allowing the remaining active thread to run faster. If a value of -1 is specified, the thread will not cede while waiting in the idle loop. If the snooze delay is set to 0, the thread will cede immediately upon entering an idle wait. If a cede call is received from the last active thread, the processor is made available for hypervisor cycles if needed, or returned to the operating system immediately.

For a *shared partition*, if more than one thread is active and a cede call is sent to the hypervisor, the sending thread is put in a dormant state. If a cede call is received from the last active thread, the processor is deemed vacated.

To maximize speed, a value of 0 should be used. For maximum throughput, a value of -1 should be used.

```
# schedo -a
           v_repage_hi = 0
         v_repage_proc = 4
            v_sec_wait = 1
         v_min_process = 2
         v_exempt_secs = 2
              pacefork = 10
               sched_D = 16
               sched_R = 16
             timeslice = 1
               maxspin = 16384
              %usDelta = 100
          affinity_lim = 7
idle_migration_barrier = 4
       fixed_pri_global = 0
          big_tick_size = 1
              force_grq = 0
       smt_snooze_delay = 0
  sidle_setnewrq_mload = 384
  pidle_setnewrq_mload = 0
 optimistic_ficd_setrq = 0
  sidle_steal_S1_mload = 134
  sidle_steal_S2_mload = 134
 search_globalrq_mload = 256
  search_smtrunq_mload = 256
```

```
search_primrunq_mload = 64
  shed_primrunq_mload = 64
      bind_switchload = 0
       unboost_inflih = 1
   n_idle_loop_vlopri = 0
      hotlocks_enable = 0
        krlock_enable = 1
 krlock_conferb4alloc = 0
   krlock_spinb4alloc = 1
   krlock_confer2self = 1
  krlock_spinb4confer = 1024
   slock_spinb4confer = 1024
```

3

Virtualization engine technologies on p5 servers

This chapter discusses the virtualization engine technologies that are now integrated into the p5 servers, AIX 5L Version 5.3, and Linux.

These technologies include:

- POWER Hypervisor
- Micro-Partitioning
- Virtual Ethernet
- Shared Ethernet Adapter
- Virtual SCSI

The first part of the discussion is how these features are packaged and their relationship between the hardware and the operating system. The remainder of the discussion explains the virtual services provided by a combination of hardware, software, and hardware features.

3.1 Advanced POWER Virtualization feature

This section provides information about the packaging and ordering information for the Advanced POWER Virtualization feature available on p5 systems.

The Advanced POWER Virtualization feature is a combination of hardware enablement that includes the following components that are available together as a single priced feature:

- Firmware enablement for Micro-Partitioning
- Installation image for the Virtual I/O Server software which supports:
  - Shared Ethernet Adapter
  - Virtual SCSI server
- Partition Load Manager

Note that virtual Ethernet is available without this feature when the server is attached to an HMC as well as LPAR and dynamic LPAR. SMT is available on the base hardware with no additional features required.

Figure 3-1 shows a detailed overview of the different parts of the hardware order that enables firmware and includes the software orders.

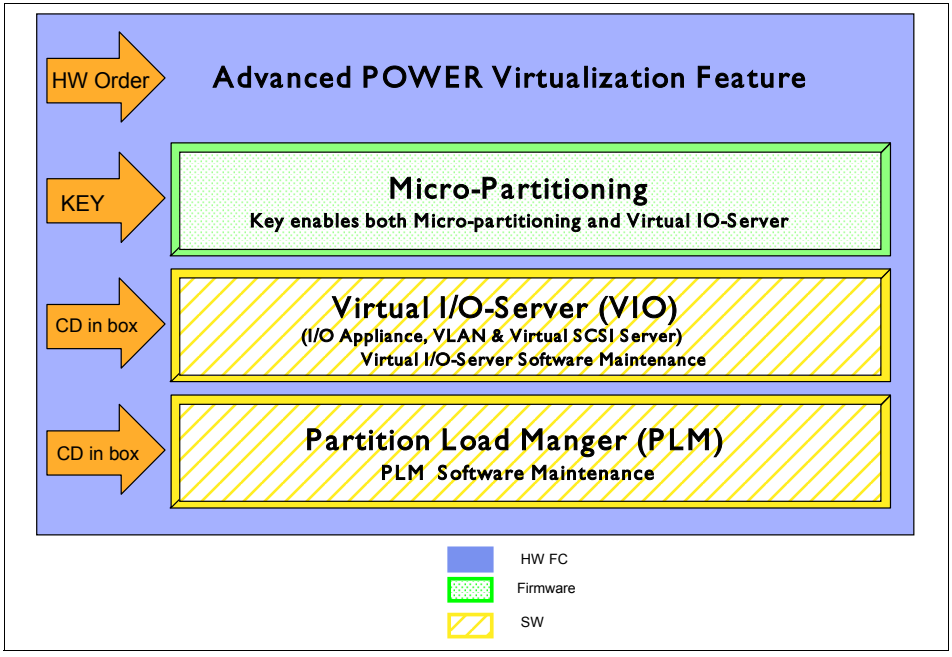

Figure 3-1 Advanced POWER Virtualization feature

When the hardware feature is specified with the initial system order, the firmware is shipped activated to support Micro-Partitioning and the Virtual I/O Server. For upgrade orders, IBM will ship a key to enable the firmware similar to the CUoD key.

Figure 3-2 shows the HMC panel where you enable the *Virtualization Engine Technologies*.

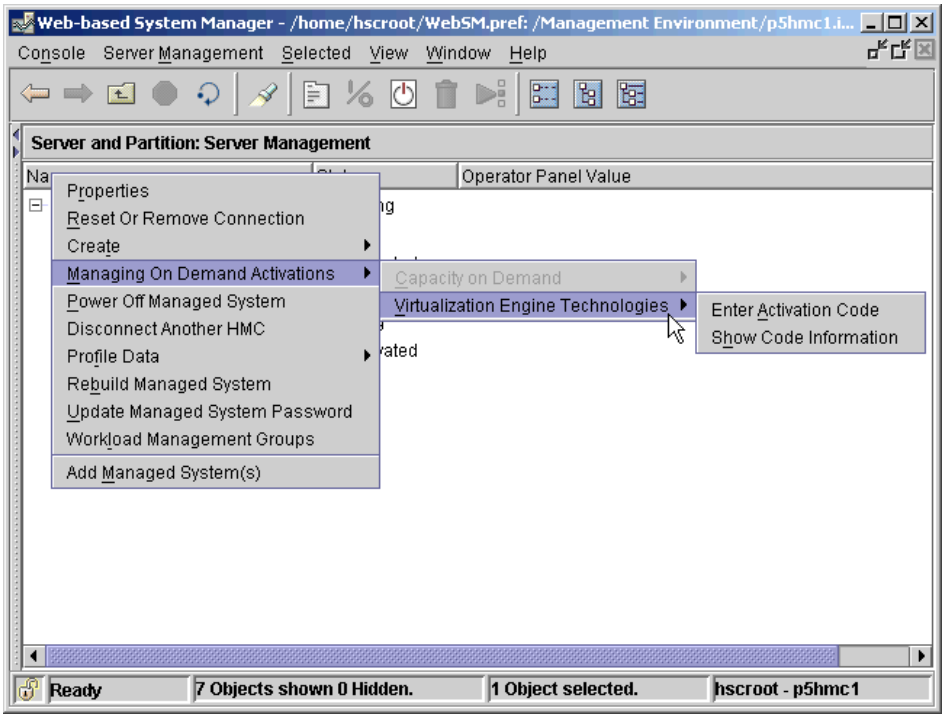

Figure 3-2 HMC panel to enable the Virtualization Engine Technologies

Virtual I/O Server and Partition Load Manager are licensed software components of the Advanced POWER Virtualization feature. They contain one charge unit per installed processor, including software maintenance. The initial software license charge for Virtual I/O Server and PLM is included in the price of the Advanced POWER Virtualization feature. The related hardware features that include Virtual I/O Server and PLM are:

- 9111-520 Feature 7940 (also the form number of this publication)
- 9113-550 Feature 7941
- 9117-570 Feature 7942

For each Virtual I/O Server V1.1 license ordered, an order for either the one-year (5771-VIO) or three-year (5773-VIO) Software Maintenance (SWMA) is also submitted.

The processor-based license enables you to install multiple Virtual I/O Server partitions on a single server to provide redundancy and to spread the Virtual I/O Server workload across multiple partitions.

The Virtual I/O Server resides in a POWER5 partition as a single function appliance that is created using the Hardware Management Console (HMC). The Virtual I/O Server installation media ships with the POWER5 system that is configured with the Advanced POWER Virtualization feature and supports network install (NIMOL from HMC) or CD installation. It supports AIX 5L Version 5.3, SUSE LINUX Enterprise Server 9 for POWER, and Red Hat Enterprise Linux AS for POWER Version 3 as Virtual I/O clients.

The Virtual I/O Server provides the Virtual SCSI server and Shared Ethernet Adapter virtual I/O function to client partitions (Linux or AIX). This POWER5 partition is not intended to run applications or for general user login.

For each Partition Load Manager V1.1 (5765-G31) license ordered, an order for either the one-year (5771-PLM) or three-year (5773-PLM) Software Maintenance (SWMA) must also be submitted. The Software Maintenance for the Partition Load Manager is priced on a per processor basis, by processor group.

The Partition Load Manager for AIX 5L helps customers to maximize the utilization of processor and memory resources on pSeries and p5 servers that support dynamic logical partitioning. Within the constraints of a user-defined policy, resources are automatically moved to partitions with a high demand, from partitions with a lower demand. Resources that would otherwise go unused can now be more fully utilized.

Partition Load Manager supports management of any dynamic LPAR running AIX 5L Version 5.2 with the 5200-04 Recommended Maintenance Package or AIX 5L Version 5.3 or later. The Partition Load Manager server can run on any system with the appropriate level of IBM AIX.

The summary of features is as follows:

- Automated processor and memory reconfiguration
- Real-time partition configuration and load statistics
- Support for dedicated and shared processor partition groups
- Support for manual provisioning of resources
- Graphical user interface (Web-based System Manager)

3.2 Introduction to the POWER Hypervisor

The POWER Hypervisor is an essential element of the IBM Virtualization Engine system technologies implemented in the POWER5 processor based @server family of products. Combined with features designed into the POWER5 processor, the POWER Hypervisor delivers functions that enable other system technologies including Micro-Partitioning, virtualized processors, IEEE VLAN compatible virtual switch, virtual SCSI adapters, and virtual consoles.

The POWER Hypervisor is a component of system firmware that will always be installed and activated, regardless of system configuration. It operates as a hidden partition, with no processor resources assigned to it.

Newly architected *hypervisor calls* (hcalls) provide a means for the operating system to communicate with the POWER Hypervisor, allowing more efficient use of physical processor capacity.

The POWER Hypervisor is a key component of the functions shown in Figure 3-3. It performs the following tasks:

- Provides an abstraction layer between the physical hardware resources and the logical partitions using them
- Enforces partition integrity by providing a security layer between logical partitions
- Controls the dispatch of virtual processors to physical processors
- Saves and restores all processor state information during logical processor context switch
- Controls hardware I/O interrupts management facilities for logical partitions

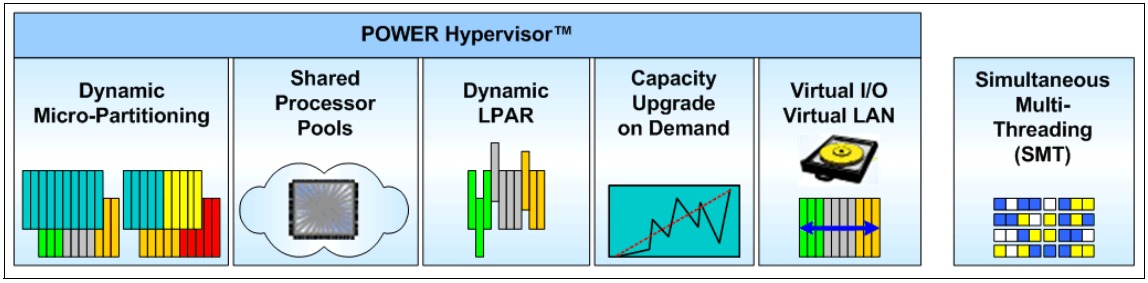

Figure 3-3 POWER Hypervisor functions

The POWER Hypervisor, acting as the abstraction layer between the system hardware and the logical partitions, allows multiple operating systems to run on POWER5 technology with little or no modifications.

3.2.1 POWER Hypervisor implementation

The POWER4 processor introduced support for logical partitioning with a new privileged processor state called *hypervisor mode*. It is accessed using hypervisor call functions, which are generated by the operating system kernel running in a partition. Hypervisor mode allows for a secure mode of operation that is required for various system functions where logical partition integrity and security are required. The hypervisor validates that the partition has ownership of the resources it is attempting to access, such as processor, memory, and I/O, then completes the function. This mechanism allows for complete isolation of partition resources.

In the POWER5 processor, further design enhancements are introduced that enable the sharing of processors by multiple partitions. The *hypervisor decrementer* (HDECR) is a new hardware facility in the POWER5 design that is programmed to provide the POWER Hypervisor with a timed interrupt independent of partition activity. HDECR interrupts are routed directly to the POWER Hypervisor, and use only POWER Hypervisor resources to capture state information from the partition. The HDECR is used for fine-grained dispatching of multiple partitions on shared processors. It also provides a means for the POWER Hypervisor to dispatch physical processor resources for its own execution.

With the addition of shared partitions and SMT, a mechanism was required to track physical processor resource utilization at a processor thread level. System architecture for POWER5 introduces a new register called the *Processor Utilization Resource Register* (PURR) to accomplish this. It provides the partition with an accurate cycle count to measure activity during timeslices dispatched on a physical processor. The PURR is a POWER Hypervisor resource, assigned one per processor thread, that is incremented at a fixed rate whenever the thread running on a virtual processor is dispatched on a physical processor.

3.2.2 POWER Hypervisor processor dispatch

Multiple logical partitions configured to run with a pool of shared physical processors require a robust mechanism to guarantee the distribution of available processing cycles. The POWER Hypervisor manages this task in the POWER5 processor based system.

Shared processor partition resource definition is discussed in detail in 3.3, “Micro-Partitioning introduction” on page 54. This section introduces the concepts of how the POWER Hypervisor dispatches work from multiple partitions across physical processors.

Each shared processor partition is configured with a specific processor entitlement, based on a quantity of processing units, which is referred to as the partition's *entitled capacity*. The entitled capacity, along with a defined number of virtual processors, defines the physical processor resource that will be allotted to the partition. The POWER Hypervisor uses the POWER5 HDECR, which is programmed to generate an interrupt every 10 ms, as a timing mechanism for controlling the dispatch of physical processors to system partitions. Each virtual processor is guaranteed to get its entitled share of processor cycles during each 10 ms dispatch window.

Primary POWER Hypervisor hcalls

The primary POWER Hypervisor hcalls used by the operating system in the dispatch of a virtual processor are:

cede The cede call is used when a virtual processor or SMT thread becomes idle, allowing the POWER Hypervisor to dispatch other work.

confer The confer call is used to grant the remaining cycles in a dispatch interval to another virtual processor in the partition. It may be used when one virtual processor cannot make forward progress because it is waiting on an event to complete on another virtual processor, such as a lock miss.

prod The prod call is used to activate a virtual processor that has ceded or conferred processor cycles.

A virtual processor will always be in one of four logical states. These states are:

running Currently dispatched on a physical processor

runnable Ready to run, waiting for dispatch

not-runnable Has ceded or conferred its cycles

expired Consumed its full entitled cycles for the current dispatch window

Monitoring POWER Hypervisor hcalls

In AIX 5L Version 5.3, the **lparstat** command using the **-h** and **-H** flags will display hypervisor statistical data about many hcalls, including cede, confer, and prod. Using the **-h** flag adds summary hypervisor statistics to the default **lparstat** output.

The following shows an example of this command, collecting statistics for one five-second interval.

```
# lparstat -h 5 1
System configuration: type=Shared mode=Uncapped smt=On lcpu=4 mem=512 ent=0.50
%user  %sys  %wait  %idle physc %entc  lbusy   app  vcsw phint  %hypv hcalls
-----  ----  -----  ----- ----- ----- ------   ---  ---- -----  ----- ------
  0.0   0.5    0.0   99.4  0.00   1.0    0.0     -  1524     0    0.5   1542
```

Using the **-H** flag displays detailed hypervisor information, including statistics for many hcall functions. The command shows the following for each of these hcalls:

Number of calls Number of hypervisor calls made

Total Time Spent Percentage of total time spent in this type of call

Hypervisor Time Spent Percentage of hypervisor time spent in this type of call

Average Call Time Average call time for this type of call in nano-seconds

Maximum Call Time Maximum call time for this type of call in nano-seconds

```
# lparstat -H 5 1

System configuration: type=Shared mode=Uncapped smt=On lcpu=4 mem=512 ent=0.50

           Detailed information on Hypervisor Calls

Hypervisor        Number of    %Total Time   %Hypervisor   Avg Call    Max Call
  Call              Calls         Spent      Time Spent    Time(ns)    Time(ns)

remove                   2            0.0           0.0         417         550
read                    21            0.0           0.0         148         294
nclear_mod               0            0.0           0.0           1           0
page_init                3            0.0           0.0         774        2280
clear_ref                0            0.0           0.0           1           0
protect                  0            0.0           0.0           1           0
put_tce                 14            0.0           0.1         544         908
xirr                    10            0.0           0.0         536         758
eoi                     10            0.0           0.0         526         695
ipi                      0            0.0           0.0           1           0
cppr                     0            0.0           0.0           1           0
asr                      0            0.0           0.0           1           0
others                   0            0.0           0.0           1           0
enter                    5            0.0           0.0         397         685
cede                  1595            0.5          99.6        7918     1343207
migrate_dma              0            0.0           0.0           1           0
put_rtce                 0            0.0           0.0           1           0
confer                   0            0.0           0.0           1           0
prod                    27            0.0           0.1         672         922
get_ppp                  1            0.0           0.0        1502        2579
set_ppp                  0            0.0           0.0           1           0
```

```
purr                    0          0.0          0.0          1          0
pic                     1          0.0          0.0        309        410
bulk_remove             0          0.0          0.0          1          0
send_crq                0          0.0          0.0          1          0
copy_rdma               0          0.0          0.0          1          0
get_tce                 0          0.0          0.0          1          0
send_logical_lan        0          0.0          0.0          1          0
add_logicl_lan_buf      0          0.0          0.0          1          0
--------------------------------------------------------------------------------
```

The basic concept of this dispatch mechanism is illustrated in Figure 3-4. In this figure there are three logical partitions defined, sharing the processor cycles from two physical processors, spanning two 10 ms hypervisor dispatch intervals.

Logical partition 1 is defined with an entitlement capacity of 0.8 processing units, with two virtual processors. This allows the partition 80% of one physical processor for each 10 ms dispatch window for the shared processor pool. For each dispatch window, the workload is shown to use 40% of each physical processor during each dispatch interval. It is possible for a virtual processor to be dispatched more than one time during a dispatch interval. Note that in the first dispatch interval, the workload executing on virtual processor 1 is not a continuous utilization of physical processor resource. This can happen if the operating system confers cycles, and is reactivated by a prod hcall.

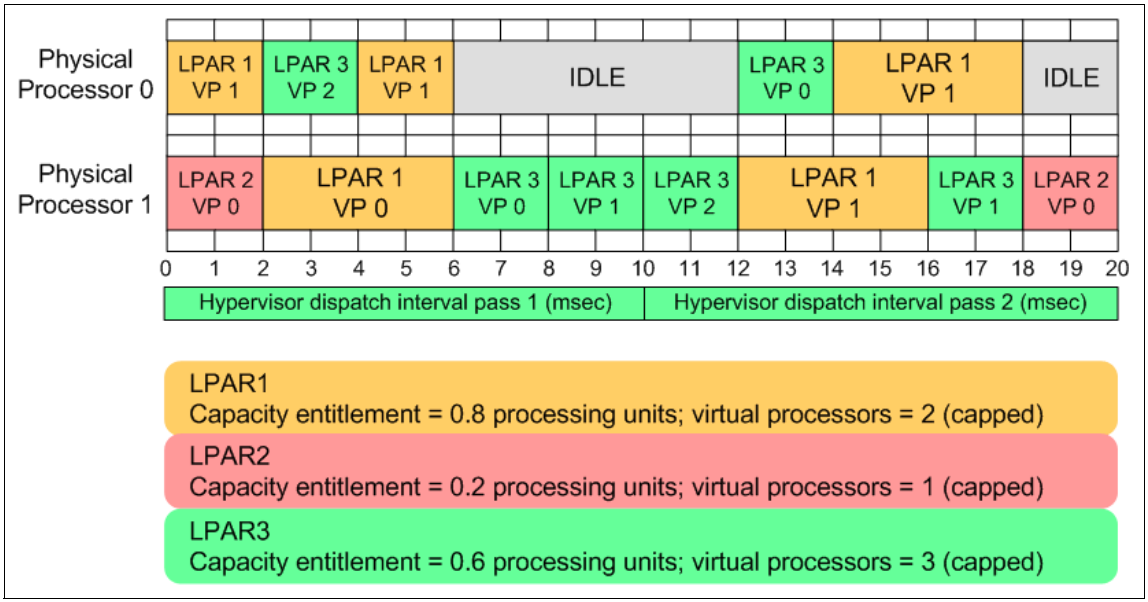

Figure 3-4 Micro-Partitioning processor dispatch

Logical partition 2 is configured with one virtual processor and a capacity of 0.2 processing units, entitling it to 20% usage of a physical processor during each dispatch interval. In this example, a worst case dispatch latency is shown for this virtual processor, where the 2 ms are used in the beginning of dispatch interval 1 and the last 2 ms of dispatch interval 2, leaving 16 ms between processor allocation.

Logical partition 3 contains 3 virtual processors, with an entitled capacity of 0.6 processing units. Each of the partition's three virtual processors consumes 20% of a physical processor in each dispatch interval, but in the case of virtual processor 0 and 2, the physical processor they run on changes between dispatch intervals. The POWER Hypervisor attempts to maintain physical processor affinity when dispatching virtual processors. It will always first try to dispatch the virtual processor on the same physical processor as it last ran on, and depending on resource utilization will broaden its search out to the other processor on the POWER5 chip, then to another chip on the same MCM, then to a chip on another MCM.

Processor dispatch communication

The dispatch mechanism utilizes hcalls to communicate between the operating system and the POWER Hypervisor. Implementing hcalls in the operating system is desirable for performance reasons since they minimize the idle time of a physical processor. Operating systems must be designed to use this new call to exploit the full potential of POWER5.

When a virtual processor is active on a physical processor and the operating system detects an inability to utilize processor cycles, it may *cede* or *confer* its cycles back to the POWER Hypervisor, enabling it to schedule another virtual processor on the physical processor for the remainder of the dispatch cycle. Reasons for a cede or confer may include the virtual processor running out of work and becoming idle, entering a spin loop to wait for a resource to free, or waiting for a long latency access to complete. There is no concept of credit for cycles that are ceded or conferred. Entitled cycles not used during a dispatch interval are lost.

A virtual processor that has ceded cycles back to the POWER Hypervisor can be reactivated using a *prod* hcall. If the operating system running on another virtual processor within the logical partition detects that work is available for one of its idle processors, it can use the prod hcall to signal the POWER Hypervisor to make the prodded virtual processor runnable again. Once dispatched, this virtual processor would resume execution at the return from the cede hcall.

In IBM AIX 5L Version 5.3, the **mpstat** command using the **-d** flag displays detailed affinity and migration statistics for AIX threads and dispatching statistics for logical processors.

```
# mpstat -d
System configuration: lcpu=4 ent=0.5

cpu     cs    ics  bound  rq push S3pull S3grd  S0rd  S1rd  S2rd  S3rd  S4rd  S5rd   ilcs   vlcs
  0  68598  38824      0   0    0      0     0  95.6   0.0   0.0   4.4   0.0   0.0 174110 237393
  1    291    244      0   0    0      0     0  90.9   7.4   0.0   1.7   0.0   0.0   1092 237759
  2  54514  30174      1   1    0      0     0  94.0   0.1   0.0   6.0   0.0   0.0   2756  71779
  3    751    624      0   0    0      0     0  91.3   2.9   0.0   5.8   0.0   0.0   1192  72971
ALL 124154  69866      1   1    0      0     0  94.8   0.1   0.0   5.1   0.0   0.0  89575 309951
```

The POWER Hypervisor dispatch affinity domains are defined as follows, and statistics for virtual processor dispatch across these domains is given by the **mpstat** command.

S0 The process redispatch occurs within the same logical processor. This happens in the case of SMT-enabled systems.

S1 The process redispatch occurs within the same physical processor, among different logical processors. This involves sharing of the L1, L2, and L3 cache.

S2 The process redispatch occurs within the same processor chip, but among different physical processors. This involves sharing of the L2 and L3 cache.

S3 The process redispatch occurs within the same MCM module, but among different processor chips.

S4 The process redispatch occurs within the same CEC plane, but among different MCM modules. This involves access to the main memory or L3-to-L3 transfer.

S5 The process redispatch occurs outside of the CEC plane.

As previously stated, the POWER Hypervisor will always first try to dispatch the virtual processor on the same physical processor that it last ran on, and depending on resource utilization will broaden its search out to the other processor on the POWER5 chip, then to another chip on the same MCM, then to a chip on another MCM.

Systems using DCMs share similar boundaries between processor cards, or SMP Flex cables, as MCMs do between MCMs.

3.2.3 POWER Hypervisor and virtual I/O

Detailed discussion of POWER5 *@server* virtual I/O implementation is found in various sections of this publication. This section introduces POWER Hypervisor involvement in the virtual I/O functions.

With the introduction of Micro-Partitioning, the ability to dedicate physical hardware adapter *slots* to each partition becomes impractical. Virtualization of I/O devices allows many partitions to communicate with each other, and access networks and storage devices external to the server, without dedicating physical I/O resources to an individual partition. Many of the I/O virtualization capabilities introduced with the POWER5 processor based *@server* products are accomplished by functions designed into the POWER Hypervisor.

The POWER Hypervisor does not *own* any physical I/O devices, and it does not provide virtual interfaces to them. All physical I/O devices in the system are owned by logical partitions. Virtual I/O devices are owned by the Virtual I/O Server, which provides access to the real hardware that the virtual device is based on.

The POWER Hypervisor implements the following operations required by system partitions to support virtual I/O:

- Provides control and configuration structures for virtual adapter images required by the logical partitions
- Operations that allow partitions controlled and secure access to physical I/O adapters in a different partition

Along with these operations, the POWER Hypervisor allows for the virtualization of I/O interrupts. To maintain partition isolation, the POWER Hypervisor controls the hardware interrupt management facilities. Each logical partition is provided controlled access to the interrupt management facilities using hcalls. Virtual I/O adapters and real I/O adapters use the same set of hcall interfaces.

I/O types supported

Three types of virtual I/O adapters are supported by the POWER Hypervisor:

- SCSI
- Ethernet
- Serial console (virtual TTY)

Virtual I/O adapters are defined by system administrators during logical partition definition. Configuration information for the virtual adapters is presented to the partition operating system by the system firmware.

Virtual SCSI support

The p5 server uses SCSI as the mechanism for virtual storage devices. This is accomplished using two paired adapters: a virtual SCSI server adapter and a virtual SCSI client adapter. These adapters are used to transfer SCSI commands between partitions. To the operating system, the virtual SCSI client adapter is no different than a typical SCSI adapter. The SCSI server, or target, adapter is responsible for executing any SCSI command it receives. It is owned by the Virtual I/O server partition. How this SCSI command is executed is dependent on the hosting partition operating system. See 3.6, “Virtual SCSI introduction” on page 87 for a detailed discussion of virtual SCSI.

Virtual SCSI operation is dependant on two system functions implemented in the POWER Hypervisor.

- Message queuing function that enables the passing of messages between partitions, with an interrupt mechanism to confirm receipt of the message
- DMA function between partitions that provides control structures for secure transfers between logical partition memory spaces

Virtual Ethernet switch support

The POWER Hypervisor provides a Virtual Ethernet switch function that allows partitions on the same server a means for fast and secure communication. It is based on the IEEE 802.1Q VLAN standard. No physical I/O adapter is required when creating a VLAN connection between partitions, and no access to an outside network is required. Each partition operating system sees the VLAN switch as an Ethernet adapter, without the physical link properties and asynchronous data transmit operations. See 3.4, “Virtual Ethernet introduction” on page 73 for a detailed discussion of Virtual Ethernet.

Virtual Ethernet transmit operations are performed synchronously and are complete as soon as the hypervisor call to send a frame returns. Receive operations are performed using a pool of buffers provided to the POWER Hypervisor for receiving frames. As incoming frames are received by the adapter, the POWER Hypervisor sends a virtual interrupt back to the device driver.

Each Virtual Ethernet adapter can also be configured as a trunk adapter. This enables layer-2 bridging to a physical Ethernet adapter to be performed, extending the Virtual Ethernet network outside the server to a physical Ethernet network. If a partition sends an Ethernet frame to a MAC address that is unknown by the POWER Hypervisor virtual switch, it will be forwarded to any trunk adapter defined for the same VLAN ID.

Virtual TTY console support

Each partition needs to have access to a system console. Tasks such as operating system install, network setup, and some problem analysis activities require a dedicated system console. The POWER Hypervisor provides virtual console using a virtual TTY or serial adapter and a set of hypervisor calls to operate on them.

Depending on the system configuration, the operating system console can be provided by the Hardware Management Console (HMC) virtual TTY, or from a terminal emulator connected to physical serial ports on the system's service processor.

3.3 Micro-Partitioning introduction

The concept of Micro-Partitioning is to allow the resource definition of a partition to allocate fractions of processors to the partition.

- On POWER4 systems, all partitions are considered *dedicated*, in that the processors assigned to a partition can only be in whole multiples and only used by that partition.
- On POWER5 systems, you can choose between dedicated processor partitions and shared processor partitions using Micro-Partitioning.

The benefit of Micro-Partitioning is that it allows increased overall utilization of system resources by automatically applying only the required amount of processor resource needed by each partition. Sales material tends to discuss Micro-Partitioning in terms of fractional processor units; however, resource can also be defined as increments greater than a single processor.

The POWER Hypervisor continually adjusts the amount of processor capacity allocated to each shared processor partition and any excess capacity unallocated based on current partition profiles within a shared pool. Tuning parameters allow the administrator extensive control over the amount of processor resources that each partition can use.

This section discusses the following topics about Micro-Partitioning:

- Shared processor partitions
- Shared pool overview
- CUoD
- Dynamic LPAR
- Limitations of Micro-Partitioning

3.3.1 Shared processor partitions

The virtualization of processors enables the creation of a partitioning model which is fundamentally different from the POWER4 systems, where whole processors are assigned to partitions and are owned by them. In the new model, physical processors are abstracted into virtual processors that are then assigned to partitions, but the underlying physical processors are shared by these partitions.

Virtual processor abstraction is implemented in the hardware and microcode. From an operating system perspective, a virtual processor is indistinguishable from a physical processor. The key benefit of implementing partitioning in the hardware is to allow any operating system to run on POWER5 technology with little or no changes. Optionally, for optimal performance, the operating system can be enhanced to exploit shared processor pools more in-depth, for instance, by voluntarily relinquishing CPU cycles to the hardware when they are not needed. AIX 5L Version 5.3 is the first version of AIX 5L that includes such enhancements.

Micro-Partitioning allows for multiple partitions to share one physical processor. Partitions using Micro-Partitioning technology are referred to as shared processor partitions.

A partition may be defined with a processor capacity as small as 10 processor units. This represents 1/10 of a physical processor. Each processor can be shared by up to 10 shared processor partitions. The shared processor partitions are dispatched and time-sliced on the physical processors under control of the POWER Hypervisor.

Micro-Partitioning is supported across the entire POWER5 product line from the entry to the high-end systems. Table 3-1 shows the maximum number of logical partitions and shared processor partitions supported on the different models.

Table 3-1 Micro-Partitioning overview on p5 systems

p5 servers	Model 520	Model 550	Model 570
Processors	2	4	16
Dedicated processor partitions	2	4	16
Shared processor partitions	20	40	160

It is important to point out that the maximums stated are supported by the hardware, but the practical limits based on production workload demands may be significantly lower.

Shared processor partitions still need dedicated memory, but the partitions' I/O requirements can be supported through Virtual Ethernet and Virtual SCSI. Utilizing all virtualization features, support for up to 254 shared processor partitions is planned.

The shared processor partitions are created and managed by the HMC. When you start creating a partition you have to choose between a shared processor partition and a dedicated processor partition.

When setting up a partition you have to define the resources that belong to the partition, such as memory and I/O resources. For processor shared partitions you have to configure these additional options:

- Minimum, desired, and maximum processing units of capacity
- The processing sharing mode, either capped or uncapped

  If the partition is uncapped, specify its variable capacity weight
- Minimum, desired, and maximum virtual processors

These settings are the topic of the following sections.

Processing units of capacity

Processing capacity can be configured in fractions of 1/100 of a processor. The minimum amount of processing capacity which has to be assigned to a partition is 1/10 of a processor.

On the HMC, processing capacity is specified in terms of *processing units*. The minimum capacity of 1/10 of a processor is specified as 0.1 processing units. To assign a processing capacity representing 75% of a processor, 0.75 processing units are specified on the HMC.

On a system with two processors a maximum of 2.0 processing units can be assigned to a partition. Processing units specified on the HMC are used to quantify the minimum, desired, and maximum amount of processing capacity for a partition.

Once a partition is activated, processing capacity is usually referred to as capacity entitlement or entitled capacity.

Figure 3-5 on page 57 shows a graphical view of the definitions of processor capacity.

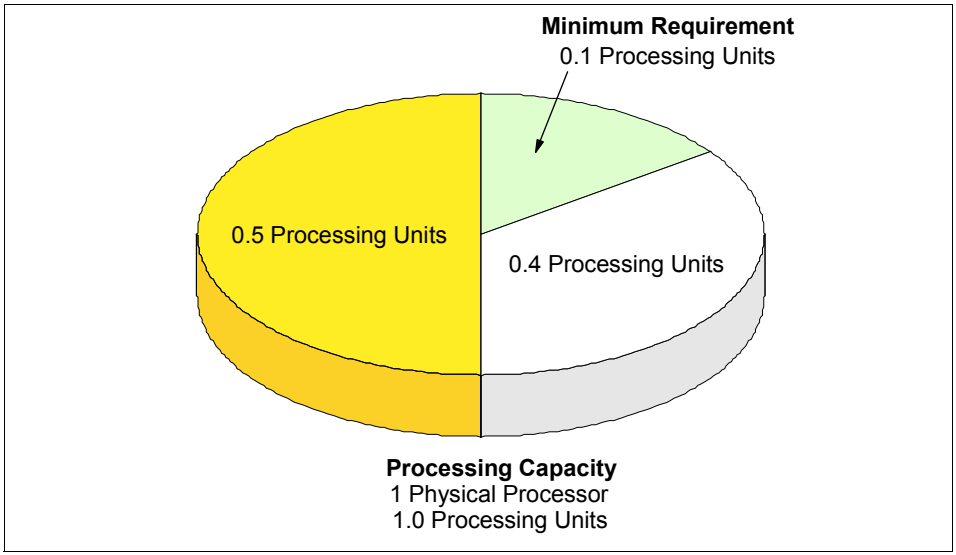

Figure 3-5 Processing units of capacity

Capped and uncapped mode

The next step in defining a shared processor partition is to specify whether the partition is running in a capped or uncapped mode.

Capped mode The processor unit never exceeds the assigned processing capacity.

Uncapped mode The processing capacity may be exceeded when the shared processing pools have available resource.

When a partition is run in an uncapped mode, you must specify the uncapped weight of that partition.

If multiple uncapped logical partitions require idle processing units, the managed system distributes idle processing units to the logical partitions in proportion to each logical partition's uncapped weight. The higher the uncapped weight of a logical partition, the more processing units the logical partition gets.

The uncapped weight must be a whole number from 0 to 255. The default uncapped weight for uncapped logical partitions is 128. A partition's share is computed by dividing its variable capacity weight by the sum of the variable capacity weights for all uncapped partitions. If you set the uncapped weight at 0, the managed system treats the logical partition as a capped logical partition. A logical partition with an uncapped weight of 0 cannot use more processing units than those that are committed to the logical partition.

Virtual processors

Virtual processors are the whole number of concurrent operations that the operating system can use. The processing power can be conceptualized as being spread equally across these virtual processors. Selecting the optimal number of virtual processors depends on the workload in the partition. Some partitions benefit from greater concurrence, whereas other partitions require greater power.

By default, the number of processing units that you specify is rounded up to the minimum number of virtual processors needed to satisfy the assigned number of processing units. The default settings maintain a balance of virtual processors to processor units. For example:

- If you specify 0.50 processing units, one virtual processor will be assigned.
- If you specify 2.25 processing units, three virtual processors will be assigned.

You also can use the Advanced tab in your partitions profile to change the default configuration and to assign more virtual processors.

At the time of publication, the maximum number of virtual processors per partition is 64.

A logical partition in the shared processing pool will have at least as many virtual processors as its assigned processing capacity. By making the number of virtual processors too small, you limit the processing capacity of an uncapped partition. If you have a partition with 0.50 processing units and 1 virtual processor, the partition cannot exceed 1.00 processing units because it can only run one job at a time, which cannot exceed 1.00 processing units. However, if the same partition with 0.50 processing units was assigned two virtual processors and processing resources were available, the partition could use an additional 1.50 processing units.

Dedicated processors

Dedicated processors are whole processors that are assigned to a single partition. If you choose to assign dedicated processors to a logical partition, you must assign at least one processor to that partition.

You cannot mix shared processors and dedicated processors in one partition.

By default, a powered-off logical partition using dedicated processors will have its processors available to the shared processing pool. When the processors are in the shared processing pool, an uncapped partition that needs more processing power can use the idle processing resources. However, when you power on the dedicated partition while the uncapped partition is using the processors, the activated partition will regain all of its processing resources. If you want to

prevent dedicated processors from being used in the shared processing pool, you can disable this function using the logical partition profile properties panels on the Hardware Management Console.

Note: You cannot disable the "`Allow idle processor to be shared`" function when you create a partition. You need to open the properties for the created partition and change it on the Processor tab.

3.3.2 Shared pool overview

The POWER Hypervisor schedules shared processor partitions from a set of physical processors that is called the shared processor pool. By definition, these processors are not associated with dedicated partitions.

In shared partitions there is no fixed relationship between virtual processors and physical processors. The POWER Hypervisor can use any physical processor in the shared processor pool when it schedules the virtual processor. By default, it attempts to use the same physical processor, but this cannot always be guaranteed. The POWER Hypervisor uses the concept of a home node for virtual processors, enabling it to select the best available physical processor from a memory affinity perspective for the virtual processor that is to be scheduled.

Affinity scheduling is designed to preserve the content of memory caches, so that the working data set of a job can be read or written in the shortest time period possible. Affinity is actively managed by the POWER Hypervisor since each partition has a completely different context. Currently, there is one shared processor pool, so all virtual processors are implicitly associated with the same pool.

Figure 3-6 on page 60 shows the relationship between two partitions using a shared processor pool of a single physical CPU. One partition has two virtual processors and the other a single one. The figure also shows how the capacity entitlement is evenly divided over the number of virtual processors.

When you set up a partition profile, you set up the desired, minimum, and maximum values you want for the profile. When a partition is started, the system chooses the partition's entitled processor capacity from this specified capacity range. The value that is chosen represents a commitment of capacity that is reserved for the partition. This capacity cannot be used to start another shared partition, otherwise capacity could be overcommitted.

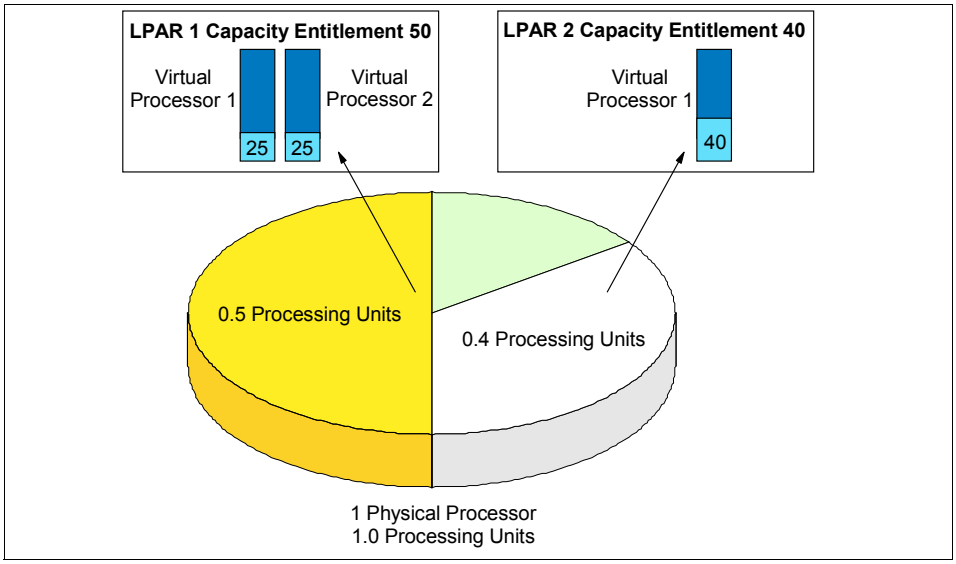

Figure 3-6 Distribution of capacity entitlement on virtual processors

When starting a partition, preference is given to the desired value, but this value cannot always be used because there may not be enough unassigned capacity in the system. In that case, a different value is chosen, which must be greater than or equal to the minimum capacity attribute. Otherwise, the partition cannot be started.

The entitled processor capacity is distributed to the partitions in the sequence the partitions are started. For example, consider a shared pool that has 2.0 processing units available.

Partitions 1, 2, and 3 are activated in sequence:

- Partition 1 activated
  Min. = 1.0, max = 2.0, desired = 1.5
  Allocated capacity entitlement: 1.5
- Partition 2 activated
  Min. = 1.0, max = 2.0, desired = 1.0
  Partition 2 does not start because the minimum capacity is not met.
- Partition 3 activated
  Min. = 0.1, max = 1.0, desired = 0.8
  Allocated capacity entitlement: 0.5

The maximum value is only used as an upper limit for dynamic operations.

Figure 3-7 shows the usage of a capped partition of the shared processor pool. Partitions using the capped mode are not able to assign more processing capacity from the shared processor pool than the capacity entitlement will allow.

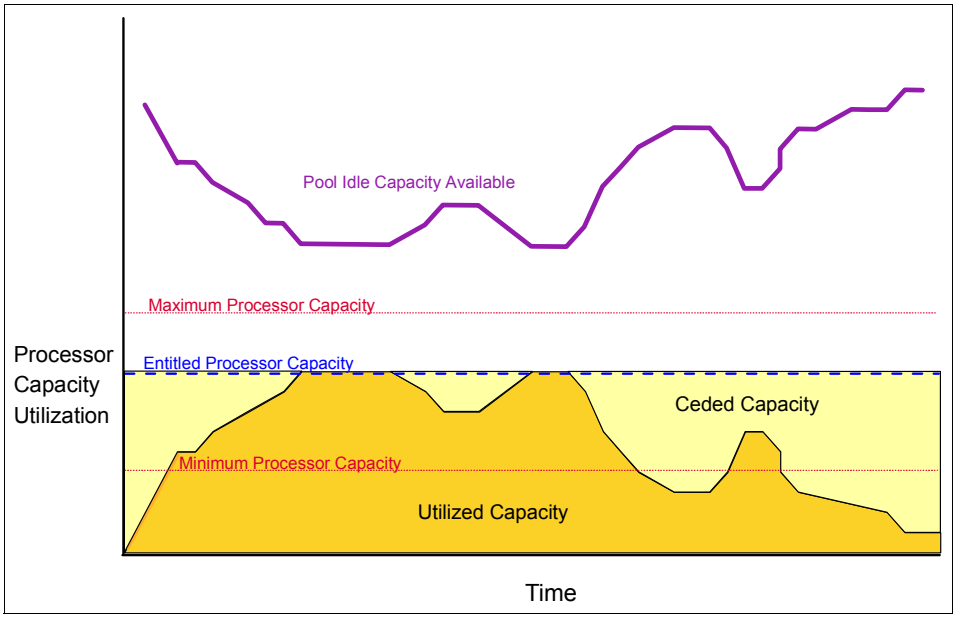

Figure 3-7 Capped shared processor partitions

Figure 3-8 on page 62 shows the usage of the shared processor pool by an uncapped partition. The uncapped partition is able to assign idle processing capacity if it needs more than the entitled capacity.

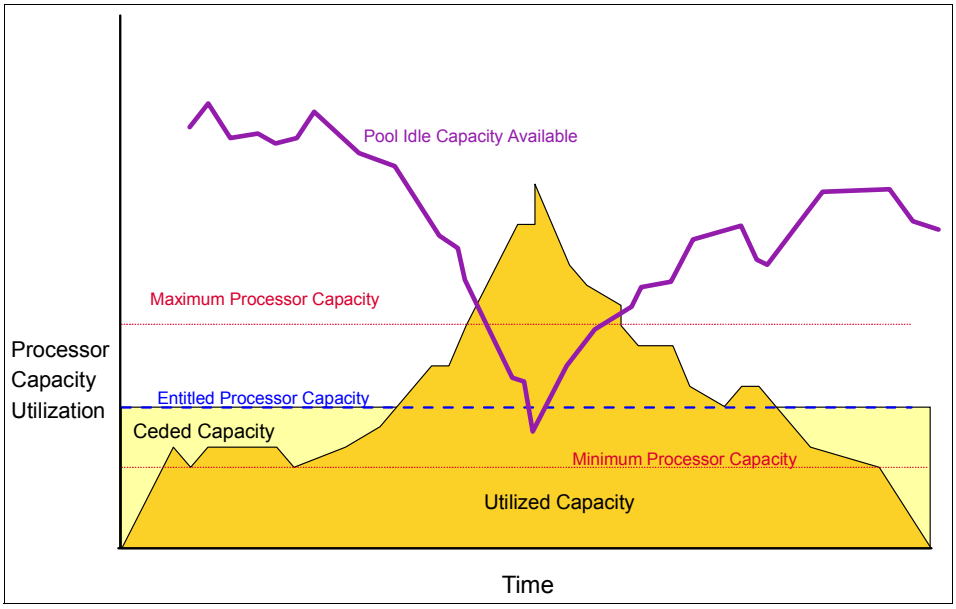

Figure 3-8 Uncapped shared processor partition

3.3.3 Capacity on Demand

Capacity on Demand (CoD) adds operational and configuration flexibility for IBM ℮server p5 and pSeries systems. Available for a fee, CoD allows additional resources to be added as they are needed. Processors and memory can be brought online to meet increasing workload demands. If the system is configured for dynamic LPAR, this can be accomplished without impacting operations.

When activating a processor featured for Capacity on Demand on a system with defined shared processor partitions, the activated processor is automatically assigned to the shared processor pool. You can then decide to add the processor dynamically to a dedicated processor partition or to dynamically add capacity entitlement to the shared processor partitions.

When the system operates in full system partition mode, the processor is automatically added to the system's processor capacity.

To remove a Capacity on Demand processor (for example, when using On/Off Capacity on Demand, which enables users to temporarily activate processors), you have to make sure that there are enough processing units left to deactivate the processor. You can dynamically remove the needed capacity entitlement from the partitions.

A type of Capacity on Demand is named “reserve CoD.” It represents an *autonomic* way to activate temporary capacity. Reserve CoD enables the user to place a quantity of inactive processors into the server's Shared Processor Pool which then become available to the pool's resource manager. When the server recognizes that the base (purchased/active) processors assigned across uncapped partitions have been 100% utilized, and at least 10% of an additional processor is needed, then a *Processor Day* (good for a 24 hour period) is charged against the Reserve CoD account balance. Another Processor Day will be charged for each additional processor put into use based on the 10% utilization rule. After a 24-hour period elapses, and there is no longer a need for the additional performance, no Processor Days will be charged until the next performance spike.

3.3.4 Dynamic processor deallocation and processor sparing

Dynamic processor deallocation and dynamic processor sparing are also supported in the shared processor pool.

If a physical processor reaches a failure threshold and needs to be taken offline (guarded out), the POWER Hypervisor will analyze the system environment to determine what action will be taken to replace the processor resource. The options for handling this condition are the following:

- If there is a CoD processor available, the POWER Hypervisor will transparently switch the processor to the shared pool, and no partition loss of capacity would result.
- If there is at least 1.0 unallocated processor capacity available, it can be used to replace the capacity lost due to the failing processor.

If not enough unallocated resource exists, the POWER Hypervisor will determine how much capacity each partition must lose to eliminate the 1.00 processor units from the shared pool. As soon as each partition varies off the processing capacity and/or virtual processors, the failing processor is taken offline by the service processor and hypervisor.

The amount of capacity that you request each shared partition to vary off is proportional to the total amount of entitled capacity in the partition. This is based on the amount of capacity that can be varied off, which is controlled by the *min* capacity of the partition. The larger the difference between the current entitled capacity of the partition and the minimum entitled capacity, the more the partition will be asked to vary off.

3.3.5 Dynamic partitioning

Dynamic partitioning was introduced with AIX 5L Version 5.2. An AIX 5L Version 5.2 based dynamic partition can consists of the following resource elements:

- A dedicated processor
- 256 MB memory region
- I/O adapter slot

Multiple resources can be placed under the exclusive control of a given logical partition. Dynamic LPAR extends these capabilities by allowing this fine-grained resource allocation to occur not only when activating a logical partition, but also while the partitions are running. Individual processors, memory regions, and I/O adapter slots can be released into a *free pool*, acquired from that free pool, or moved directly from one partition to another.

On POWER5 with AIX 5L Version 5.3, a partition can consist of dedicated processors, or virtual processors with a specific capacity entitlement running in capped or uncapped mode, dedicated memory region, and virtual or physical I/O adapter slots.

For dedicated and shared processor partitions it is possible to dynamically:

- Add, move, or remove memory in a granularity of 16 MB regions
- Add, move, or remove physical I/O adapter slots
- Add or remove virtual I/O adapter slots

For a dedicated processor partition it is only possible to dynamically add, move, or remove whole processors. When you dynamically remove a processor from a dedicated partition on a system that uses shared processor partitions it is then assigned to the shared processor pool.

For shared processor partitions it is also possible to dynamically:

- Remove, move, or add entitled shared processor capacity
- Change between capped and uncapped processing
- Change the weight of an uncapped partition
- Add and remove virtual processors

Figure 3-9 on page 65 shows the panel for dynamic reconfiguration of the processor resources on the HMC. Here you can choose to add, remove, or move your resources. Select the partition that you want to change dynamically and press the right mouse button. Then choose **Dynamic Logical Partitioning** from the menu, select **Processor Resources**, and choose the action you want to perform**.**

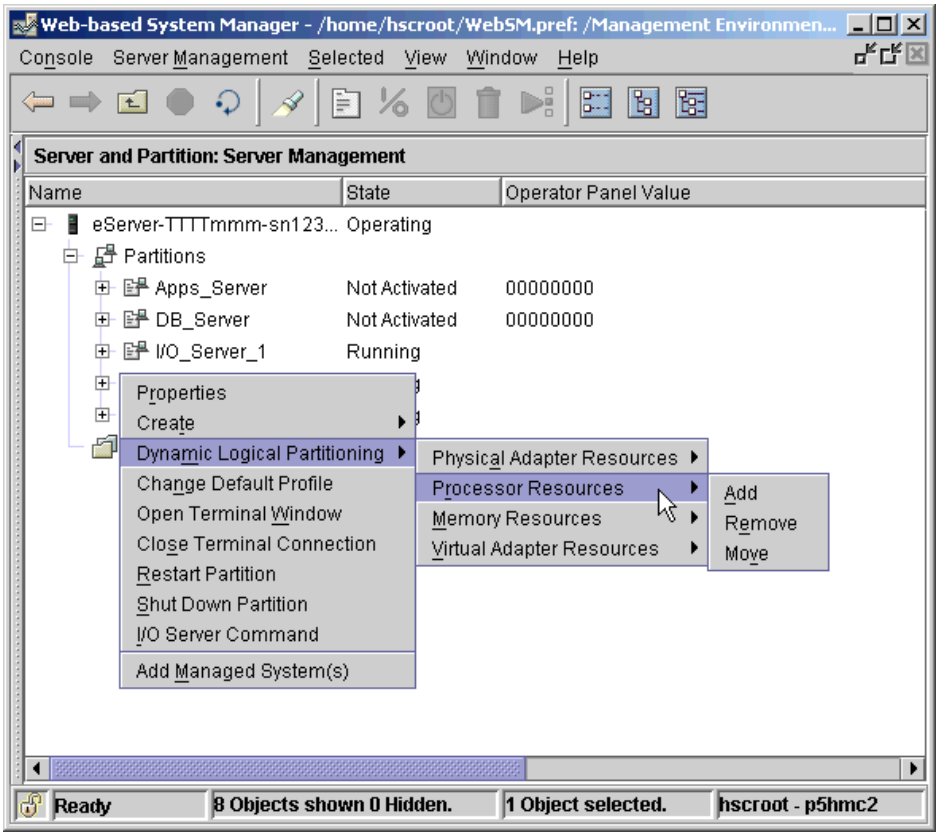

Figure 3-9 HMC panel Dynamic Logical Partitioning

When you select **Processor Resources** → **Add**, the panel shown in Figure 3-10 on page 66 is displayed. Here you can specify the processing units and the number of virtual processors you want to add to the selected partition. The limits for adding processing units and virtual processors are the maximum values defined in the partition profile. This panel also allows you to add variable weight when the partition runs in uncapped mode.

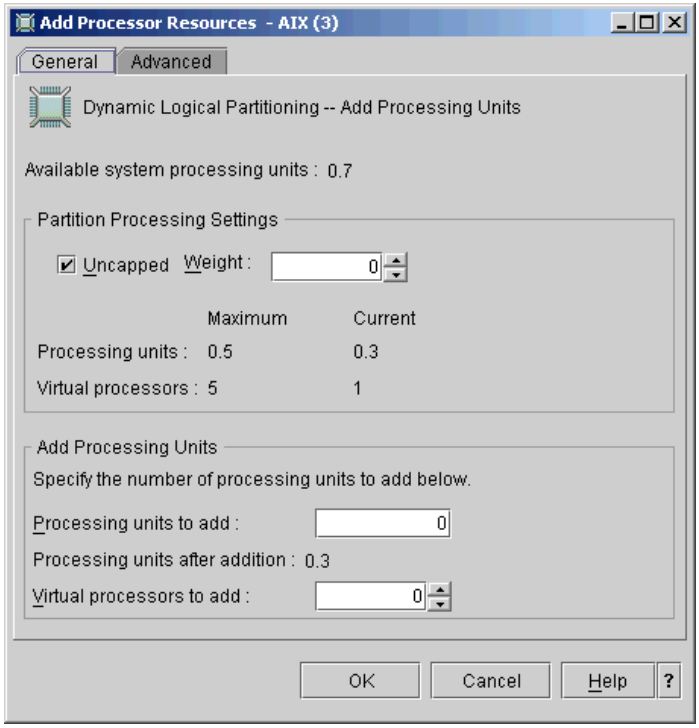

Figure 3-10 Add Processor Resource panel on the HMC

Additionally, the Add Processor Resource panel allows you to dynamically change the partition mode from uncapped to capped or vice versa. To show the actual status of the partition, use the **`lparstat -i`** command from the AIX command line interface of the partition, as shown in the following:

```
# lparstat -i
Node Name                                  : applsrv
Partition Name                             : Apps_Server
Partition Number                           : 4
Type                                       : Shared-SMT
Mode                                       : Uncapped
Entitled Capacity                          : 0.30
Partition Group-ID                         : 32772
Shared Pool ID                             : 0
Online Virtual CPUs                        : 2
Maximum Virtual CPUs                       : 10
Minimum Virtual CPUs                       : 1
Online Memory                              : 512 MB
Maximum Memory                             : 1024 MB
Minimum Memory                             : 128 MB
Variable Capacity Weight                   : 128
```

```
Minimum Capacity                        : 0.20
Maximum Capacity                        : 1.00
Capacity Increment                      : 0.01
Maximum Dispatch Latency                : 16999999
Maximum Physical CPUs in system         : 2
Active Physical CPUs in system          : 2
Active CPUs in Pool                     : -
Unallocated Capacity                    : 0.00
Physical CPU Percentage                 : 15.00%
Unallocated Weight                      : 0
```

Figure 3-11 shows the way to change the mode of the partition from uncapped to capped mode. Un-check the Uncapped checkbox and click **OK**.

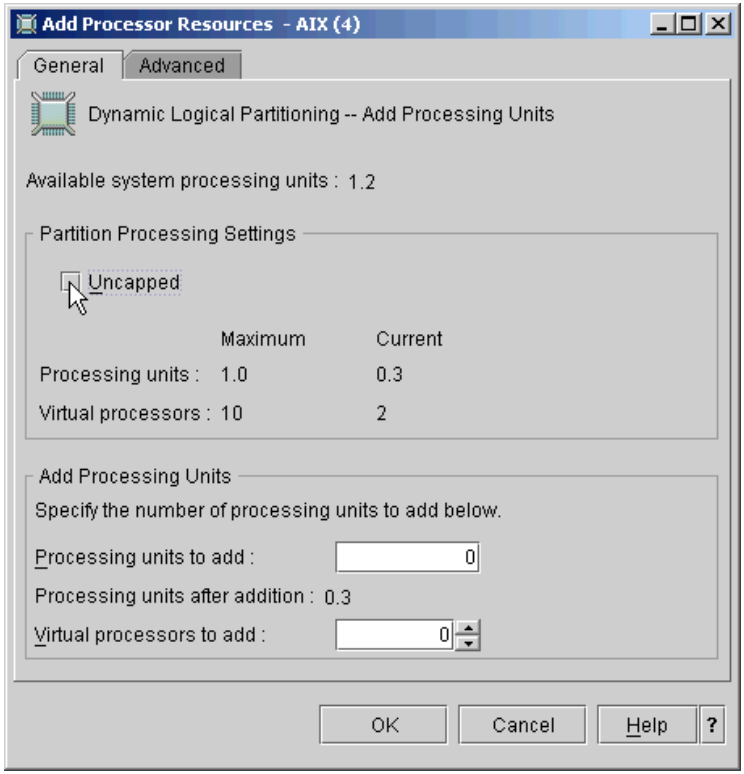

Figure 3-11 Changing the partition mode from Uncapped to Capped

To verify this dynamic action, use the **lparstat -i** command on the selected partition again. The partition mode changed from uncapped to capped.

```
# lparstat -i
Node Name                                  : applsrv
Partition Name                             : Apps_Server
Partition Number                           : 4
Type                                       : Shared-SMT
Mode                                       : Capped
Entitled Capacity                          : 0.30
Partition Group-ID                         : 32772
Shared Pool ID                             : 0
Online Virtual CPUs                        : 2
Maximum Virtual CPUs                       : 10
Minimum Virtual CPUs                       : 1
Online Memory                              : 512 MB
Maximum Memory                             : 1024 MB
Minimum Memory                             : 128 MB
Variable Capacity Weight                   : 128
Minimum Capacity                           : 0.20
Maximum Capacity                           : 1.00
Capacity Increment                         : 0.01
Maximum Dispatch Latency                   : 16999999
Maximum Physical CPUs in system            : 2
Active Physical CPUs in system             : 2
Active CPUs in Pool                        : -
Unallocated Capacity                       : 0.00
Physical CPU Percentage                    : 15.00%
Unallocated Weight                         : 0
```

Figure 3-12 shows the Remove Processor Resources panel that allows you to dynamically remove processing units and virtual processors. The limits for the removal of processing units and virtual processors are the minimum values defined in the partition profile.

This panel also allows you to remove variable weight when the partition runs in uncapped mode.

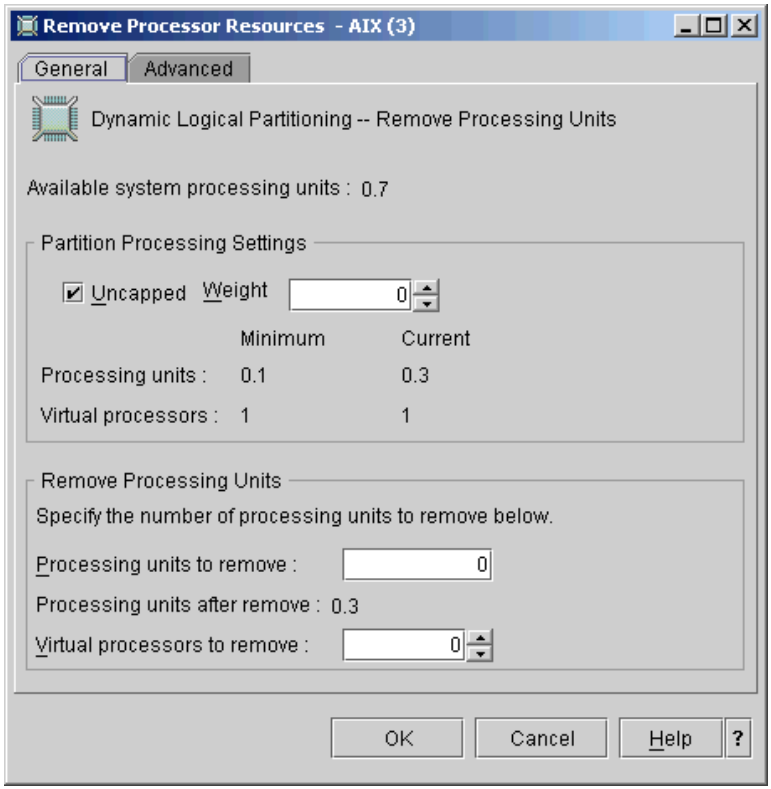

Figure 3-12 Remove Processor Resource panel on the HMC

It is also possible to change the partition mode from capped to uncapped and vice versa from the Remove Processor Resource panel.

When moving processing units you have to select the partition you want the processing units removed from and choose the Move Processor Resource Panel shown in Figure 3-13.

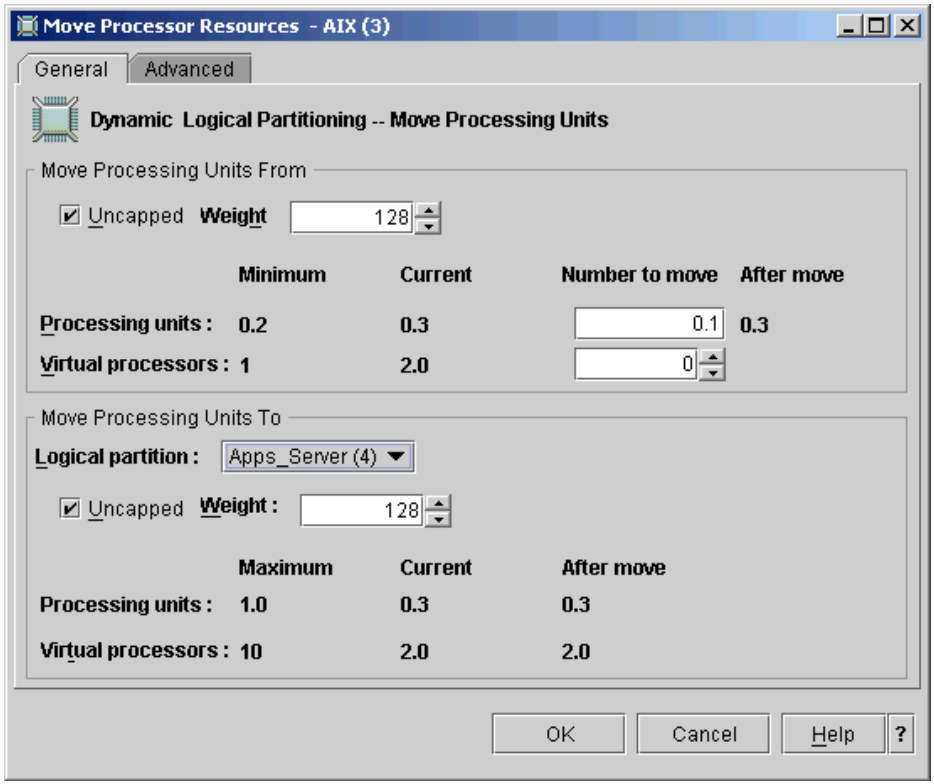

Figure 3-13 Move Processor Resources panel on HMC

In the Processing units field, select the amount of processor capacity you want to remove from the selected partition and move to the partition you select from the drop-down menu under Logical Partition. In this example, we want to move 0.1 processing units from the Web_Server partition to the Apps_Server partition.

You can also choose to move virtual processors to adjust the number of virtual processors of your partition. This does not actually move the virtual processor but removes and adds the defined number of virtual processors to the chosen partitions.

3.3.6 Limitations and considerations

The following limitations must be considered when implementing shared processor partitions:

- The limitation for a shared processor partition is 0.1 processing units of a physical processor. So the number of shared processor partitions you can create for a system depends mostly on the number of processors in a system.
- The maximum number of partitions planned is 254.
- The maximum number of virtual processors in a partition is 64.
- A mix of dedicated and shared processors within the same partition is not supported.
- If you dynamically remove a virtual processor you cannot specify a particular virtual CPU to be removed. The operating system will choose the virtual CPU to be removed.
- Shared processors may render AIX affinity management useless. AIX will continue to utilize affinity domain information as provided by firmware to build associations of virtual processors to memory, and will continue to show preference to redispatching a thread to the virtual CPU that it last ran on.

There is overhead associated with the maintenance of online virtual processors, so you should carefully consider their capacity requirements before choosing values for these attributes.

Virtual processors have dispatch latency since they are scheduled. When a virtual processor is made runnable, it is placed on a run queue by the POWER Hypervisor, where it waits until it is dispatched. The time between these two events is referred to as dispatch latency.

The dispatch latency of a virtual processor depends on the partition entitlement and the number of virtual processors that are online in the partition. The capacity entitlement is equally divided among these online virtual processors, so the number of online virtual processors impacts the length of each virtual processor's dispatch. The smaller the dispatch cycle, the greater the dispatch latency.

At the time of writing, the worst case virtual processor dispatch latency is 18 milliseconds since the minimum dispatch cycle that is supported at the virtual processor level is one millisecond. This latency is based on the minimum partition entitlement of 1/10 of a physical processor and the 10 millisecond rotation period of the Hypervisor's dispatch wheel. It can be easily visualized by imagining that a virtual processor is scheduled in the first and last portions of two 10 millisecond intervals. In general, if these latencies are too great, then clients may increase entitlement, minimize the number of online virtual processors without reducing entitlement, or use dedicated processor partitions.

In general, the value of the minimum, desired, and maximum virtual processor attributes should parallel those of the minimum, desired, and maximum capacity attributes in some fashion. A special allowance should be made for uncapped partitions, since they are allowed to consume more than their entitlement.

If the partition is uncapped, then the administrator may want to define the desired and maximum virtual processor attributes x% above the corresponding entitlement attributes. The exact percentage is installation-specific, but 25 to 50 percent is a reasonable number.

Table 3-2 shows several reasonable settings of number of virtual processor, processing units, and the capped and uncapped mode.

Table 3-2 Reasonable settings for shared processor partitions

Min VPs[a]	Desired VPs	Max VPs	Min PU[b]	Desired PU	Max. PU	Capped
1	2	4	0.1	2.0	4.0	Y
1	3 or 4	6 or 8	0.1	2.0	4.0	N
2	2	6	2.0	2.0	6.0	Y
2	3 or 4	8 or 10	2.0	2.0	6.0	N

a - Virtual processors
b - Processing units

Operating systems and applications running in shared partitions need not be aware that they are sharing processors. However, overall system performance can be significantly improved by minor operating system changes. AIX 5L Version 5.3 provides support for optimizing overall system performance of shared processor partitions.

In a shared partition there is not a fixed relationship between the virtual processor and the physical processor. The POWER Hypervisor will try to use a physical processor with the same memory affinity as the virtual processor, but it is not guaranteed. Virtual processors have the concept of a home physical processor. If it can't find a physical processor with the same memory affinity, then it gradually broadens its search to include processors with weaker memory affinity, until it finds one that it can use. As a consequence, memory affinity is expected to be weaker in shared processor partitions.

Workload variability is also expected to be increased in shared partitions because there are latencies associated with the scheduling of virtual processors and interrupts. SMT may also increase variability, since it adds another level of

resource sharing, which could lead to a situation where one thread interferes with the forward progress of its sibling.

Therefore, if an application is cache-sensitive or cannot tolerate variability, then it should be deployed in a dedicated partition with SMT disabled. In dedicated partitions, the entire processor is assigned to a partition. Processors are not shared with other partitions, and they are not scheduled by the POWER Hypervisor. Dedicated partitions must be explicitly created by the system administrator using the Hardware Management Console.

Processor and memory affinity data is only provided in dedicated partitions. In a shared processor partition, all processors are considered to have the same affinity. Affinity information is provided through RSET APIs, which contain discovery and bind services.

3.4 Virtual Ethernet introduction

Virtual Ethernet enables inter-partition communication without the need for physical network adapters assigned to each partition. Virtual Ethernet allows the administrator to define in-memory point-to-point connections between partitions. These connections exhibit characteristics similar to physical high-bandwidth Ethernet connections and support multiple protocols (IPv4, IPv6, ICMP). Virtual Ethernet requires a p5 system with either AIX 5L Version 5.3 or the appropriate level of Linux and an HMC to define the Virtual Ethernet devices. Virtual Ethernet does not require the purchase of any additional features or software such as the Advanced POWER Virtualization Feature.

The concepts of implementing Virtual Ethernet on p5 systems are categorized in the following sections:

- Virtual LAN
- Virtual Ethernet connections
- Benefits of Virtual Ethernet
- Limitations

3.4.1 Virtual LAN

This section discusses the concepts of Virtual LAN (VLAN) technology with specific reference to its implementation within AIX.

Virtual LAN overview

Virtual LAN is a technology used for establishing virtual network segments on top of physical switch devices. If configured appropriately, a VLAN definition can

straddle multiple switches. Typically, a VLAN is a broadcast domain that enables all nodes in the VLAN to communicate with each other without any L3 routing or inter-VLAN bridging. In Figure 3-14, two VLANs (VLAN 1 and 2) are defined on three switches (Switch A, B, and C). Although nodes C-1 and C-2 are physically connected to the same switch C, traffic between two nodes can be blocked. To enable communication between VLAN 1 and 2, L3 routing or inter-VLAN bridging should be established between them; this is typically provided by an L3 device.

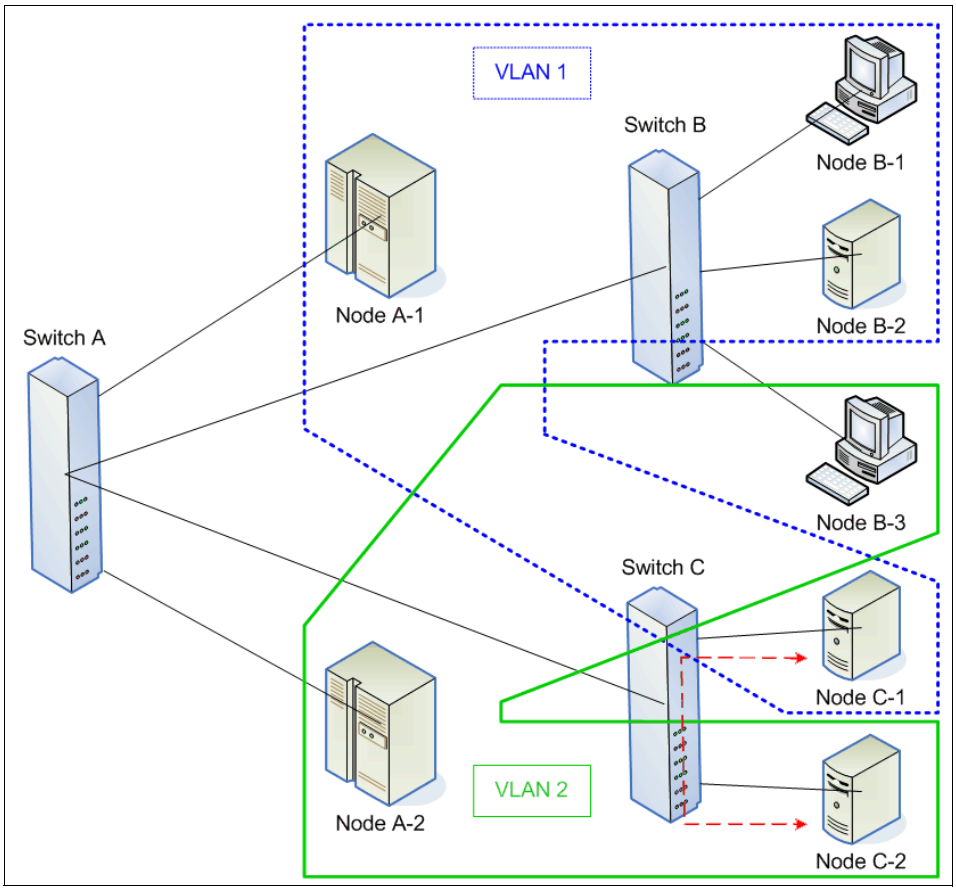

Figure 3-14 Example of a VLAN

The use of VLAN provides increased LAN security and flexible network deployment over traditional network devices.

AIX virtual LAN support

Some of the technologies for implementing VLANs include:

- Port-based VLAN
- Layer 2 VLAN
- Policy-based VLAN
- IEEE 802.1Q VLAN

VLAN support in AIX is based on the IEEE 802.1Q VLAN implementation. The IEEE 802.1Q VLAN is achieved by adding a VLAN ID tag to an Ethernet frame, and the Ethernet switches restricting the frames to ports that are authorized to receive frames with that VLAN ID. Switches also restrict broadcasts to the logical network by ensuring that a broadcast packet is delivered to all ports which are configured to receive frames with the VLAN ID that the broadcast frame was tagged with.

A port on a VLAN-capable switch has a default PVID (Port virtual LAN ID) that indicates the default VLAN the port belongs to. The switch adds the PVID tag to untagged packets that are received by that port. In addition to a PVID, a port may belong to additional VLANs and have those VLAN IDs assigned to it that indicate the additional VLANs the port belongs to.

A port will only accept untagged packets or packets with a VLAN ID (PVID or additional VIDs) tag of the VLANs the port belongs to. A port configured in the untagged mode is only allowed to have a PVID and will receive untagged packets or packets tagged with the PVID. The untagged port feature helps systems that do not understand VLAN tagging communicate with other systems using standard Ethernet.

Each VLAN ID is associated with a separate Ethernet interface to the upper layers (for example IP) and creates unique logical Ethernet adapter instances per VLAN (for example ent1 or ent2).

You can configure multiple VLAN logical devices on a single system. Each VLAN logical device constitutes an additional Ethernet adapter instance. These logical devices can be used to configure the same Ethernet IP interfaces as are used with physical Ethernet adapters.

VLAN communication by example

This section discusses how VLAN communication between partitions and with external networks works in more detail, using the sample configuration in Figure 3-15 on page 76. The configuration is using four client partitions (Partition 1 through Partition 4) and one Virtual I/O Server. Each of the client partitions is defined with one Virtual Ethernet adapter. The Virtual I/O Server has a Shared

Ethernet Adapter which bridges traffic to the external network. The Shared Ethernet Adapter is discussed in more detail in 3.5, “Shared Ethernet Adapter” on page 80.

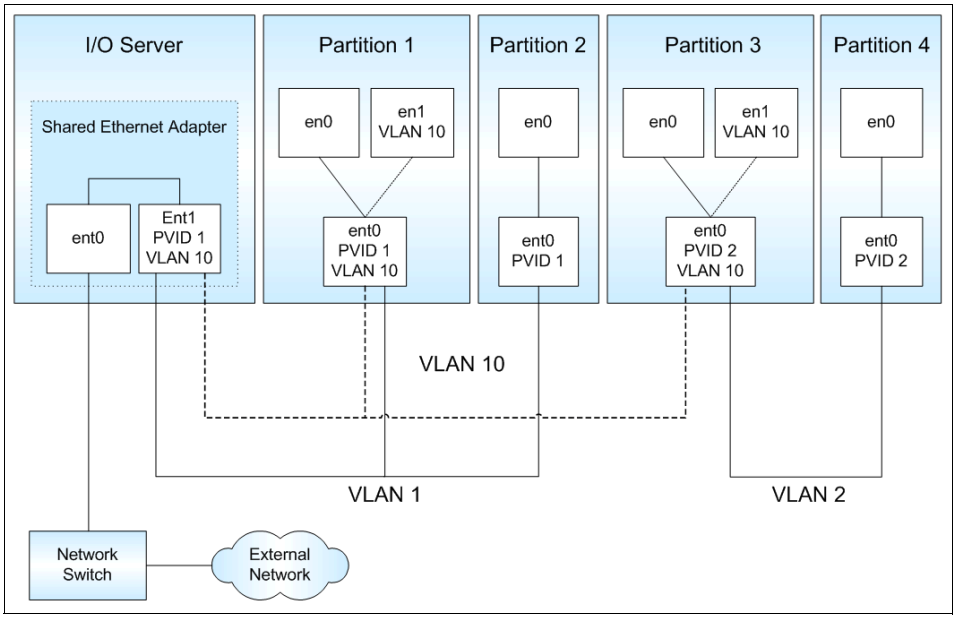

Figure 3-15 VLAN configuration

Interpartition communication

Partition 2 and Partition 4 are using the PVID (Port virtual LAN ID) only. This means that:

- Only packets for the VLAN specified as PVID are received.
- Packets sent have a VLAN tag added for the VLAN specified as PVID by the Virtual Ethernet adapter.

In addition to the PVID, the Virtual Ethernet adapters in Partition 1 and Partition 3 are also configured for VLAN 10 using specific network interface (en1) create through **`smitty vlan`**. This means that:

- Packets sent through network interfaces en1 are added a tag for VLAN 10 by the network interface in AIX.
- Only packets for VLAN 10 are received by the network interfaces en1.
- Packets sent through en0 are automatically tagged for the VLAN specified as PVID.
- Only packets for the VLAN specified as PVID are received by the network interfaces en0.

Table 3-3 lists which *client* partition can communicate which each other through what network interfaces.

Table 3-3 Interpartition VLAN communication

VLAN	Partition / Network interface
1	Partition 1 / en0 Partition 2 / en0
2	Partition 3 / en0 Partition 4 / en0
10	Partition 1 / en1 Partition 3 / en1

Communication with external networks

The Shared Ethernet Adapter is configured with PVID 1 and VLAN 10. This means that untagged packets that are received by the Shared Ethernet Adapter are tagged for VLAN 1. Handling of outgoing traffic depends on the VLAN tag of the outgoing packets.

- Packets tagged with the VLAN which matches the PVID of the Shared Ethernet Adapter are untagged before being sent out to the external network.
- Packets tagged with a VLAN *other than* the PVID of the Shared Ethernet Adapter are sent out with the VLAN tag unmodified.

In our example, Partition 1 and Partition 2 have access to the external network through network interface en0 using VLAN 1. Since these packets are using the PVID, the Shared Ethernet Adapter will remove the VLAN tags before sending the packets to the external network.

Partition 1 and Partition 3 have access to the external network using network interface en1 and VLAN 10. These packets are sent out by the Shared Ethernet Adapter with the VLAN tag. Therefore, only VLAN-capable destination devices will be able to receive the packets. Table 3-4 lists this relationship.

Table 3-4 VLAN communication to external network

VLAN	Partition / Network interface
1	Partition 1 / en0 Partition 2 / en0
10	Partition 1 / en1 Partition 3 / en1

3.4.2 Virtual Ethernet connections

Virtual Ethernet connections supported in POWER5 systems use VLAN technology to ensure that the partitions can only access data directed to them. The POWER Hypervisor provides a Virtual Ethernet switch function based on the IEEE 802.1Q VLAN standard that allows partition communication within the same server. The connections are based on an implementation internal to the hypervisor that moves data between partitions. This section describes the various elements of a Virtual Ethernet and implications relevant to different types of workloads. Figure 3-16 is an example of an inter-partition VLAN.

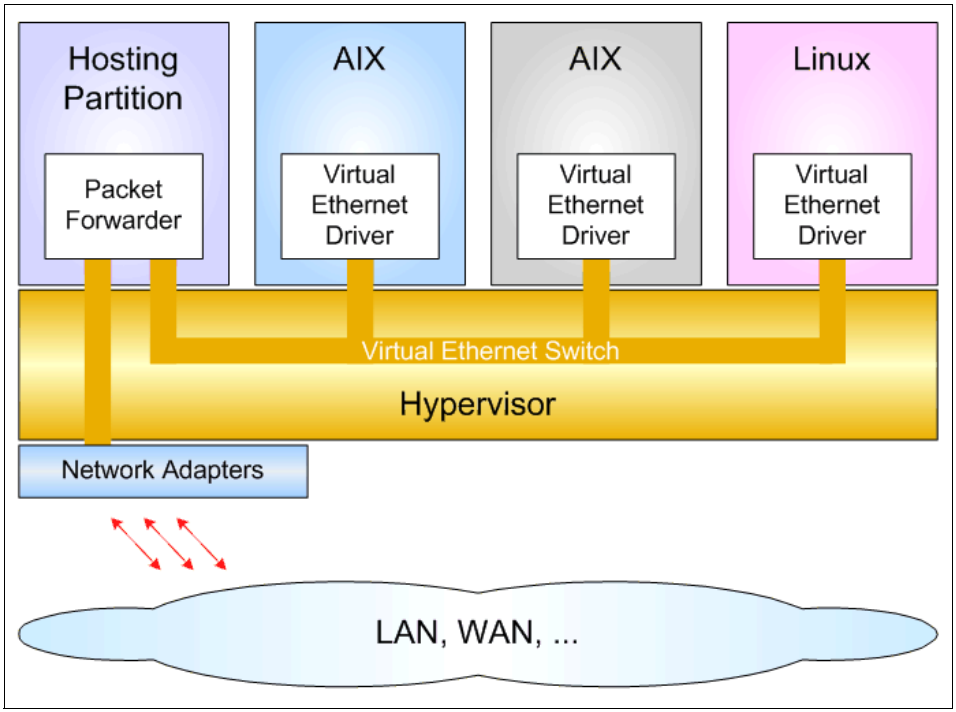

Figure 3-16 logical view of an inter-partition VLAN

Virtual Ethernet adapter concepts

Partitions that communicate through a Virtual Ethernet channel will need to have an additional in-memory channel. This requires the creation of an in-memory channel between partitions on the HMC. The kernel creates a virtual device for each memory channel indicated by the firmware. The AIX configuration manager creates the device special files. A unique Media Access Control (MAC) address is also generated when the Virtual Ethernet device is created. A *prefix* value can be assigned for the system so that the generated MAC addresses in a system

consists of a common system prefix, plus an algorithmically-generated unique part per adapter.

The Virtual Ethernet can also be used as a bootable device to allow such tasks as operating system installations to be performed using NIM. Section 5.3.5, “Defining Virtual I/O resources for the clients” on page 166 discusses network boot of Virtual Ethernet in more detail.

Performance considerations

The transmission speed of Virtual Ethernet adapters is in the range of 1-3 Gigabits per second, depending on the transmission (MTU) size. A partition can support up to 256 Virtual Ethernet adapters with each Virtual Ethernet capable of being associated with up to 21 VLANs (20 VID and 1 PVID).

The Virtual Ethernet connections generally take up more processor time than a local adapter to move a packet (DMA versus copy). For shared processor partitions, performance will be gated by the partition definitions (for example, entitled capacity and number of processors). Small partitions communicating with each other will experience more packet latency due to partition context switching. In general, high bandwidth applications should *not* be deployed in small shared processor partitions. For dedicated partitions, throughput should be comparable to a 1 Gigabit Ethernet for small packets providing much better performance than 1 Gigabit Ethernet for large packets. For large packets, the Virtual Ethernet communication is copy bandwidth limited.

For more detailed information relating to Virtual Ethernet performance considerations refer to *Advanced POWER Virtualization on IBM eServer p5 Servers Architecture and Performance Considerations*, SG24-5768.

3.4.3 Benefits of virtual Ethernet

Due to the number of partitions possible on many systems being greater than the number of I/O slots, Virtual Ethernet is a convenient and cost saving option to enable partitions within a single system to communicate with one another through a VLAN. The VLAN creates logical Ethernet connections between one or more partitions and is designed to help prevent a failed or malfunctioning operating system from being able to impact the communication between two functioning operating systems. The Virtual Ethernet connections may also be *bridged* to an external network to permit partitions without physical network adapters to communicate outside of the server.

3.4.4 Dynamic partitioning for Virtual Ethernet devices

Virtual Ethernet resources can be assigned and removed dynamically. On the HMC, Virtual Ethernet target and server adapters can be assigned and removed from a partition using dynamic logical partitioning. The mapping between physical and virtual resources on the Virtual I/O Server can also be done dynamically.

3.4.5 Limitations and considerations

The following are limitations that must be considered when implementing a Virtual Ethernet:

- A maximum of up to 256 Virtual Ethernet adapters are permitted per partition.
- Virtual Ethernet can be used in both shared and dedicated processor partitions provided the partition is running AIX 5L Version 5.3 or Linux with the 2.6 kernel or a kernel that supports virtualization.
- A mixture of Virtual Ethernet connections, real network adapters, or both are permitted within a partition.
- Virtual Ethernet can only connect partitions within a single system.
- Virtual Ethernet requires a POWER5 system and an HMC to define the Virtual Ethernet adapters.
- Virtual Ethernet connections from AIX or Linux partitions to an i5/OS partition may work, however at the time of writing, these capabilities were unsupported.
- Virtual Ethernet uses the system processors for all communication functions instead of offloading that load to processors on network adapter cards. As a result, there is an increase in system processor load generated by the use of Virtual Ethernet.

3.5 Shared Ethernet Adapter

A Shared Ethernet Adapter can be used to connect a physical Ethernet to the Virtual Ethernet. It also provides the possibility for several client partitions to share one physical adapter.

The following sections discuss the various aspects of Shared Ethernet Adapters such as:

- Connecting Virtual Ethernet to external networks
- Ethernet adapter sharing

- Benefits of Shared Ethernet Adapters
- Using Link Aggregation (EtherChannel) for external network interface
- Limitations and considerations

3.5.1 Connecting a virtual Ethernet to external networks

There are two ways you can connect the Virtual Ethernet that enables the communication between logical partitions on the same server to an external network.

Routing

By enabling the AIX routing capabilities (ipforwarding network option) one partition with a physical Ethernet adapter connected to an external network can act as router. Figure 3-17 shows a sample configuration. In this type of configuration the partition that routes the traffic to the external work does not necessarily have to be the Virtual I/O Server as in the pictured example. It could be any partition with a connection to the outside world. The client partitions would have their default route set to the partition which routes traffic to the external network.

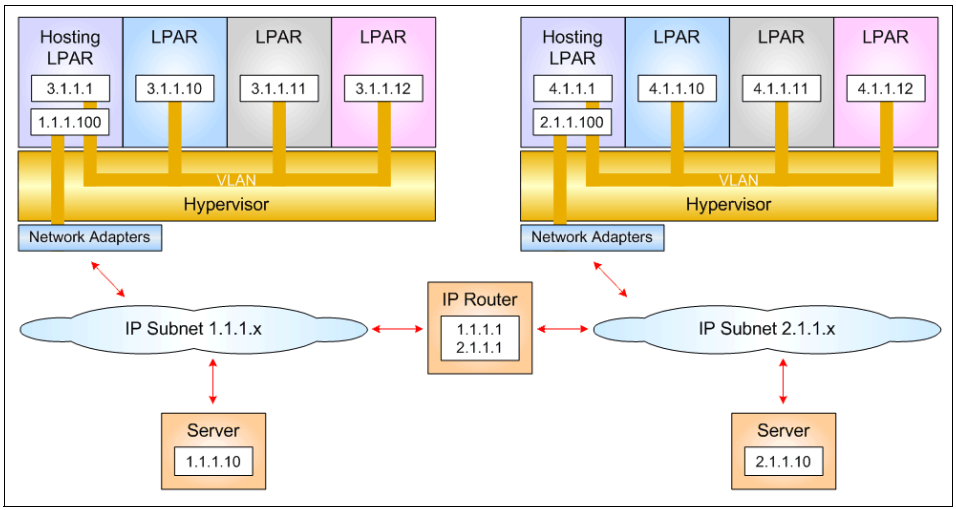

Figure 3-17 Connection to external network using AIX routing

Shared Ethernet Adapter

Using a Shared Ethernet Adapter (SEA) you can connect internal and external VLANs using one physical adapter. The Shared Ethernet Adapter hosted in the Virtual I/O Server acts as a layer 2 switch between the internal and external network.

Shared Ethernet Adapter is a new service that acts as a layer 2 network bridge to securely transport network traffic from a virtual Ethernet to a real network adapter. The Shared Ethernet Adapter service runs in the Virtual I/O Server. It cannot be run in a general purpose AIX partition.

Shared Ethernet Adapter requires the POWER Hypervisor component of POWER5 systems and therefore cannot be used on POWER4 systems. It also cannot be used with AIX 5L Version 5.2 because the device drivers for Virtual Ethernet are only available for AIX 5L Version 5.3 and Linux. Thus there is no way to connect an AIX 5L Version 5.2 system to a Shared Ethernet Adapter.

The Shared Ethernet Adapter allows partitions to communicate outside the system without having to dedicate a physical I/O slot and a physical network adapter to a client partition. The Shared Ethernet Adapter has the following characteristics:

- Virtual Ethernet MAC addresses are visible to outside systems
- Broadcast/multicast is supported
- ARP and NDP can work across a shared Ethernet

In order to bridge network traffic between the Virtual Ethernet and external networks the Virtual I/O Server has to be configured with at least one physical Ethernet adapter. One Shared Ethernet Adapter can be shared by multiple VLANs and multiple subnets can connect using a single adapter on the Virtual I/O Server. Figure 3-18 shows a configuration example. A Shared Ethernet Adapter can include up to 16 Virtual Ethernet adapters that share the physical access.

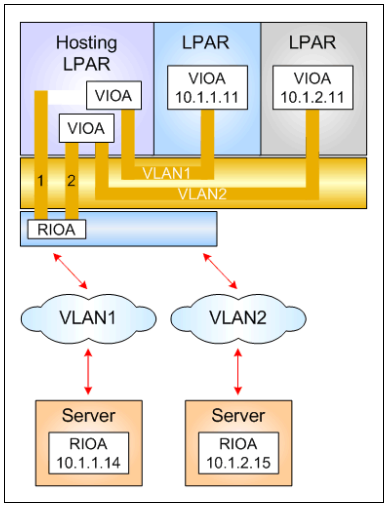

Figure 3-18 Shared Ethernet Adapter configuration

A Virtual Ethernet adapter connected to the Shared Ethernet Adapter must have the trunk flag set. Once an Ethernet frame is sent from the Virtual Ethernet adapter on a client partition to the POWER Hypervisor, the POWER Hypervisor searches for the destination MAC address within the VLAN. If no such MAC address exists within the VLAN, it forwards the frame to the trunk Virtual Ethernet adapter that is defined on the same VLAN. The trunk virtual Ethernet adapter enables a layer 2 bridge to a physical adapter.

The shared Ethernet directs packets based on the VLAN ID tags. It learns this information based on observing the packets originating from the virtual adapters. One of the virtual adapters in the Shared Ethernet adapter is designated as the default PVID adapter. Ethernet frames without any VLAN ID tags are directed to this adapter and assigned the default PVID.

When the shared Ethernet receives IP (or IPv6) packets that are larger than the MTU of the adapter that the packet is forwarded through, either IP fragmentation is performed and the fragments forwarded or an ICMP packet too big message is returned to the source when the packet cannot be fragmented.

Theoretically, one adapter can act as the only contact with external networks for all client partitions. Depending on the number of client partitions and the network load they produce performance can become a critical issue. Because the Shared Ethernet Adapter is dependant on Virtual I/O, it consumes processor time for all communications. A significant amount of CPU load can be generated by the use of Virtual Ethernet and Shared Ethernet Adapter.

There are several different ways to configure physical and virtual Ethernet adapters into Shared Ethernet Adapters to maximize throughput.

- Using Link Aggregation (EtherChannel), several physical network adapters can be aggregated. See 3.5.2, “Using Link Aggregation (EtherChannel) to external networks” on page 84 for more details.
- Using several Shared Ethernet Adapters provides more queues and more performance. An example for this configuration is shown in Figure 3-19 on page 84.

Other aspects which have to be taken into consideration are availability (see 4.3.1, “Providing higher availability for Virtual I/O Server” on page 121) and the possibility to connect to different networks.

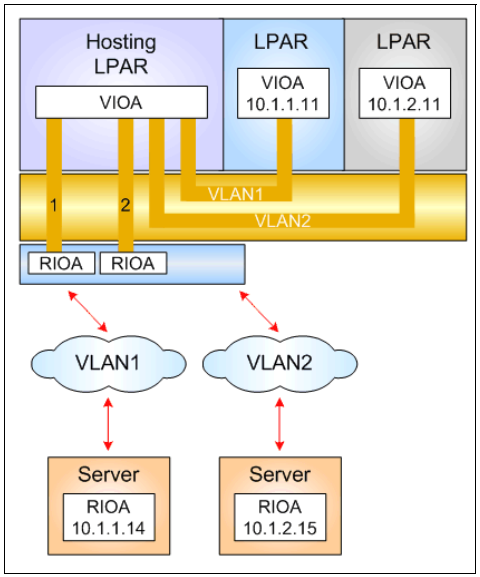

Figure 3-19 Multiple Shared Ethernet Adapter configuration

3.5.2 Using Link Aggregation (EtherChannel) to external networks

Link aggregation is network port aggregation technology that allows several Ethernet adapters to be aggregated together to form a single pseudo Ethernet device. This technology can be used to overcome the bandwidth limitation of a single network adapter and avoid bottlenecks when sharing one network adapter among many client partitions.

For example, ent0 and ent1 can be aggregated to ent3. Interface en3 would then be configured with an IP address. The system considers these aggregated adapters as one adapter. Therefore, IP is configured as on any other Ethernet adapter. In addition, all adapters in the link aggregation are given the same hardware (MAC) address, so they are treated by remote systems as if they were one adapter. The main benefit of link aggregation is that they have the network bandwidth of all of their adapters in a single network presence. If an adapter fails, the packets are automatically sent on the next available adapter without disruption to existing user connections. The adapter is automatically returned to service on the link aggregation when it recovers.

You can use EtherChannel (EC) or IEEE 802.3ad Link Aggregation (LA) to aggregate network adapters. While EC is an AIX-specific implementation of adapter aggregation, LA follows the IEEE 802.3ad standard. Table 3-5 on page 85 shows the main differences between EC and LA.

Table 3-5 Main differences between EC and LA aggregation

EtherChannel	IEEE 802.3ad Link Aggregation
Requires switch configuration	Little, if any, configuration of switch required to form aggregation. Some initial setup of the switch may be required.
Supports different packet distribution modes	Supports only standard distribution mode

The main benefit of using LA is, that if the switch supports the *Link Aggregation Control Protocol* (LACP) no special configuration of the switch ports is required. The benefit of EC is the support of different packet distribution modes. This means it is possible to influence the load balancing of the aggregated adapters. In the remainder of this document, we use Link Aggregation where possible since that is considered a more universally understood term.

Note: Only outgoing packets are subject to the following discussion; incoming packets are distributed by the Ethernet switch.

Standard distribution mode selects the adapter for the outgoing packets by algorithm. The adapter selection algorithm uses the last byte of the destination IP address (for TCP/IP traffic) or MAC address (for ARP and other non-IP traffic). Therefore all packets to a specific IP-address will always go through the same adapter. There are other adapter selection algorithms based on source, destination, or a combination of source and destination ports available. EC provides one further distribution mode called *round robin*. This mode will rotate through the adapters, giving each adapter one packet before repeating. The packets may be sent out in a slightly different order than they were given to the EC. It will make the best use of its bandwidth, but consider that it also introduces the potential for out-of-order packets at the receiving system. This risk is particularly high when there are few, long-lived, streaming TCP connections. When there are many such connections between a host pair, packets from different connections could be intermingled, thereby decreasing the chance of packets for the same connection arriving out-of-order.

To avoid the loss of network connectivity by switch failure, EC and LA can provide a backup adapter. The backup adapter should be connected to a different switch than the adapter of the aggregation. Now in case of switch failure the traffic can be moved with no disruption of user connections to the backup adapter.

Figure 3-20 on page 86 shows the aggregation of three plus one adapters to a single pseudo Ethernet device including a backup feature.

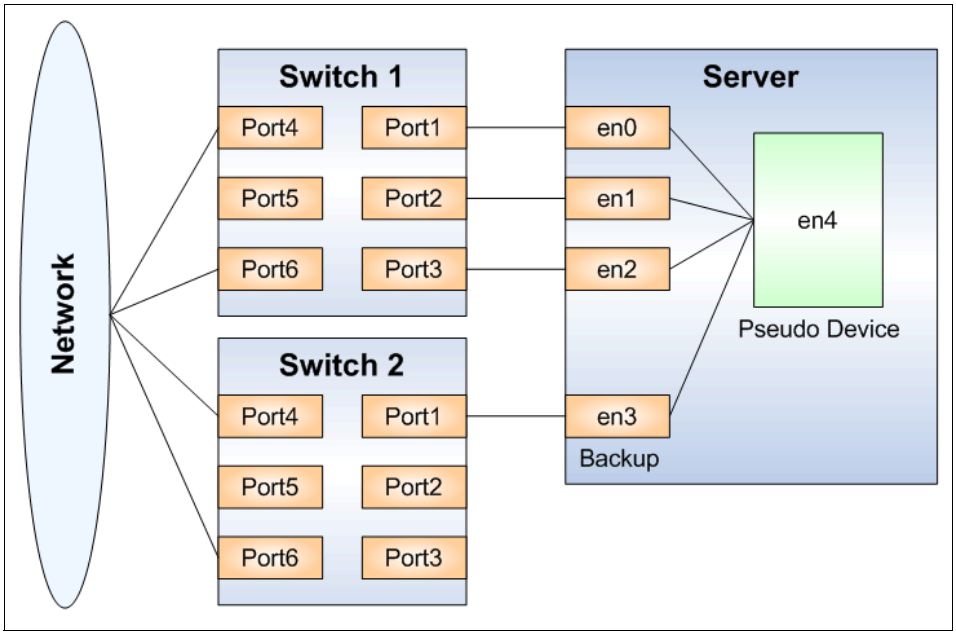

Figure 3-20 Link Aggregation (EtherChannel) pseudo device

3.5.3 Limitations and considerations

You must consider the following limitations when implementing Shared Ethernet Adapters in the Virtual I/O Server:

- Because Shared Ethernet Adapter depends on virtual Ethernet, which uses the system processors for all communications functions, a significant amount of system processor load can be generated by the use of virtual Ethernet and Shared Ethernet Adapter.
- One of the virtual adapters in the Shared Ethernet Adapter on the Virtual I/O Server must be defined as default adapter with a default PVID. This virtual adapter is designated as the PVID adapter and Ethernet frames without any VLAN ID tags are assigned the default PVID and directed to this adapter.
- Up to 16 virtual Ethernet adapters with 21 VLANs (20 VID and 1 PVID) on each can be shared on a single physical network adapter. There is no limit on the number of partitions that can attach to a VLAN, so the theoretical limit is very high. In practice, the amount of network traffic will limit the number of clients that can be served through a single adapter.

For performance and latency relevant information refer to *Advanced POWER Virtualization on IBM eServer p5 Servers Architecture and Performance Considerations*, SG24-5768.

3.6 Virtual SCSI introduction

Available at the time of writing is the first support of Virtual I/O, which pertains to a virtualized implementation of the SCSI protocol.

Virtual SCSI requires POWER5 hardware with the Advanced POWER Virtualization feature activated. It provides Virtual SCSI support for AIX 5L Version 5.3 and Linux (see 1.5, “Operating system support details” on page 9).

The driving forces behind virtual I/O are:

- The advanced technological capabilities of today’s hardware and operating systems like POWER5 and IBM AIX 5L Version 5.3.
- The value proposition enabling on demand computing and server consolidation. Virtual I/O also provides a more economic I/O model by using physical resources more efficiently through sharing.

At the time of writing, the virtualization features of the POWER5 platform support up to 254 partitions, while the server hardware only provides up to 160 I/O slots per machine. With each partition typically requiring one I/O slot for disk attachment and another one for network attachment, this puts a constraint on the number of partitions. To overcome these physical limitations, I/O resources have to be shared. Virtual SCSI provides the means to do this for SCSI storage devices.

Furthermore, virtual I/O allows attachment of previously unsupported storage solutions. As long as the Virtual I/O Server supports the attachment of a storage resource, any client partition can access this storage by using Virtual SCSI adapters.

For example, if there is no native support for EMC storage devices on Linux, running Linux in a logical partition of a POWER5 server makes this possible.

A Linux client partition can access the EMC storage through a Virtual SCSI adapter. Requests from the virtual adapters are mapped to the physical resources in the Virtual I/O Server. Driver support for the physical resources is therefore only needed in the Virtual I/O Server.

Note: You will see different terms in this publication that refer to the various components involved with virtual SCSI. Depending on the context, these terms may vary. With SCSI, usually the terms *initiator* and *target* are used, so you may see terms such as *virtual SCSI initiator* and *virtual SCSI target*. On the HMC, the terms *virtual SCSI server adapter* and *virtual SCSI client adapter* are used. Basically they refer to the same thing. When describing the client/server relationship between the partitions involved in virtual SCSI, the terms *hosting partition* (meaning the Virtual I/O Server) and *hosted partition* (meaning the client partition) are used.

The terms *Virtual I/O Server partition* and *Virtual I/O Server* both refer to the Virtual I/O Server as described in Chapter 4, "Virtual I/O Server configuration" on page 99. The terms are used interchangeably in this section.

3.6.1 Partition access to virtual SCSI devices

The following sections describe the virtual SCSI architecture and the protocols used. You find more details about the internals of virtual SCSI in 3.2, "Introduction to the POWER Hypervisor" on page 45 and *Advanced POWER Virtualization on IBM eServer p5 Servers Architecture and Performance Considerations*, SG24-5768.

Virtual SCSI client and server architecture overview

Virtual SCSI is based on a client/server relationship. The Virtual I/O Server owns the physical resources and acts as server or, in SCSI terms, target device. The logical partitions access the virtual SCSI resources provided by the Virtual I/O Server as clients.

The virtual I/O adapters are configured using an HMC. The provisioning of virtual disk resources is provided by the Virtual I/O Server.

Often the Virtual I/O Server is also referred to as hosting partition and the client partitions as hosted partitions.

Physical disks owned by the Virtual I/O Server can either be exported and assigned to a client partition whole, or can be partitioned into several logical volumes. The logical volumes can then be assigned to different partitions. Therefore, Virtual SCSI enables sharing of adapters as well as disk devices.

To make a physical or a logical volume available to a client partition it is assigned to a virtual SCSI server adapter in the Virtual I/O Server. Section 5.3.5, "Defining Virtual I/O resources for the clients" on page 166 covers in detail how this is done.

The client partition accesses its assigned disks through a virtual SCSI client adapter. The virtual SCSI client adapter sees standard SCSI devices and LUNs through this virtual adapter. The commands in the following example show how the disks appear on an AIX client partition.

```
# lsdev -Cc disk -s vscsi
hdisk2 Available  Virtual SCSI Disk Drive

# lscfg -vpl hdisk2
hdisk2 111.520.10DDEDC-V3-C5-T1-L810000000000 Virtual SCSI Disk Drive
```

Figure 3-21 shows an example where one physical disk is partitioned into two logical volumes inside the Virtual I/O Server. Each of the two client partitions is assigned one logical volume which it accesses through a virtual I/O adapter (vSCSI Client Adapter). Inside the partition the disk is seen as normal hdisk.

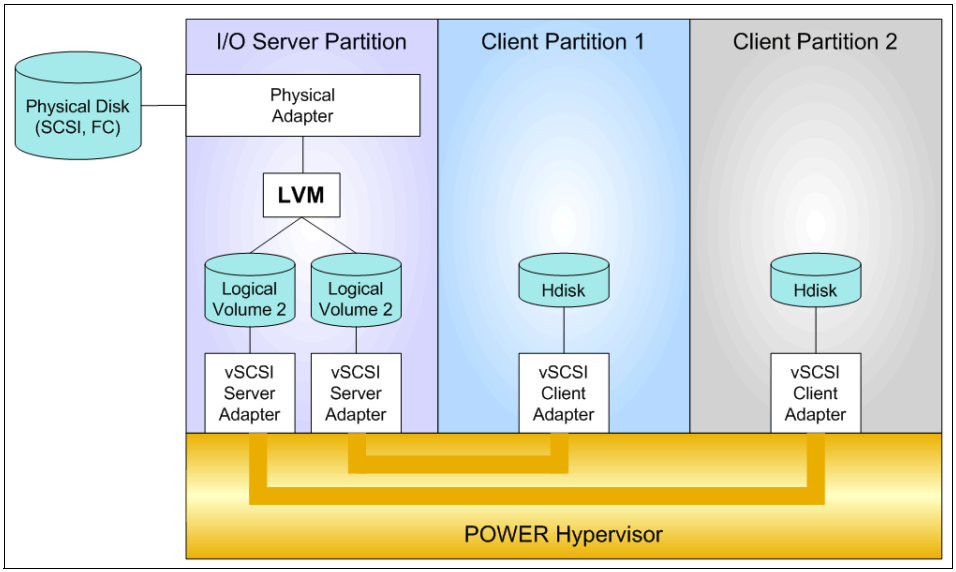

Figure 3-21 Virtual SCSI architecture overview

SCSI Remote Direct Memory Access

The SCSI family of standards provides many different transport protocols that define the rules for exchanging information between SCSI initiators and targets. Virtual SCSI uses the SCSI RDMA Protocol (SRP) which defines the rules for exchanging SCSI information in an environment where the SCSI initiators and targets have the ability to directly transfer information between their respective address spaces.

SCSI requests and responses are sent using the virtual SCSI adapters that communicate through the POWER Hypervisor.

The actual data transfer, however, is done directly between a data buffer in the client partition and the physical adapter in the Virtual I/O Server by using the Logical Remote Direct Memory Access (LRDMA) protocol.

Figure 3-22 shows how the data transfer using LRDMA appears.

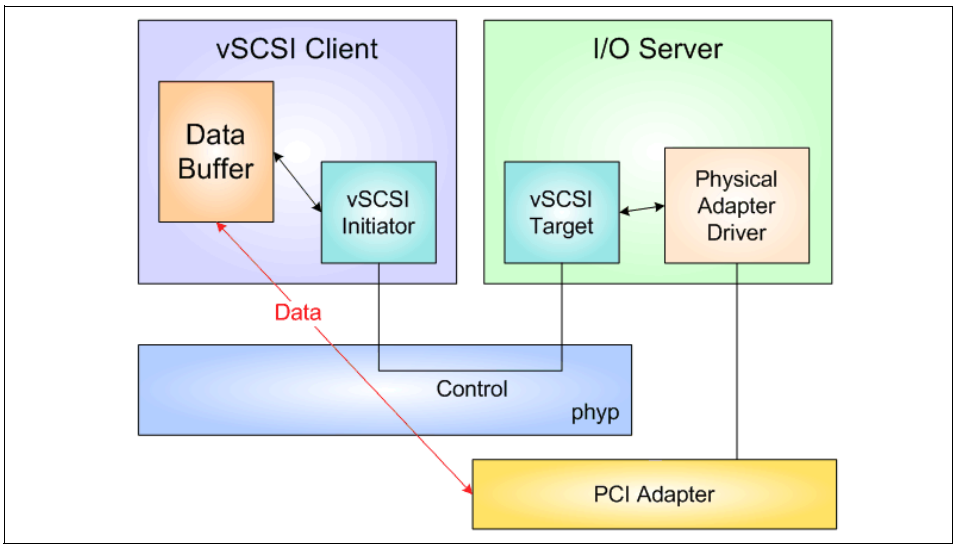

Figure 3-22 Logical Remote Direct Memory Access

AIX device configuration for virtual SCSI

The virtual I/O adapters are connected to a virtual host bridge which AIX treats much like a PCI host bridge. It is represented in the ODM as a bus device whose parent is sysplanar0. The virtual I/O adapters are represented as adapter devices with the virtual host bridge as their parent.

On the Virtual I/O Server, each logical volume or physical volume that is exported to a client partition is represented by a virtual target device that is a child of a Virtual SCSI server adapter.

On the client partition, the exported disks are visible as normal hdisks, but they are defined in subclass vscsi. They have a virtual SCSI client adapter as parent.

Figure 3-23 and Figure 3-24 on page 91 show the relationship of the devices used by AIX for Virtual SCSI and their physical counterparts.

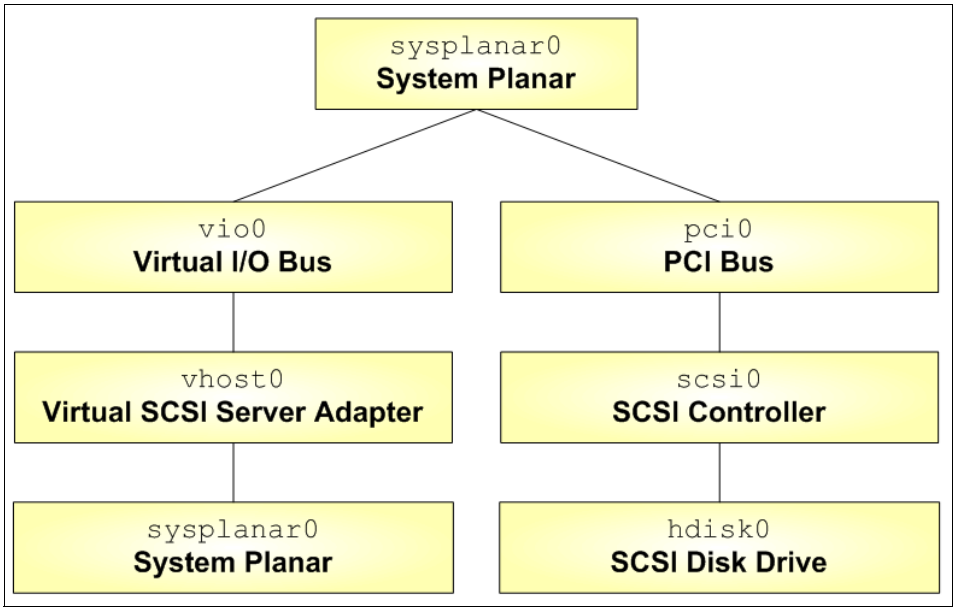

Figure 3-23 Virtual SCSI device relationship on Virtual I/O Server

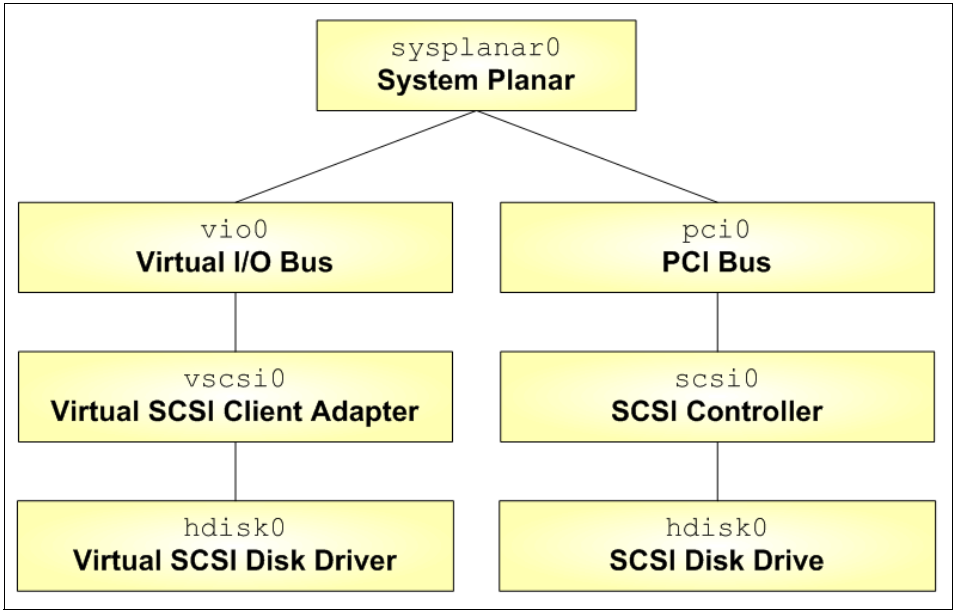

Figure 3-24 Virtual SCSI device relationship on AIX client partition

Dynamic partitioning for virtual SCSI devices

Virtual SCSI resources can be assigned and removed dynamically. On the HMC, Virtual SCSI target and server adapters can be assigned and removed from a partition using dynamic logical partitioning.

The mapping between physical and virtual resources on the Virtual I/O Server can also be done dynamically.

3.6.2 Limitations and considerations

The following areas should be considered when implementing Virtual SCSI:

- At the time of writing virtual SCSI supports Fibre Channel, parallel SCSI, and SCSI RAID devices. Other protocols such as SSA or tape and CD-ROM devices are not supported.
- Virtual SCSI itself does not have any limitations in terms of number of supported devices or adapters. However, the Virtual I/O Server supports a maximum of 65535 virtual I/O slots. A maximum of 256 virtual I/O slots can be assigned to a single partition.

  Every I/O slot needs some resources to be instantiated. Therefore, the size of the Virtual I/O Server puts a limit to the number of virtual adapters that can be configured. For details see *Advanced POWER Virtualization on IBM eServer p5 Servers Architecture and Performance Considerations*, SG24-5768.
- The SCSI protocol defines mandatory and optional commands. While virtual SCSI supports all the mandatory commands, not all optional commands are supported. You can find a complete list of the supported SCSI commands in Appendix B, “Supported SCSI commands” on page 245.
- There are performance implications when using virtual SCSI devices. It is important to understand that, due to the overhead associated with POWER Hypervisor calls, virtual SCSI will use additional CPU cycles when processing I/O requests. When putting heavy I/O load on virtual SCSI devices, this means you will use considerably more CPU cycles. Provided that there is sufficient CPU processing capacity available the performance of virtual SCSI should be comparable to dedicated I/O devices.

  Suitable applications for virtual SCSI include boot disks for the operating system or Web servers which will typically cache a lot of data. When designing a virtual I/O configuration, performance is an important aspect which should be given careful consideration. For a more in-depth discussion of performance issues see *Advanced POWER Virtualization on IBM eServer p5 Servers Architecture and Performance Considerations*, SG24-5768.

3.7 Partition Load Manager introduction

The Partition Load Manager software is part of the Advanced POWER Virtualization feature and helps customers maximize the utilization of processor and memory resources of DLPAR-capable logical partitions running AIX 5L on pSeries servers.

The Partition Load Manager is a resource manager that assigns and moves resources based on defined policies and utilization of the resources. PLM manages memory, both dedicated processors and partitions using Micro-Partitioning technology to readjust the resources. This adds additional flexibility on top of the micro-partitions flexibility added by the POWER Hypervisor.

PLM, however, has no knowledge about the importance of a workload running in the partitions and cannot readjust priority based on the changes of types of workloads. PLM does not manage Linux and i5/OS partitions. Figure 3-25 shows a comparison of features between PLM and the POWER Hypervisor.

PLM Differentiation	Capability	PLM	P5 PHYP
HW Support	**POWER4** PLM automates DLPAR adjustment for P4 install base	X	
	POWER5	X	X
OS Support	**AIX 5.2** PLM runs on AIX 5.2 on P4 and P5 systems (through PRPQ)	X	
	AIX 5.3	X	X
	pLinux		X
Physical Processor Management	**Dedicated** PLM runs on AIX 5.2 and/or P4 systems	X	
	Capped shared	X	
	Uncapped shared	X	X
Virtual Processor Management	**Virtual processor minimization for efficiency**	X	
	Virtual processor adjustment for physical processor growth	X	
Physical Memory Management	**Share-based**	X	
	Minimum and maximum entitlements	X	
Management Policy	**Entitlement-based**	X	X
	Goal-based		
	Application/middleware instrumentation required		
Management Domains	**Multiple management domains on a single CEC**	X	
	Cross platform (CEC)		
Administration	**Simple administration**	X	X
	Centralized LPAR monitoring (PLM command provides usage stats)	X	
	TOD-driven policy adjustment (PLM command supports new policy load based as TOD)	X	

Figure 3-25 Comparison of features of PLM and POWER Hypervisor

Partition Load Manager is set up in a partition or on another system running AIX 5L Version 5.2 ML4 or AIX 5L Version 5.3. Linux or i5OS support for PLM and the clients is not available. You can have other installed applications on the partition or system running the Partition Load Manager as well. A single instance of the Partition Load Manager can only manage a single server.

To configure Partition Load Manager, you can use the command line interface or the Web-based System Manager for graphical setup. For more information on the configuration and setup see 6.4, “Basic Partition Load Manager configuration” on page 216.

Partition Load Manager uses a client/server model to report and manage resource utilization. The clients (managed partitions) notify the PLM server when resources are either under- or over-utilized. Upon notification of one of these events, the PLM server makes resource allocation decisions based on a policy file defined by the administrator.

Partition Load Manager uses the Resource Monitoring and Control (RMC) subsystem for network communication, which provides a robust and stable framework for monitoring and managing resources. Communication with the HMC to gather system information and execute commands PLM requires a configured SSH connection.

Figure 3-26 shows an overview of the components of Partition Load Manager.

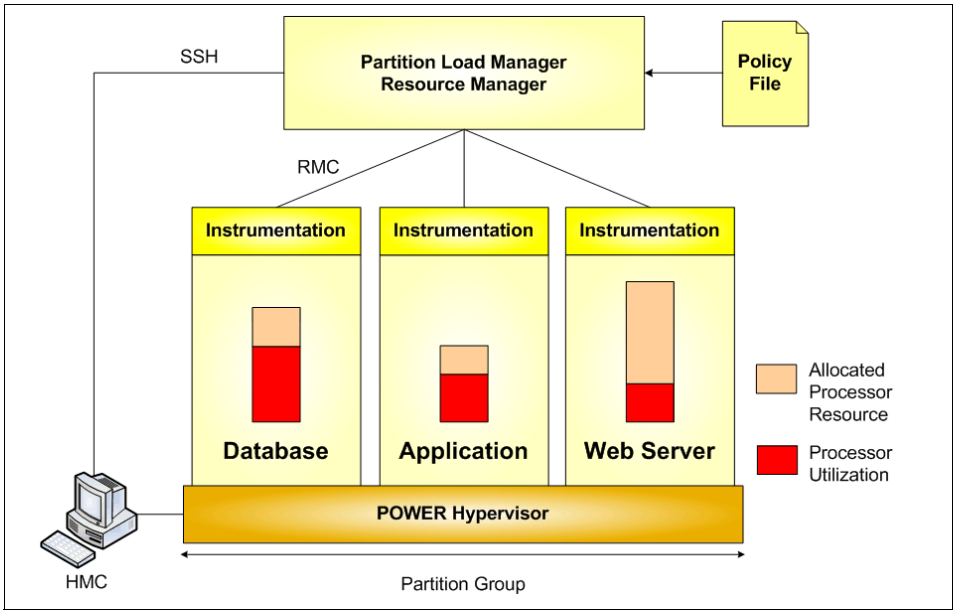

Figure 3-26 Partition Load Manager overview

The policy file defines managed partitions, their entitlements, their thresholds, and organizes the partitions into groups. Every node managed by PLM must be defined in the policy file along with several associated attribute values:

- Optional maximum, minimum, and guaranteed resource values
- The relative priority or weight of the partition
- Upper and lower load thresholds for resource event notification

For each resource (processor and memory), the administrator specifies an upper and a lower threshold for which a resource event should be generated. You can also choose to manage only one resource.

Partitions that have reached an upper threshold become resource *requesters*. Partitions that have reached a lower threshold become resource *donors*. When a request for a resource is received, it is honored by taking resources from one of three sources when the requester has not reached its maximum value:

- A pool of free, unallocated resources
- A resource donor
- A lower priority partition with excess resources over entitled amount

As long as there are resources available in the free pool, they will be given to the requester. If there are no resources in the free pool, the list of resource donors is checked. If there is a resource donor, the resource is moved from the donor to the requester. The amount of resource moved is the minimum of the delta values for the two partitions, as specified by the policy. If there are no resource donors, the list of excess users is checked.

When determining if resources can be taken from an excess user, the weight of the partition is determined to define the priority. Higher priority partitions can take resources from lower priority partitions. A partition's priority is defined as the ratio of its excess to its weight, where excess is expressed with the formula (current amount - desired amount) and weight is the policy-defined weight. A lower value for this ratio represents a higher priority. Figure 3-27 on page 96 shows an overview of the process for partitions.

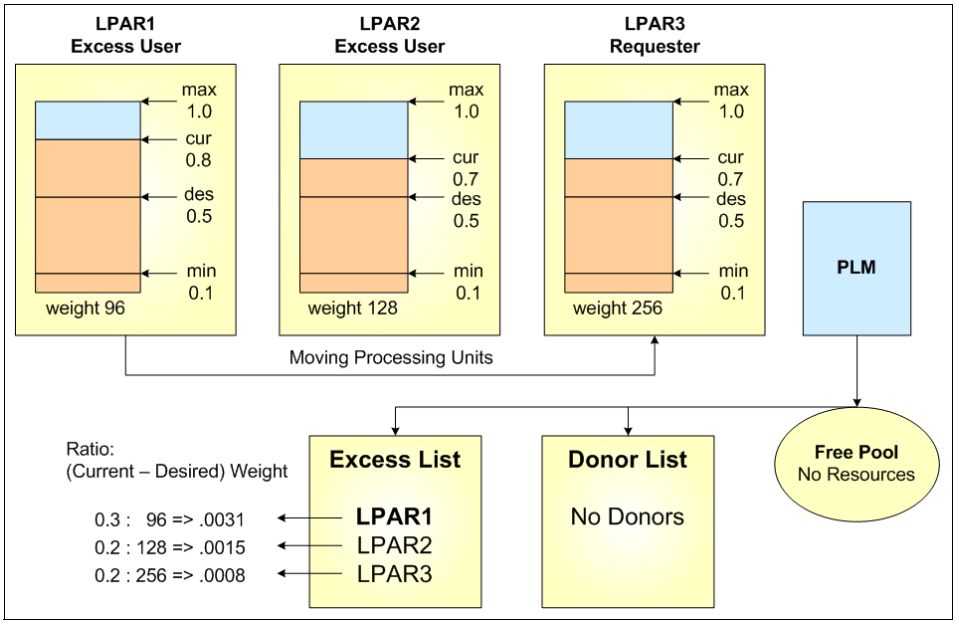

Figure 3-27 PLM resource distribution for partitions

In Figure 3-27, all partitions are capped partitions. LPAR3 is under heavy load and over its high CPU average threshold value becoming a requestor. There are no free resources in the free pool and no donor partitions available. PLM now checks the excess list to find a partition having resources allocated over its guaranteed value and with a lower priority. Calculating the priority, LPAR1 has the highest ratio number and therefore the lowest priority. PLM deallocates resources from LPAR1 and allocates them to LPAR3.

If the request for a resource cannot be honored, it is queued and re-evaluated when resources become available. A partition cannot fall below its minimum or rise above its maximum definition for each resource.

The policy file, once loaded, is static, and has no knowledge of the nature of the workload on the managed partitions. A partition's priority does not change upon the arrival of high priority work. The priority of partitions can only be changed by some action, external to PLM, by loading a new policy.

Partition Load Manager handles memory and both types of processor partitions: dedicated and shared processor partitions. All the partitions in a group must be of the same processor type.

3.7.1 Memory management

PLM manages memory by moving Logical Memory Blocks (LMBs) across partitions. To determine when there is demand for memory, PLM uses two metrics:

- Utilization percentage (ratio of memory in use to available)
- The page replacement rate

For workloads that result in significant file caching, the memory utilization on AIX may never fall below the specified lower threshold. With this type of workload, a partition may never become a memory donor, even if the memory is not currently being used.

In the absence of memory donors, PLM can only take memory from excess users. Since the presence of memory donors cannot be guaranteed, and is unlikely with some workloads, memory management with PLM may only be effective if there are excess users present. One way to ensure the presence of excess users is to assign each managed partition a low guaranteed value, such that it will always have more than its guaranteed amount. With this sort of policy, PLM will always be able to redistribute memory to partitions based on their demand and priority.

3.7.2 Processor management

For dedicated processor partitions, PLM moves physical processors, one at a time, from partitions that are not utilizing them, to partitions that have demand for them. This enables dedicated processor partitions running AIX 5L Version 5.2 and AIX 5L Version 5.3 to better utilize their resources. If one partition needs more processor capacity, PLM automatically moves processors from a partition that has idle capacity.

For shared processor partitions, PLM manages the entitled capacity and the number of virtual processors (VPs) for capped or uncapped partitions. When a partition has requested more processor capacity, PLM will increase the entitled capacity for the requesting partition if additional processor capacity is available. For uncapped partitions, PLM can increase the number of virtual processors to increase the partition's potential to consume processor resources under high load conditions. Conversely, PLM will also decrease entitled capacity and the number of virtual processors under low-load conditions, to more efficiently utilize the underlying physical processors.

With the goal of maximizing a partition's and the system's ability to consume available processor resources, the administrator now has two choices:

1. Configure partitions that have high workload peaks as uncapped partitions with a large number of virtual processors. This has the advantage of allowing these partitions to consume more processor resource when it is needed and available, with very low latency and no dynamic reconfiguration. For example, consider a 16-way system utilizing two highly loaded partitions configured with eight virtual processors each, in which case, all physical processors could have been fully utilized. The disadvantage of this approach is that when these partitions are consuming at or below their desired capacity there is an overhead associated with the large number of virtual processors defined.
2. Use PLM to vary the capacity and number of virtual processors for the partitions. This has the advantages of allowing partitions to consume all of the available processor resource on demand, and it maintains a more optimal number of VPs. The disadvantage to this approach is that since PLM performs dynamic reconfiguration operations to shift capacity to and from partitions, there is a much higher latency for the reallocation of resources. Though this approach offers the potential to more fully utilize the available resource in some cases, it significantly increases the latency for redistribution of available capacity under a dynamic workload, since dynamic reconfiguration operations are required.

3.7.3 Limitations and considerations

You must consider the following limitations when managing your system with the Partition Load Manager:

- The Partition Load Manager can be used in partitions running AIX 5L Version 5.2 ML4 or AIX 5L Version 5.3. Linux or i5OS support is not available.
- A single instance of the Partition Load Manager can only manage a single server. However, multiple instances of the Partition Load Manager can be run on a single system, each managing a different server.
- The Partition Load Manager cannot move I/O resources between partitions. Only processor and memory resources can be managed by Partition Load Manager.
- Partition Load Manager requires HMC Release 3 Version 2.6 or newer on an HMC and an IBM @server p5 system.

4

Virtual I/O Server configuration

The Virtual I/O Server is an appliance that provides virtual storage and shared Ethernet capability to client logical partitions on a POWER5 system. It allows a physical adapter with attached disks on the Virtual I/O Server partition to be shared by one or more partitions, enabling clients to consolidate and potentially minimize the number of physical adapters.

In this chapter, the following topics related to the Virtual I/O Server are discussed:

- Operating environment
- Capabilities of the Virtual I/O Server
- Installation of the Virtual I/O Server
- Configuration approaches for high availability
- Maintenance and monitoring
- Security issues of the Virtual I/O Server
- Interaction with AIX partitions
- Interaction with Linux partitions
- Interaction with i5/OS partitions

4.1 Getting started

This section provides the following information about the operating environment of the Virtual I/O Server:

- Command line interface of the Virtual I/O Server
- Supported hardware resources
- Software

4.1.1 Command line interface

The Virtual I/O Server provides a restricted scriptable command line user interface (CLI). All aspects of Virtual I/O Server administration are accomplished through the CLI, including:

- Device management (physical, virtual, LVM)
- Network configuration
- Software installation and update
- Security
- User management
- Installation of OEM software
- Maintenance tasks

For the initial logon to the Virtual I/O Server, use the user ID padmin, which is the prime administrator. When logging in, you are prompted for a new password, so there is no default password to remember.

Upon logging into the I/O server, you will be placed into a restricted Korn shell. The restricted Korn shell works the same way as a regular Korn shell with some restrictions. Specifically, users cannot do the following:

- Change the current working directory
- Set the value of the SHELL, ENV, or PATH variables
- Specify the path name of the command that contains a redirect output of a command with a **>**, **>|**, **<>**, or **>>**.

As a result of these restrictions, you are not able to execute commands that are not accessible to your PATH. In addition, these restrictions prevent you from directly sending the output of the command to a file, requiring you to pipe the output to the tee command instead.

After you are logged on, you can type **help** to get an overview of the supported commands, as in the following:

```
$ help

Install Commands       Physical Volume Commands    Security Commands
 updateios              lspv                        lsgcl
 lssw                   migratepv                   cleargcl
 ioslevel                                           lsfailedlogin
 remote_management     Logical Volume Command
 oem_setup_env          lslv                       UserID Commands
 oem_platform_level     mklv                        mkuser
 license                extendlv                    rmuser
                        rmlvcopy                    lsuser
LAN Commands            rmlv                        passwd
 mktcpip                mklvcopy                    chuser
 hostname
 cfglnagg
 netstat               Volume Group Commands       Maintenance Commands
 entstat                lsvg                        chlang
 cfgnamesrv             mkvg                        diagmenu
 traceroute             chvg                        shutdown
 ping                   extendvg                    fsck
 optimizenet            reducevg                    backupios
 lsnetsvc               mirrorios                   savevgstruct
                        unmirrorios                 restorevgstruct
Device Commands         activatevg                  starttrace
 mkvdev                 deactivatevg                stoptrace
 lsdev                  importvg                    cattracerpt
 lsmap                  exportvg                    bootlist
 chdev                  syncvg                      snap
 rmdev                                              startsysdump
 cfgdev                                             topas
 mkpath                                             mount
 chpath                                             unmount
 lspath                                             showmount
 rmpath                                             startnetsvc
                                                    errlog
                                                    stopnetsvc
```

To receive further help on these commands, use the **help** command, as shown in the following:

```
$ help errlog
Usage: errlog [-ls | -rm Days]

       Displays or clears the error log.

       -ls         Displays information about errors in the error log file
                   in a detailed format.

       -rm         Deletes all entries from the error log older than the
                   number of days specified by the Days parameter.
```

The Virtual I/O Server command line interface supports two execution modes:

- Traditional mode
- Interactive mode

The traditional mode is for single command execution. In this mode, you execute one command at a time from the shell prompt. For example, to list all virtual devices, enter the following:

```
#ioscli lsdev -virtual
```

To reduce the amount of typing required in traditional shell level mode, an alias has been created for each sub-command. With the aliases set, you are not required to type the **ioscli** command. For example, to list all devices of type adapter, you can enter the following:

```
#lsdev -type adapter
```

In interactive mode the user is presented with the **ioscli** command prompt by executing the **ioscli** command without any sub-commands or arguments. From this point on, **ioscli** commands are executed one after the other without having to retype **ioscli**. For example, to enter interactive mode, enter:

```
#ioscli
```

Once in interactive mode, to list all virtual devices, enter:

```
#lsdev -virtual
```

External commands, such as **grep** or **sed**, cannot be executed from the interactive mode command prompt. You must first exit interactive mode by entering **quit** or **exit**.

4.1.2 Hardware resources managed

The optional Advanced POWER Virtualization feature that enables Micro-Partitioning on a p5 server provides the Virtual I/O Server installation CD. A logical partition with enough resources to share to other partitions is required. The following minimum hardware requirements must be available to create the Virtual I/O Server:

POWER5 server The Virtual I/O capable machine

Hardware management console
HMC to create the partition and assign resources.

Storage adapter The server partition needs at least one storage adapter.

Physical disk If you want to share your disk to client partitions you need a disk large enough to make *sufficient-sized* logical volumes on it.

Ethernet adapter If you want to allow securely routed network traffic from a virtual Ethernet to a real network adapter.

Memory At least 128 MB of memory.

Virtual I/O Server Version 1.1 is designed for selected configurations that include specific models of IBM and other vendor storage products.

Consult your IBM representative or Business Partner for the latest information and included configurations.

Virtual devices exported to client partitions by the Virtual I/O Server must be attached through one of the following physical adapters:

- PCI 4-Channel Ultra3 SCSI RAID Adapter (#2498)
- PCI-X Dual Channel Ultra320 SCSI RAID Adapter (#5703)
- Dual Channel SCSI RAID Enablement Card (#5709)
- PCI-X Dual Channel Ultra320 SCSI Adapter (#5712)
- 2 Gigabit Fibre Channel PCI-X Adapter (#5716)
- 2 Gigabit Fibre Channel Adapter for 64-bit PCI Bus (#6228)
- 2 Gigabit Fibre Channel PCI-X Adapter (#6239)

Careful planning is recommended before you begin the configuration and installation of your I/O server and client partitions. Depending on the type of workload and needs of an application, consider mixing virtual and physical devices. For example, if your application benefits from fast disk access, then plan a physical adapter dedicated to this partition.

4.1.3 Software packaging and support

Installation of the Virtual I/O Server partition is performed from a special **mksysb** CD that is provided to customers that order the Advanced POWER Virtualization feature, at an additional charge. This is dedicated software only for the Virtual I/O Server operations, so the Virtual I/O Server software is only supported in Virtual I/O Server partitions.

You can install the Virtual I/O Server from CD or using NIM on Linux (NIMoL) from the HMC. For more information on the installation of the Virtual I/O Server refer to 5.3.3, “Virtual I/O Server software installation” on page 157.

The Virtual I/O Server supports the following operating systems as Virtual I/O clients:

- IBM AIX 5L Version 5.3
- SUSE LINUX Enterprise Server 9 for POWER
- Red Hat Enterprise Linux AS for POWER Version 3

4.1.4 Virtual I/O Server software installation

This section provides a description of the installation procedure of the Virtual I/O Server software to the formerly created partition I/O_Server_1. We assume that you are familiar with the tasks of a basic AIX installation.

See 5.3.1, “Creating the Virtual I/O Server partition” on page 140 for information on how to create this partition.

The following steps show the installation using the CD install device.

1. Activate the I/O_Server_1 partition by right-clicking the partition name and select the **Activate** bar as shown in Figure 4-1. Select the default profile you used to create this server.

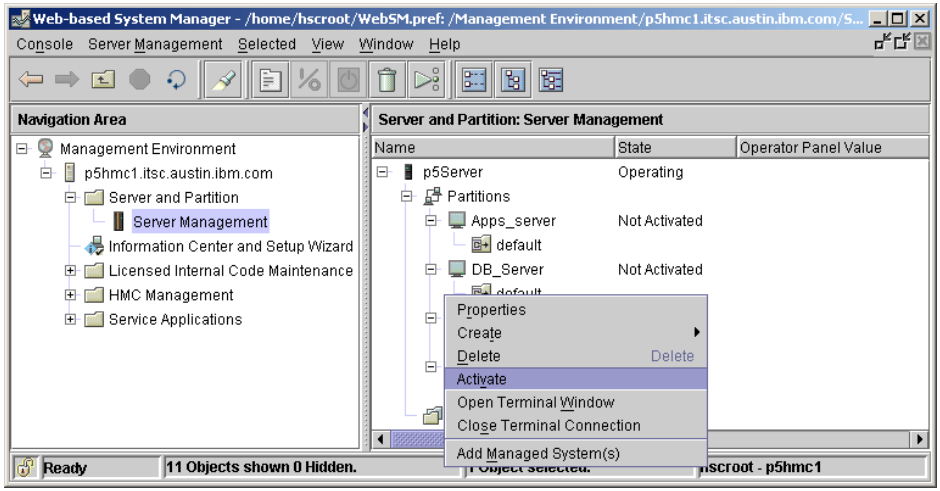

Figure 4-1 Activate I/O_Server_1 partition

2. Select the default profile, activate the **Open a terminal window or console session** checkbox, and click the **(Advanced...)** button, as shown in Figure 4-2.

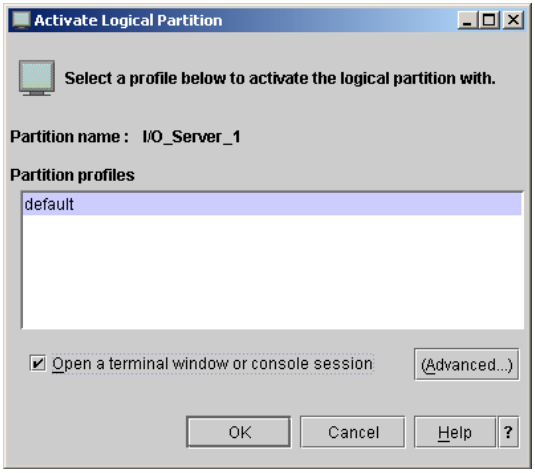

Figure 4-2 Selecting the profile

3. Choose the SMS boot mode as shown in Figure 4-3, click the **OK** button in this window to return to the previous window. When at the previous window, click the **OK** button to activate the partition and launch a terminal window.

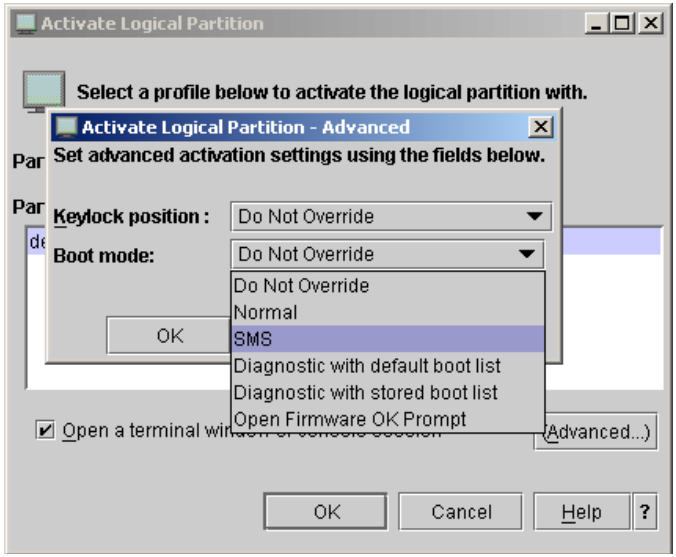

Figure 4-3 Choosing SMS boot mode

4. Figure 4-4 shows you an IBM @server® pSeries SMS menu. Proceed through the installation procedure as for any other AIX installation, choosing CD as the installation device.

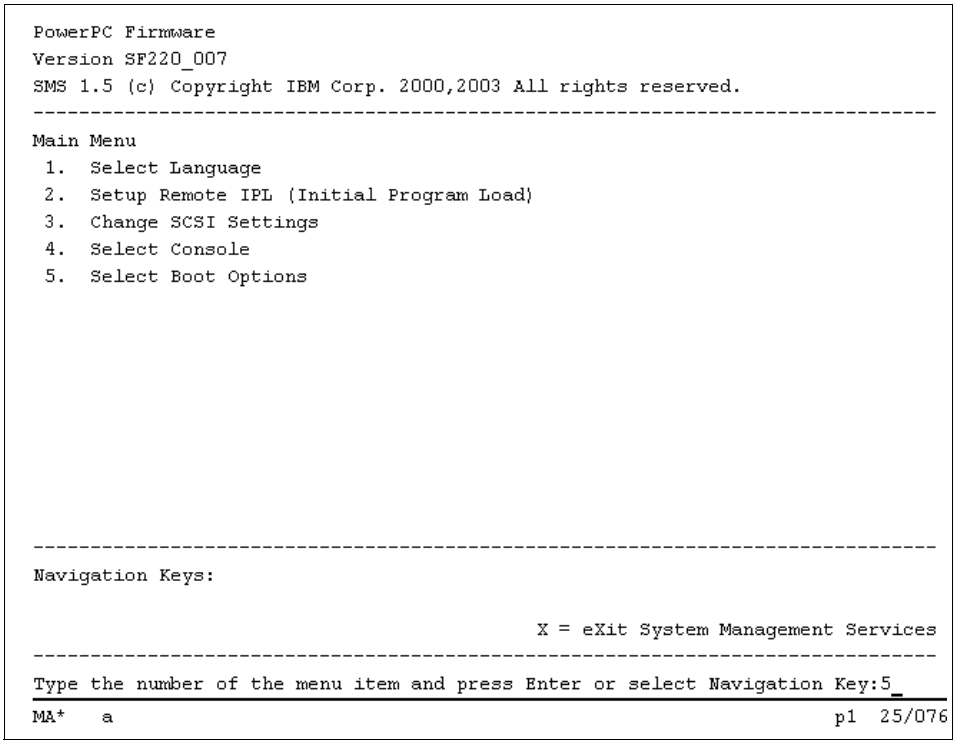

Figure 4-4 SMS menu

5. When the installation procedure has finished, use `padmin` for username at the login prompt, choose a new password, and accept the license using the **`licence`** command on **$** prompt, as shown in Figure 4-5. Now, for example, you can use the **`lspv`** command to show the available disks.

```
0513-059 The aixmibd Subsystem has been started. Subsystem PID is 655468.
0513-059 The muxatmd Subsystem has been started. Subsystem PID is 659536.
Finished starting tcpip daemons.
Starting NFS services:
0513-059 The biod Subsystem has been started. Subsystem PID is 622644.
0513-059 The rpc.lockd Subsystem has been started. Subsystem PID is 663688.
Completed NFS services.
Virtual I/O Server
login: 0513-059 The ctrmc Subsystem has been started. Subsystem PID is 356530.

Virtual I/O Server
login: padmin
[compat]: 3004-610 You are required to change your password.
        Please choose a new one.

padmin's New password:
Enter the new password again:

$ license -accept
$ lspv
hdisk0          00cddedc00efe72a                    rootvg          active
hdisk1          00cddedc15d81aaf                    None
hdisk2          00cddedc003416f4                    None
hdisk3          00cddedc04a9aa04                    None
$ _
MA*    a                                                              p1  25/003
```

Figure 4-5 Finished Virtual I/O Server installation

With this step you have finished the installation of the Virtual I/O Server. The I/O_Server_1 partition is now ready for further configuration.

4.2 Basic configuration

The Virtual I/O Server provides the virtual SCSI and shared Ethernet adapter virtual I/O to client partitions. This is accomplished by assigning physical devices to the Virtual I/O Server partition, then configuring virtual adapters on the clients to allow communication between the client and the Virtual I/O Server.

Using virtual I/O devices facilitates the following functions:

- Sharing of physical resources between partitions on a POWER5 system

- Providing Virtual SCSI and Shared Ethernet Adapter function to client partitions
- Creation of partitions without requiring additional physical I/O resources
- Creation of more partitions than I/O slots or physical devices with the ability for partitions to have dedicated I/O, virtual I/O, or both
- Maximizing the utilization of physical resources on a POWER5 system

4.2.1 Ethernet adapter sharing

Shared Ethernet Adapter (SEA) enables the client partitions to communicate with other systems outside the CEC without requiring physical Ethernet adapters in the partitions. This is accomplished by sharing the physical Ethernet adapters in the Virtual I/O Server partition.

VLANs that are bridged outside using a Shared Ethernet Adapter require a Virtual Ethernet adapter to have the trunk adapter setting on. This Virtual Ethernet adapter is assigned to the Virtual I/O Server partition using the HMC. The Shared Ethernet Adapter setup commands are then run on the Virtual I/O Server to create associations between the physical and virtual adapters. For a detailed configuration scenario, see “Creating a shared Ethernet adapter” on page 177.

To configure a trunk Virtual Ethernet adapter on the HMC, right-click the partition profile of the Virtual I/O Server partition, and open the properties of the profile. Now choose the Virtual I/O tab. Figure 4-6 on page 110 shows the HMC panel to create a virtual adapter.

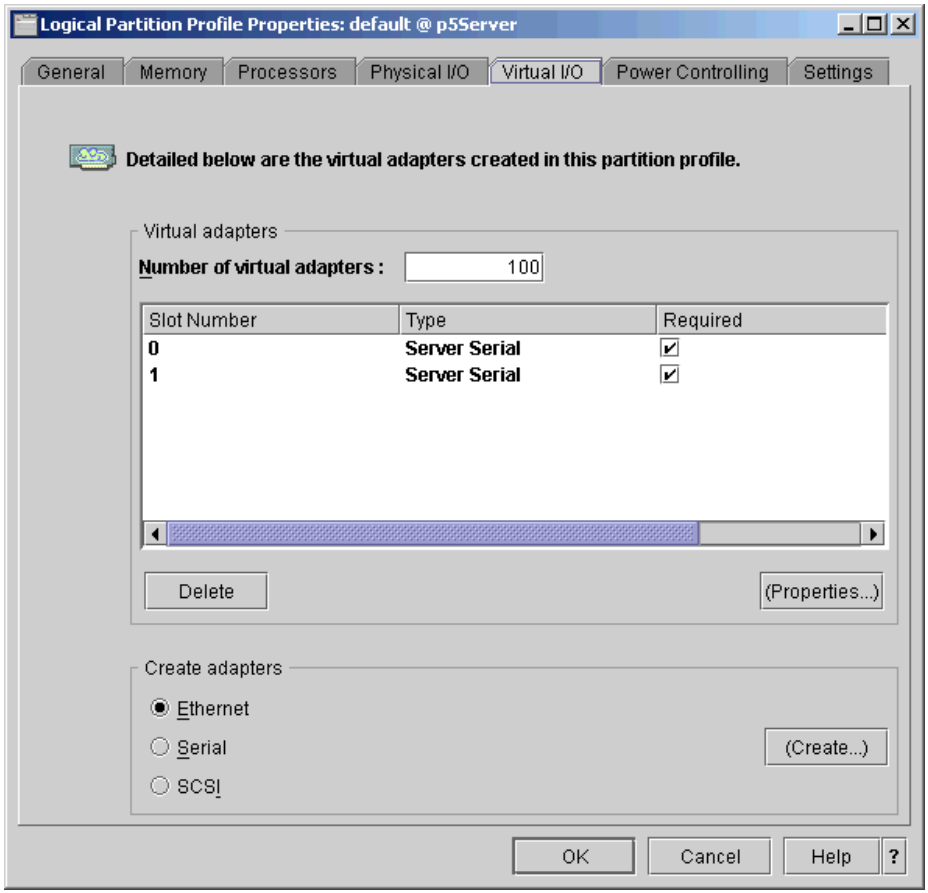

Figure 4-6 Creating the trunk Virtual Ethernet adapter on the HMC

Make sure that the value in “Number of virtual adapters” is higher than highest used Slot Number. To allow additional dynamically configured adapters, choose a number such that you can still add some more virtual adapters later on. This value is similar to the maximum value of the processor and memory definition. To change it once set you have to shut down and reactivate your partition.

Select the Ethernet radio button in the Create Adapter panel, then click the **(Create)** button to proceed.

Figure 4-7 on page 111 shows the panel where you can set the properties of the Virtual Ethernet adapter.

The slot number used for this virtual Ethernet adapter is used to identify the virtual adapter within the logical partition. The combination of the slot number and the logical partition ID uniquely identifies this slot within the managed system.

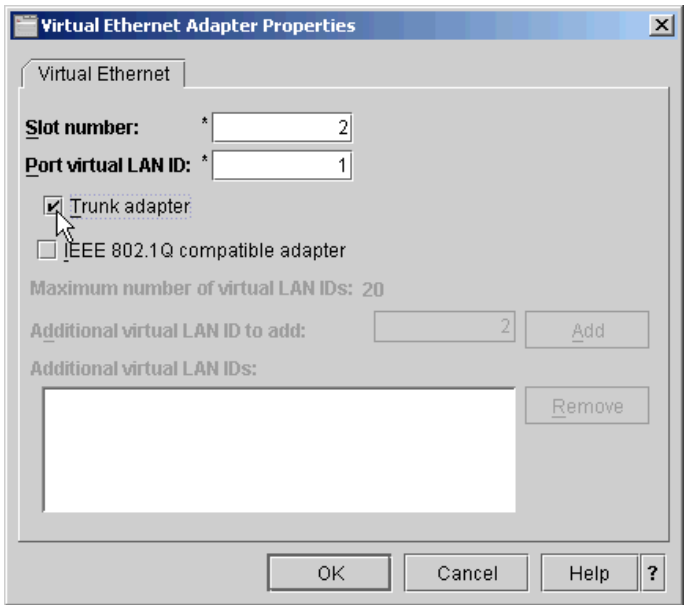

Figure 4-7 Virtual Ethernet Adapter Properties panel

Each virtual Ethernet adapter has assigned a Port virtual LAN ID (PVID) or a virtual LAN ID (VID) number. Selecting the “IEEE 802.1Q compatible adapter” option allows additional virtual LAN IDs to be configured. This allows the Virtual Ethernet adapter to be part of additional VLANs as specified in the IEEE 802.1Q network standard.

Virtual Ethernet adapters can communicate with each other only if they are assigned to the same PVID or VID number.

Important: The “Trunk adapter” checkbox *must* be selected on each Virtual Ethernet adapter that will be mapped to create a Shared Ethernet Adapter.

When a virtual adapter is defined in a partition profile, you *must* shut down and reactivate the partition to make the adapter available.

If a network connection is already configured and your RMC connection is working correctly, the Virtual Ethernet adapter can be added dynamically to the partition. Right-click the partition name and choose **Dynamic Logical**

Partitioning → **Virtual Adapter Resources** → **Add/Remove panel** as shown in Figure 4-8. See 3.3.5, “Dynamic partitioning” on page 64 for more information.

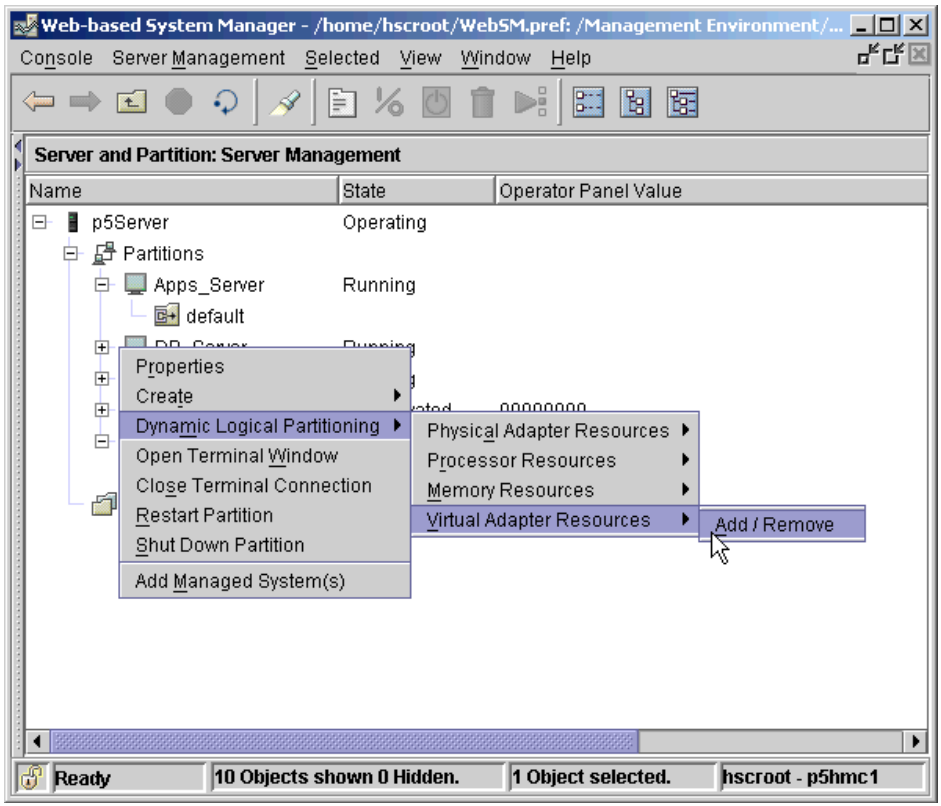

Figure 4-8 Dynamically adding or removing virtual adapters to a partition

Important: If you want to keep your dynamic virtual adapter changes after reactivation of your partition, you also have to add or remove the defined adapters to the partition profile.

After you assigned a virtual adapter to the Virtual I/O Server partition dynamically, you have to run the **`cfgdev`** command on the Command Line Interface of the Virtual I/O Server. This command refreshes the configuration from the operating system point of view.

The next step is to define the Shared Ethernet Adapter on the Virtual I/O Server. The syntax of the **mkvdev** command to create a Shared Ethernet Adapter is as follows:

```
mkvdev  -sea TargetDevice -vadapter VirtualEthernetAdapter ...
              -default DefaultVirtualEthernetAdapter
              -defaultid SEADefaultPVID [-attr Attributes=Value ...]
```

Using the example in Figure 4-9, the target devices are the physical adapters (for example, ent0 and ent1). The virtual devices are ent2, ent3, and ent4, and the defaultid is the default PVID associated with the default virtual Ethernet adapter.

Important: To set up the Shared Ethernet Adapter, all involved virtual and physical Ethernet interfaces have to be unconfigured (down or detached).

The following commands are required to set up Shared Ethernet Adapter for this example:

```
$mkvdev –sea ent0 –vadapter ent2 –default ent2 –defaultid 1
$mkvdev –sea ent1 –vadapter ent3 ent4 –default ent3 –defaultid 2
```

In the second example, the physical Ethernet adapter is ent1. With the **mkvdev** command we map the virtual Ethernet adapter ent3 and ent4 to the physical adapter. Additionally, ent3 is defined as a default adapter with the default VLAN ID of 2. This means that untagged packets received by the Shared Ethernet Adapter are tagged with the VLAN 2 ID and are send to the virtual Ethernet adapter ent3.

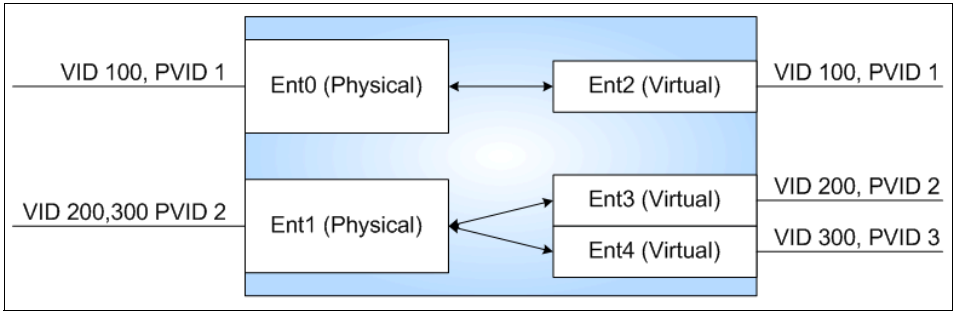

Figure 4-9 Example of an I/O server partition bridge

After running the **mkvdev** command, the system will create the Shared Ethernet Adapter ent5. You now can configure the ent5 interface with an IP address using the **mktcpip** command.

The syntax of the **mktcpip** command is as follows:

```
mktcpip -hostname HostName -inetaddr Address -interface Interface
        [-start] [-netmask SubnetMask] [-cabletype CableType]
        [-gateway Gateway] [-nsrvaddr NameServerAddress
        [-nsrvdomain Domain]]
```

Setting up the hostname and IP address for the Shared Ethernet Adapter is shown in the following example:

```
$ mktcpip -hostname p5_2ioserver1 -inetaddr 9.3.5.150 -interface en5 -netmask
255.255.255.0 -gateway 9.3.5.41
```

Restriction: Multiple subnets may connect externally using the same Shared Ethernet Adapter; however, each subnet must be tagged with a different VLAN ID.

4.2.2 Virtual SCSI disk

Virtual SCSI facilitates the sharing of physical disk resources (I/O adapters and devices) between logical partitions. Virtual SCSI enables partitions to access SCSI disk devices without requiring that physical resources be allocated to the partition. Partitions maintain a client/server relationship in the Virtual SCSI environment. Partitions that contain Virtual SCSI devices are referred to as client partitions while the partition that owns the physical resources (adapters, devices) is the Virtual I/O Server.

The Virtual SCSI disks are defined as logical volumes or as physical volumes in the Virtual I/O Server. All standard conventional rules apply to the logical volumes. The logical volumes appear as real devices (hdisks) in the client partitions and can be used as a boot device and as a NIM target.

Once a virtual disk is assigned to a client partition, the Virtual I/O Server must be available before the client partitions are able to boot.

Defining volume groups and logical volumes

If you want to create a logical volume to assign to your client partition use the **mklv** command. To create the logical volume on a separate disk you first have to create a volume group and assign one or more disks using the **mkvg** command.

The basic syntax of the **mkvg** command to create a volume group on the Virtual I/O Server is as follows:

```
mkvg [-f] [-vg VolumeGroup] PhysicalVolume ...
```

The basic syntax of the **mklv** command to create a logical volume on the Virtual I/O Server is as follows:

```
mklv [-mirror] [-lv NewLogicalVolume | -prefix Prefix]
     VolumeGroup Size [PhysicalVolume ...]
```

Create a volume group and assign a disk to this volume group using the **mkvg** command as shown. In this example the name of the volume group is rootvg_clients:

```
$ mkvg -f -vg rootvg_clients hdisk2
rootvg_clients
```

Define the logical volume which will be visible as a disk to the client partition. The size of this logical volumes will act as the size of disks which will be available to the client partition. Use the **mklv** command to create a 2 GB size logical volume called rootvg_dbsrv as follows:

```
$ mklv -lv  rootvg_dbsrv rootvg_clients 2G
rootvg_dbsrv
```

Defining the virtual SCSI server adapter on the HMC

On the Virtual I/O Server partition profile, select the **Virtual I/O** tab to create a Virtual SCSI Server adapter. Choose the **SCSI** radio button and click **(Create)** to proceed. Figure 4-10 on page 116 shows the properties panel to configure the Virtual SCSI Adapter.

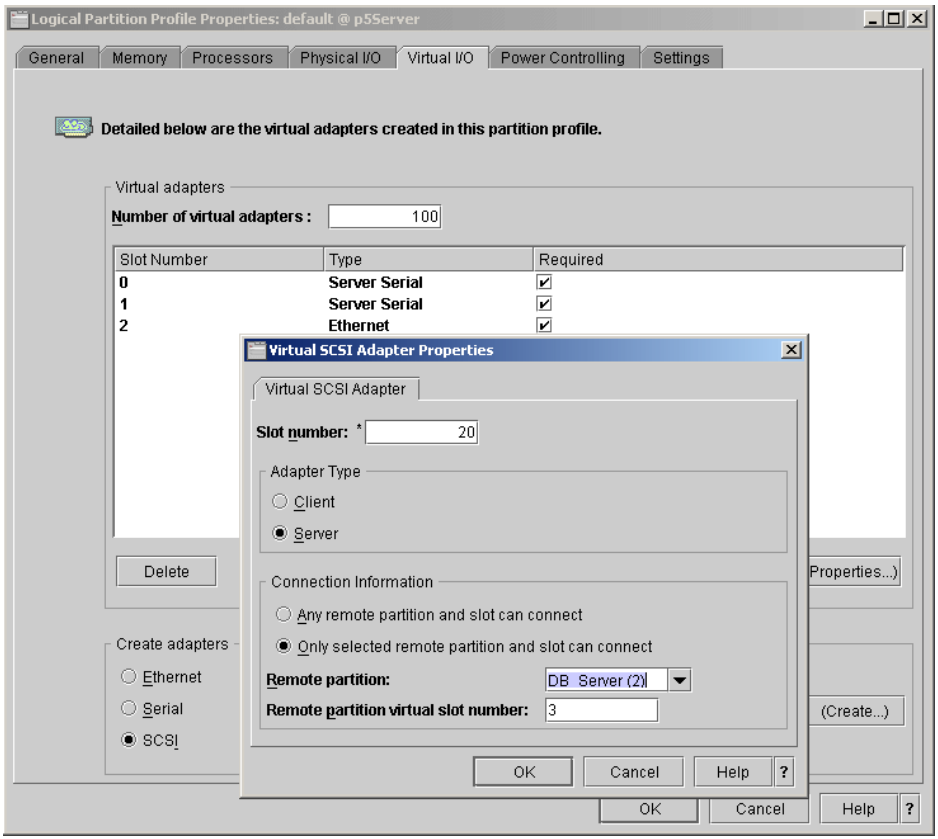

Figure 4-10 Virtual SCSI Adapter Properties panel on the I?O Server site

The slot number is used to identify the virtual adapter within the logical partition. The combination of the slot number and the logical partition ID uniquely identify this slot within the managed system.

This slot number does not refer to any physical hardware location on your system. You can therefore assign slot numbers to virtual adapters in any way that makes sense to you, provided that you follow the following guidelines:

- You can use any slot number from 2 up to (but not including) the maximum number of virtual adapters. Slot 0 and 1 are reserved for system-created virtual adapters. By default, the system displays the lowest unused slot number for this logical partition.
- You cannot use a slot number that was used for any other virtual adapter on the same logical partition.

The Adapter Type assigned on the Virtual I/O Server partition is Server.

In the Connection Information area, you have to define whether to allow any partitions to connect to this SCSI drive or only one dedicated partition.

If the client partition is not defined yet, you can enter a unique Partition ID into the Remote Partition field. Use this Partition ID when defining your client partition. The HMC will automatically assign this partition as a client for virtual SCSI resources. The Remote partition virtual slot number has to match with the slot number defined for your client SCSI adapter on the client partition.

Click the OK button and the Virtual SCSI Server adapter is ready to be configured from the Command Line Interface (CLI) of the Virtual I/O Server. When you define the adapter in the partition profile you have to shut down your partition and activate it to make the adapter available or use the dynamic reconfiguration process described in the 3.3.3, “Capacity on Demand” on page 62.

Defining the virtual SCSI client adapter on the HMC

The Virtual SCSI Client Adapter is defined in the same panel in the client partition profile. Figure 4-11 on page 118 shows the client partition DB_Server definition for the Virtual SCSI client adapter.

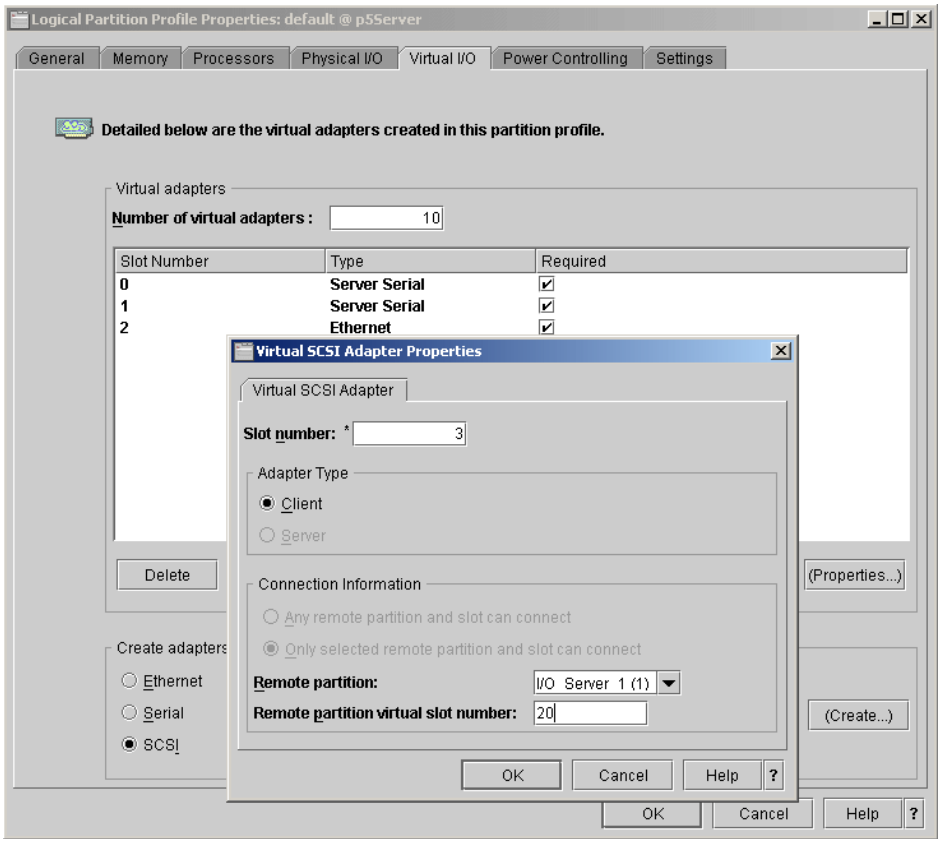

Figure 4-11 Virtual SCSI Adapter Properties panel on the client partition site

The Client SCSI adapter has Slot number 3 defined, which matches to the Remote partition virtual slot number on the Virtual I/O Server partition.

The Adapter Type assigned on the client partition is Client.

In the Connection Information area, you select the hosting I/O Server partition and fill in the Remote partition virtual slot number. In our example this is slot number 20.

Create the virtual target device on the Virtual I/O Server

The basic command to map the Virtual SCSI with the logical volume or hdisk is as follows:

```
mkvdev  -vdev TargetDevice -vadapter VirtualSCSIServerAdapter
        [-dev DeviceName]
```

Run the **lsdev -virtual** command to make sure that your new virtual SCSI adapter is available, as follows:

```
$ lsdev -virtual
name            status     description

ent2            Available  Virtual I/O Ethernet Adapter (l-lan)
vhost0          Available  Virtual SCSI Server Adapter
vhost1          Available  Virtual SCSI Server Adapter
vhost2          Available  Virtual SCSI Server Adapter
vsa0            Available  LPAR Virtual Serial Adapter
```

The next step is to create a virtual target device, which maps the Virtual SCSI Server adapter vhost0 to the logical volume rootvg_dbsrv created previously. When you do not use the -dev flag, the default name of the Virtual Target Device adapter is vtscsi*x*. Run the **mkvdev** command as follows to perform this task:

```
$ mkvdev -vdev rootvg_dbsrv -vadapter vhost0 -dev vdbsrv
vdbsrv Available
```

If you want to map a physical volume to the Virtual SCSI Server Adapter use hdisk*x* instead of the logical volume devices for the -vdev flag.

The **lsdev** command shows the newly created Virtual Target Device adapter:

```
$ lsdev -virtual
name            status     description

vhost0          Available  Virtual SCSI Server Adapter
vsa0            Available  LPAR Virtual Serial Adapter
vdbsrv          Available  Virtual Target Device - Logical Volume
```

The **lsmap** command shows us the logical connections between newly created devices, as follows:

```
$ lsmap -vadapter vhost0
SVSA            Physloc                                     Client PartitionID
--------------- ------------------------------------------- ------------------
vhost0          U9111.520.10DDEEC-V1-C20                     0x00000000

VTD                   vdbsrv
LUN                   0x8100000000000000
Backing device        rootvg_dbsrv
Physloc
```

Here you also see the physical location being a combination of the slot number, in this case 20, and the logical partition ID.

At this point the created virtual device can be attached from the client partition. You can now activate your partition into the SMS menu and install the AIX

operating system on the virtual disk or add an additional virtual disk using the **cfgmgr** command.

The Client PartitionID shows up as soon as the client partition is active.

4.2.3 Limitations and considerations

The Virtual I/O Server software is a dedicated software only for the virtual I/O Server operations, and there is no possibility of running other applications in the Virtual I/O Server partition.

There is no option to get the Virtual I/O Server partition pre-installed on new systems. At the time of writing, the pre-install manufacturing process does not allow the Virtual I/O Server partition to be pre-installed.

Other limitations can occur because of resource shortages. The Virtual I/O Server should be properly configured with enough resources. The most important are the processor resources. If a Virtual I/O Server has to host a lot of resources to other partitions, you must ensure that enough processor power is available. In case of high load, or high traffic across virtual Ethernet adapters and virtual disks, partitions can observe delays in accessing resources.

Logical volume limitation

The virtual I/O server operating system allows you to define up to 1024 logical volumes per volume group, but the actual number you can define depends on the total amount of physical storage defined for that volume group and the size of the logical volumes you configure.

Table 4-1 shows the limitations for logical storage management.

Table 4-1 Limitations for logical storage management

Category	Limit
Volume group	4096 per system
Physical volume	1024 per volume group
Physical partition	2097152 per volume group
Logical volume	4096 per volume group
Logical partition	Based on physical partitions

4.3 Advanced configuration

This section discusses the different configuration scenarios of the Virtual I/O Server to achieve a higher availability for the virtual client partitions.

In Chapter 5, “AIX and Virtual I/O Server configuration scenarios” on page 137, you find a detailed example of how to configure a basic scenario with one Virtual I/O Server and a high availability scenario with two Virtual I/O Servers.

4.3.1 Providing higher availability for Virtual I/O Server

When we talk about providing high availability for the Virtual I/O Server we are talking about incorporating the I/O resources (physical and virtual) on the Virtual I/O Server as well as the client partitions into a configuration that is designed to eliminate single points of failure.

The Virtual I/O Server is a single point of failure. In case of a crash of the Virtual I/O Server, the client partitions will see I/O errors and not be able to access the adapters and devices which are hosted by the Virtual I/O Server.

However, redundancy can be built into the configuration of the physical and virtual I/O resources at several stages.

Since the Virtual I/O Server is an AIX-based appliance, redundancy for physical devices attached to the virtual I/O server can be provided by using capabilities such as LVM mirroring, Multipath I/O, and EtherChannel.

Note: When activating the EtherChannel you may see some "`Unsupported ioctl in device driver`" errors if you are using virtual Ethernets in your Link Aggregation. These errors can be ignored. Refer to 5.4.4, “Creating Link Aggregation in client partitions” on page 191 for more details.

Figure 4-12 on page 122 shows a single Virtual I/O Server configuration with disk and network attachment. The disks are mirrored through LVM. The two physical network adapters are configured as a Link Aggregation in Network Interface Backup (NIB) mode.

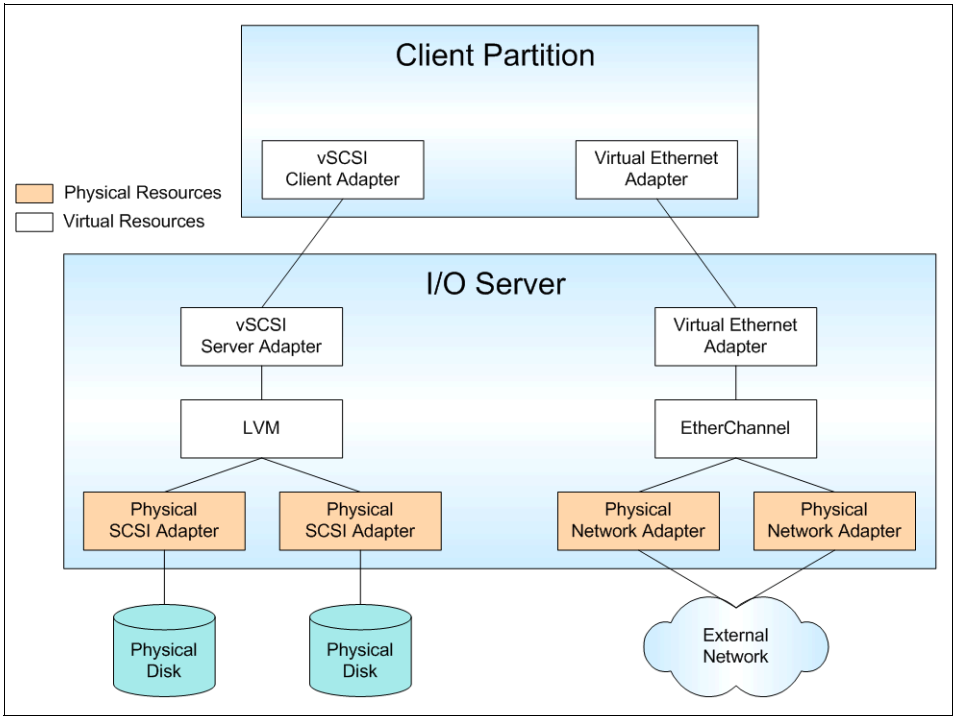

Figure 4-12 Single virtual I/O server configuration

While this kind of configuration protects you from the failure of one of the physical components, like a disk or network adapter, it will still cause the client partition to loose access to its devices if the Virtual I/O Server fails.

The Virtual I/O Server itself can be made redundant by running a second instance of it in another partition.

When running two instances of the Virtual I/O Server, you can use LVM mirroring, Multipath I/O, Link Aggregation, or Multipath routing with dead gateway detection in the client partition to provide highly available access to virtual resources hosted in separate Virtual I/O Server partitions.

Many configurations are possible and depend on the available hardware resources as well as your requirements. The following sections describe some configuration approaches in more detail. Also see Chapter 5, “AIX and Virtual I/O Server configuration scenarios” on page 137, where a possible configuration is covered with detailed setup steps.

Network interface backup

Figure 4-13 on page 124 shows a configuration using network interface backup.

The client partition has two virtual Ethernet adapters. Each adapter is assigned to a different VLAN (using the PVID). Each Virtual I/O Server is configured with a Shared Ethernet Adapter which bridges traffic between the virtual Ethernet and the external network. Both Shared Ethernet Adapters should be able to connect to the same set of hosts in the external network.

Each of the Shared Ethernet Adapters is assigned to a different VLAN (using PVID). By using two VLANs, network traffic is separated so that each virtual Ethernet adapter in the client partition seems to be connected to a different Virtual I/O Server.

The two virtual Ethernet adapters in the client partition are configured as an EtherChannel using Network Interface Backup. The Link Aggregation is configured with a primary adapter and a backup, and the operation mode is left as the default standard mode. Additionally, the EtherChannel is configured with an “Internet Address to Ping.” This address will be periodically pinged by the EtherChannel to determine if connectivity to the external network exists. Typically a router that should be always available is used as the target for the ping.

Even though a Link Aggregation with more than one primary virtual Ethernet adapter is not supported, a single virtual Ethernet adapter Link Aggregation is possible because a single adapter configured as an EtherChannel in standard mode does not require switch support from the POWER Hypervisor.

The IP address of the client partition is configured on the network interface of the EtherChannel. If the primary adapter fails, the EtherChannel will automatically switch to the backup adapter. The IP address of the client server partition which is configured on the EtherChannel network interface will remain available.

Restriction: When using the EtherChannel with two adapters as in this example and configuring one adapter as backup, no aggregation resulting in higher bandwidth will be provided. No network traffic will go through the backup adapter unless there is failure of the primary adapter.

Also note that gratuitous ARP has to be supported by the network in order for adapter failover to work.

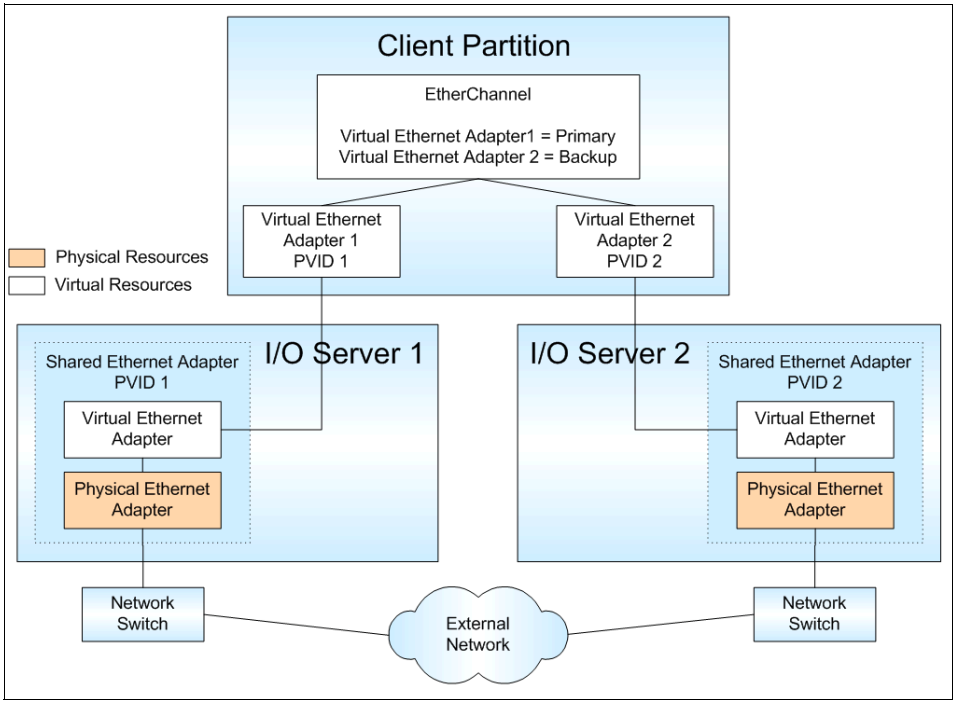

Figure 4-13 Virtual I/O server configuration with network interface backup

This configuration protects your network interface adapter against:

- Failure of one physical network adapter in one Virtual I/O Server
- Failure of one Virtual I/O Server
- Failure of one network switch (if adapters are connected to different switches as shown in this example)

The physical Ethernet adapters shown in Figure 4-13 are connected to the network switches on untagged ports. The Virtual I/O servers will strip VLAN tags from packets before delivering them to the switches. The network switches will see the MAC addresses on the virtual Ethernet adapters in the client partition, but will not see the VLAN tags. The Virtual I/O servers will propagate broadcast packets from the switches to the virtual Ethernet adapters in the client partition.

If a Virtual I/O server (or some network component) fails, the Ethernet network will see the client partition's IP address suddenly hop from one switch and MAC address to another. Such behavior will be handled acceptably if both of the following are true:

- The network supports Gratuitous ARP.

- The network switches are configured such that both ports (one on each switch) can contact the same set of hosts in the rest of the network.

It is recommended that the client partition's AIX be configured to detect network unreachability by specifying in the Network Interface Backup configuration an IP address (or host name) of a router to which connectivity should always be available.

For more details on configuring Link Aggregation (EtherChannel) see *AIX System Management Guide: Communications and Networks* available with the product documentation.

Multipath routing and dead gateway detection

Figure 4-14 on page 126 shows a configuration using multipath routing and dead gateway detection.

The client partition has two virtual Ethernet adapters. Each adapter is assigned to a different VLAN (using the PVID).

Each Virtual I/O Server is configured with a Shared Ethernet Adapter that bridges traffic between the virtual Ethernet and the external network. Each of the Shared Ethernet Adapters is assigned to a different VLAN (using PVID).

By using two VLANs, network traffic is separated so that each virtual Ethernet adapter in the client partition seems to be connected to a different Virtual I/O Server.

In the client partition, two default routes with dead gateway detection are defined: one route is going to gateway 9.3.5.10 using virtual Ethernet adapter with address 9.3.5.12; the second default route is going to gateway 9.3.5.20 using the virtual Ethernet adapter with address 9.3.5.22.

In case of a failure of the primary route, access to the external network is provided through the second route. AIX detects route failures and adjusts the cost of the route accordingly.

Restriction: It is important to note that multipath routing and dead gateway detection do not make an IP address highly available. In the case of failure of one path, dead gateway detection will route traffic through an alternate path. The network adapters and their IP addresses remain unchanged. Therefore, when using multipath routing and dead gateway detection only your access to the network will become redundant, but *not* the IP addresses.

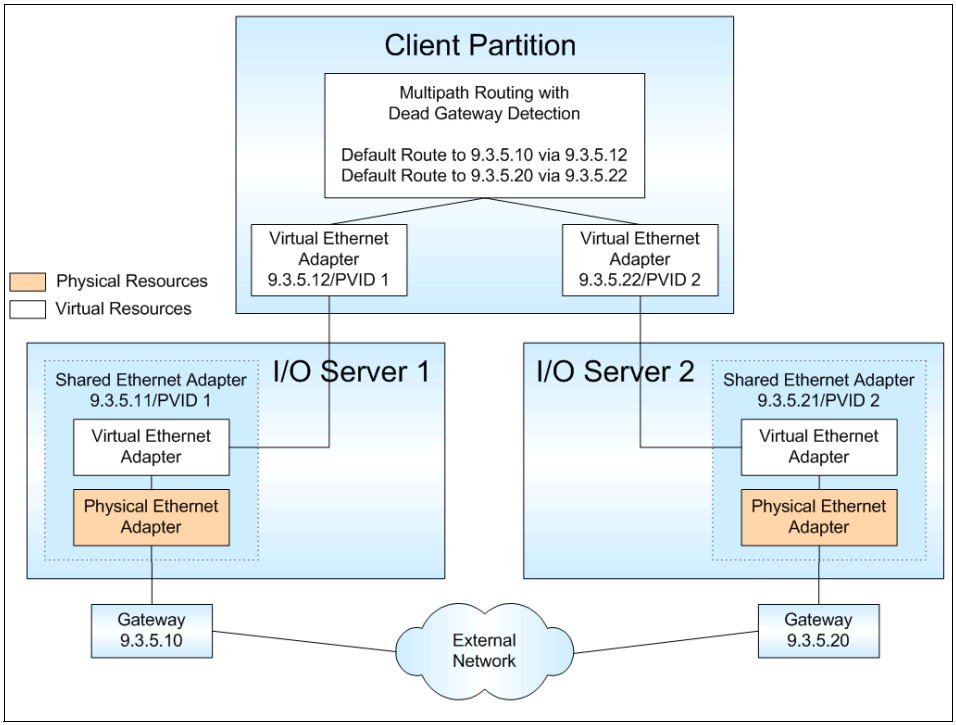

Figure 4-14 Configuration with multipath routing and dead gateway detection

This configuration protects your access to the external network against failure of:

- One physical network adapter in one Virtual I/O server
- One Virtual I/O Server
- One gateway

LVM mirroring

Figure 4-15 on page 127 shows a Virtual I/O Server configuration using LVM mirroring on the client partition.

The client partition is LVM mirroring its logical volumes using the two virtual SCSI client adapters. Each of these adapters is assigned to a separate Virtual I/O Server partition.

The two physical disks are each attached to a separate Virtual I/O Server partition and made available to the client partition through a virtual SCSI server adapter.

Restriction: At the time of writing, LVM mirroring using virtual SCSI only worked when the logical volume on the Virtual I/O Server was configured with the following settings:

- Mirror Write Consistency turned off
- Bad Block Relocation turned off
- No striping
- Logical volume must not span several physical volumes

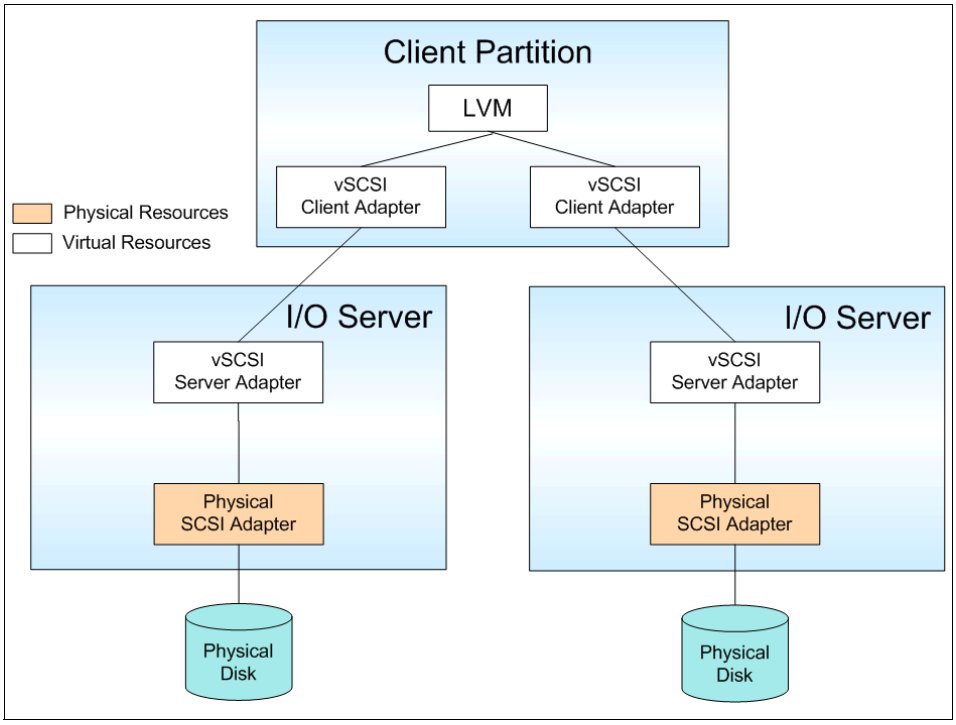

Figure 4-15 Virtual I/O server configuration with LVM mirroring

This configuration protects a virtual disk in a client partition against failure of:

- One physical disk
- One physical adapter
- One Virtual I/O Server

Multipath I/O

Figure 4-16 shows a configuration using Multipath I/O to access an ESS disk.

The client partition sees two paths to the physical disk through MPIO. Each path is using a different virtual SCSI adapter to access the disk. Each of these virtual SCSI adapters is backed by a separate Virtual I/O Server.

Note: This type of configuration will only work when the physical disk is assigned as a whole to the client partition. You cannot split up the physical disk into logical volumes at the Virtual I/O Server level.

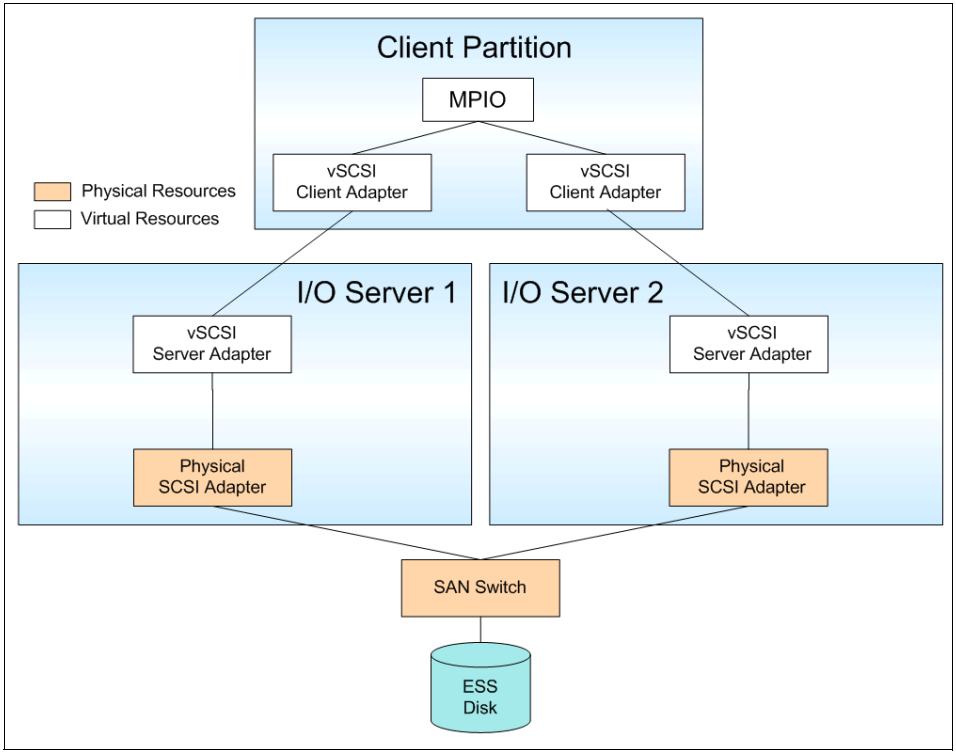

Figure 4-16 Virtual I/O server configuration with MPIO

This configuration protects a virtual disk in a client partition against failure of:

- One physical FC adapter in one Virtual I/O Server
- One Virtual I/O Server

Depending on your SAN topology each physical adapter could be connected to a separate SAN switch to provide redundancy. At the physical disk level the ESS provides redundancy because it uses RAID technology internally.

4.4 Virtual I/O Server maintenance and monitoring

The following sections describe maintenance and monitoring of the Virtual I/O Server.

Maintenance

The Virtual I/O Server should be regarded as an appliance running in a dedicated partition. Fixes or upgrades for the Virtual I/O Server will be grouped into special fix packs and distributed separately from AIX or other IBM operating systems.

The Virtual I/O Server should be backed up regularly. Section 6.1, “Backup and restore of the Virtual I/O Server” on page 202 covers backup and restore of the Virtual I/O Server in detail.

Monitoring

Virtual I/O error handling is designed in such a way that permanent error messages are only logged once. If the Virtual I/O Server cannot fulfil a request, it checks if the request came from a virtual adapter. If this is the case, the permanent error is logged on the Virtual I/O Server. The virtual adapter in the client partition will only show an informational error.

Depending on the configuration of the Virtual I/O Server, temporary errors will not necessarily be propagated to the client partition. For example, if you have physical disks attached to the Virtual I/O Server that are accessed through Multipath I/O, the client partition will not see an error if one of the paths reports a temporary error. Thus, you might miss information about events signalling the imminent failure of path.

Additionally, you might want to be notified of errors that are not directly related to I/O operations, such as the paging space in the Virtual I/O Server reaching its software or hardware capacity limits.

To make sure you get notified about all errors, the error log and other vital components of the Virtual I/O Server should be monitored. You can monitor the Virtual I/O Server by running Tivoli or installing another management software agent.

4.5 Security issues

In this section we consider two security aspects of the Virtual I/O Server:

- Virtual SCSI
- Shared and virtual Ethernet

Using Virtual SCSI means the Virtual I/O Server acts as a *storage box* to provide the data. Instead of SCSI or Fibre cable, the connection is done by the POWER Hypervisor. The Virtual SCSI device drivers of the I/O Server and the POWER Hypervisor ensure that only the owning partition has access to its data. Neither other partitions nor the I/O Server itself are able to make the client data visible. Only the control-information is going through the I/O Server, the data-information is copied directly from the PCI-adapter to the client's memory. For further details on Virtual SCSI refer to 3.6, "Virtual SCSI introduction" on page 87.

Similar to Virtual SCSI, the POWER Hypervisor also provides the connection between different partitions when using Virtual Ethernet. Inside the server the POWER Hypervisor acts as an Ethernet switch.

The connection to the external network is done by the Virtual I/O Server's shared Ethernet function. This I/O Server acts as a Layer 2 bridge to the physical adapters. The Virtual Ethernet implementation fulfills the IEEE 802.1Q standard that describes VLAN tagging.

This means that a VLAN ID tag is inserted into every Ethernet frame. The Ethernet switch restricts the frames to the ports which are authorized to receive frames with that VLAN ID. Every port of an Ethernet switch can be configured to be a member of several VLANs. Only the network adapters, both virtual and physical ones, which are connected to a port (virtual or physical) that belongs to the same VLAN can receive these frames. The implementation of this VLAN standard ensures that the partitions have no access to foreign data.

4.6 Interaction with AIX partitions

The following section describes how the Virtual I/O Server provides resources to the AIX partitions. These resources can be:

- Virtual SCSI resources
- Virtual Ethernet resources

4.6.1 Virtual SCSI resources

To enable the AIX partitions to interact with Virtual SCSI resources the following steps are necessary:

1. Define the SCSI server adapter on the I/O Server.

   This is done on the Hardware Management Console and creates a Virtual SCSI Server Adapter (for example vhost1) with a selectable slot number.

2. Define the SCSI client adapter on the AIX partition.

   This is also done on HMC and creates a Virtual SCSI Client Adapter (for example vscsi0) with a selectable slot number. When creating the Virtual SCSI Client Adapter you have to choose the desired I/O Server partition and the slot number of the Virtual SCSI Server Adapter defined during step 1.

3. Map the desired SCSI resources.

   On the I/O Server you have to map either a physical volume or a logical volume to the defined Virtual SCSI Server Adapter. This creates a Virtual Target Device (for example vtscsi2) that provides the connection between the I/O Server and the AIX partition through the POWER Hypervisor.

The mapped volume now appears on the AIX partition as an hdisk device. The Virtual SCSI Client Adapter takes care that these hdisk devices can be used like every other physically connected hdisk device. It can be used for boot, swap, mirror, or any other supported AIX feature. In Appendix B, “Supported SCSI commands” on page 245 there is a listing of supported commands of the SCSI standard. Figure 4-17 on page 132 shows the connection from a physical disk connected to the I/O Server to a file system on an AIX partition using a virtual disk.

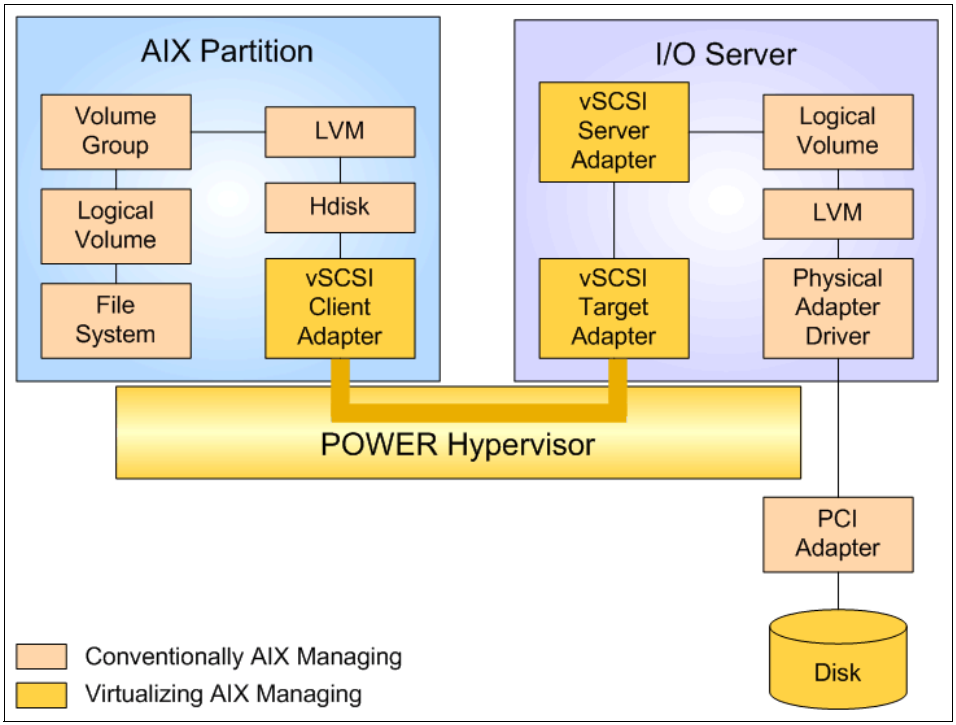

Figure 4-17 File system on virtual disk

4.6.2 Virtual Ethernet resources

The second type of resource that causes interaction between AIX partitions and the Virtual I/O Server is the Shared Ethernet Adapter. This feature allows the AIX partitions to connect to external networks without a physical adapter. The following steps enable this connectivity:

1. Define the Virtual Ethernet adapter on the I/O Server.

   This is to be done on the HMC.

2. Define the Virtual Ethernet adapters on the AIX partitions.

   This definition is done on the HMC and is not a Virtual I/O Server feature. It creates Virtual Ethernet adapters that can be used like any other Ethernet adapter. Different virtual networks can be separated using IEEE802.1Q-compatible VLAN features of the Virtual Ethernet adapters.

3. Map the I/O Server's Virtual Ethernet adapter to a physical one.

   Doing the mapping on the Virtual I/O Server will create a further Ethernet adapter that is called Shared Ethernet adapter. The I/O Server acts like a

bridge and forwards the IP packages using the Virtual Ethernet connections to the AIX partitions. Figure 4-18 shows the connection from AIX partitions to an external network, with the Virtual Ethernet adapters using a shared Ethernet adapter of the I/O server.

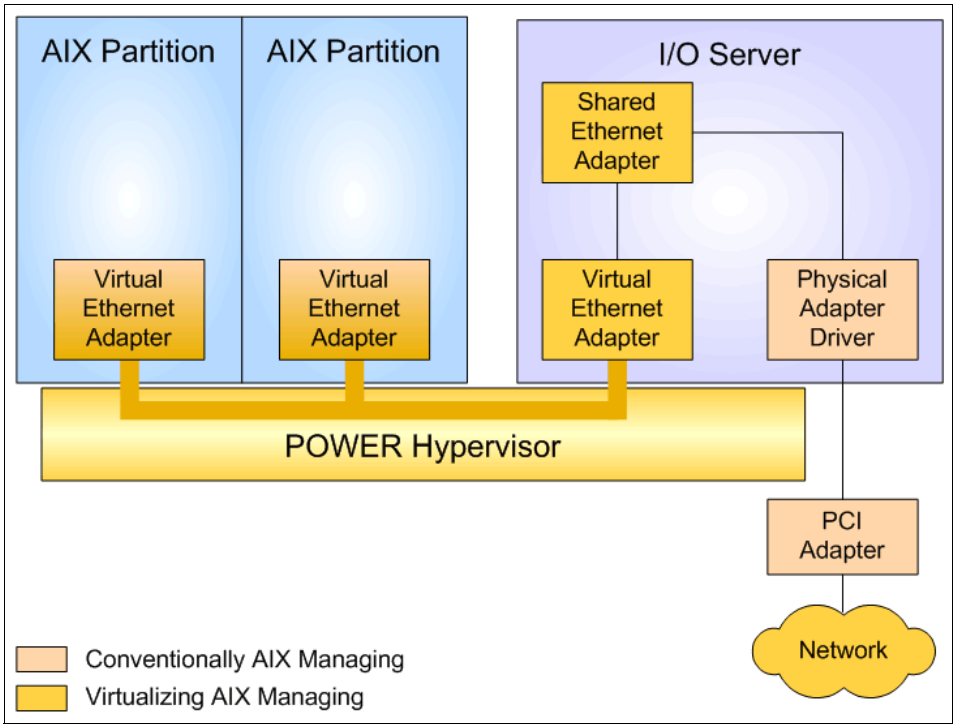

Figure 4-18 Virtual adapters connect to external network

The implementation of Virtual Ethernet adapters on an IBM ℯserver p5 system within AIX is assigned one IEEE VLAN-aware Virtual Ethernet switch in the system. All partitions talking on the Ethernet are peers. Up to 4,096 separate IEEE VLANs can be defined. Each partition can have up to 65,533 Virtual Ethernet adapters connected to the virtual switch. Each adapter can be connected to 21 IEEE VLANs (20 VID and 1 PVID).

The enablement and setup of a Virtual Ethernet does not require any special hardware or software. After a specific Virtual Ethernet is enabled for a partition, a network device named eth*XX* is created in the partition. The user can then set up TCP/IP configuration appropriately to communicate with other partitions. For information about network TCP/IP setup and configuration tools, see your AIX documentation.

4.7 Interaction with Linux partitions

The following sections describe the interaction and supported functions of Linux on IBM @server p5 systems. These resources can be:

- Virtual SCSI resources
- Virtual Ethernet resources

On IBM @server hardware systems, virtual adapters, including Virtual Ethernet and Virtual SCSI, interact with the operating system like any other adapter card, except that they are not physically present. The HMC is used to create virtual adapters in order to use virtual I/O devices. When Linux is operating with dynamic resource movement, adapters can be added while the system is running.

The server firmware running in Linux logical partitions recognizes Virtual I/O and can start the partition from Virtual I/O. Boot support can be from either network over Virtual Ethernet, or from a virtual disk.

Linux may also provide some of the hosting services of the I/O server, but a detailed discussion of that is outside the scope of this publication.

4.7.1 Virtual SCSI resources

Virtual I/O configuration is a combination of HMC and Virtual I/O Server functions, whereas adapter configurations are made in the HMC. They include:

1. Define the SCSI server adapter on the I/O Server.

   This is done on the HMC and creates a Virtual SCSI Server Adapter (for example vhost1) with a selectable slot number.

2. Define the SCSI client adapter on the Linux partition.

   This is also done on HMC and creates a Virtual SCSI Client Adapter with a selectable slot number. When creating the Virtual SCSI Client Adapter, you have to choose the desired I/O Server partition and the slot number of the Virtual SCSI Server Adapter defined during step 1.

3. Map the desired SCSI resources.

   On I/O Server you have to map either a physical volume or a logical volume to the defined Virtual SCSI Server Adapter. This creates a Virtual Target Device (for example vtscsi2) that provides the connection between the I/O Server and the Linux partition through the POWER Hypervisor.

For Linux partitions, virtual adapters are listed in the device tree. The device tree is aware of only Virtual SCSI adapters, not the devices under the adapter.

Disk units on an IBM @server p5 system are based on SCSI protocol using ANSI SCSI Remote DMA (Direct Memory Access) protocol. Therefore, Linux partitions can access data among themselves by an adapter that is directly attached to the memory of other partitions. Although the Redundant Arrays of Independent Disks (RAID) function is not available with virtual SCSI adapters, software RAID is provided by Linux.

4.7.2 Virtual Ethernet resources

Linux on an IBM @server p5 system can establish a TCP/IP connection through either a directly attached network interface or through a Virtual Ethernet interface.

1. Define Virtual Ethernet adapters on the Linux partitions.

   This definition is done on HMC and is not a Virtual I/O Server feature. It creates Virtual Ethernet adapters that can be used like any other Ethernet adapter. Different virtual networks can be separated using IEEE802.1Q-compatible VLAN features of the Virtual Ethernet adapters.

2. Map the I/O Server's Virtual Ethernet adapter to a physical adapter.

   Doing the mapping on the Virtual I/O Server will create a further Ethernet adapter that is called Shared Ethernet adapter. The I/O Server acts like a bridge and forwards the IP packages using the Virtual Ethernet connections to the Linux partitions.

The implementation of Virtual Ethernet adapters on an IBM @server p5 system within Linux is assigned one IEEE VLAN-aware Virtual Ethernet switch in the system. All partitions talking on the Ethernet are peers. Up to 4,096 separate IEEE VLANs can be defined. Each partition can have up to 65,533 virtual Ethernet adapters connected to the virtual switch. Each adapter can be connected to 21 IEEE VLANs (20 VID and 1 PVID).

The enablement and setup of a virtual Ethernet does not require any special hardware or software. After a specific Virtual Ethernet is enabled for a partition, a network device named ethXX is created in the partition. The user can then set up TCP/IP configuration appropriately to communicate with other partitions. For information about network TCP/IP setup and configuration tools, see your Linux distribution documentation.

4.7.3 Linux distributions

The IBM @server hardware systems require a Linux for POWER distribution. The Linux for POWER distribution refers to Linux distributions available from Linux distributors that run on POWER Technology-based systems. Linux distributors provide custom components that ease the installation and

maintenance of Linux systems. Before installing a distributor's version of Linux, verify that the kernel has been compiled for the IBM @server hardware. For current information about Linux distributions, refer to the Linux at IBM Web site.

4.8 Interaction with i5/OS partitions

At the time of this i5/OS partitions are not supported with the Virtual I/O Server on p5 systems.

5

AIX and Virtual I/O Server configuration scenarios

This chapter describes two basic configuration scenarios. The first is designed to assist you in basic configuration of the Virtual I/O Server and several partitions. The second is designed to help you understand high availability configuration that may assist you in providing redundancy in your production configuration. This chapter is dedicated to these scenarios:

- Scenario 1 - Basic configuration
  - Creating Virtual I/O Server partition
  - Creating client partitions
  - Virtual I/O Server software installation
  - Defining Virtual I/O resources for the Virtual I/O Server
  - Defining Virtual I/O resources for clients
  - Virtual I/O Server configuration
  - Client partition AIX installation
- Scenario 2 - High availability configuration

  Scenario 2 shows you how to perform the necessary steps using the dynamic LPAR features of POWER5. This means the changes can be made without shutting down the AIX operating systems. See Appendix A, “Worksheets for partition configuration planning” on page 237 for complete configuration details such as processor and memory settings, IP addresses, slot numbering, and so on.

5.1 Scenario 1 introduction

Scenario 1 provides a simple configuration of three partitions that receive their Ethernet and SCSI resources from one Virtual I/O Server as shown in Figure 5-1. Optionally, one of the partitions can own a dedicated physical Ethernet adapter, for example, to have a guaranteed throughput for backup issues.

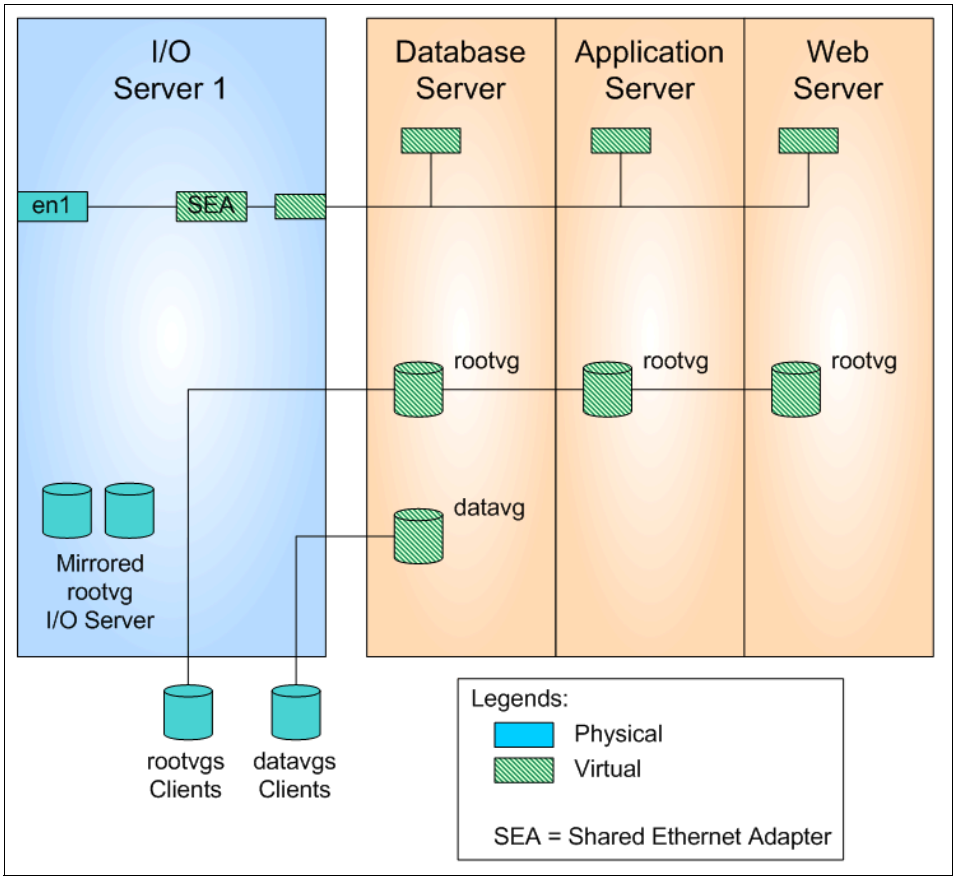

Figure 5-1 Configuration scenario 1

5.2 Scenario 2 introduction

Scenario 2 is an extension of the previous configuration and is designed to achieve higher availability. Therefore, there is a second Virtual I/O Server, which provides further virtual SCSI disks, to mirror the rootvg and datavg within the client partitions.

The availability of the network connection is increased in two steps:

1. The Virtual I/O Server's physical adapter is doubled. Then, with the creation of an EtherChannel, the configuration is protected against adapter failures.
2. The client partitions get a second virtual Ethernet adapter, provided by the second Virtual I/O Server. With these two virtual adapters an EtherChannel is created inside every client partition. In this configuration the aggregation of the two virtual adapters does not work because there is no common point (like an Ethernet switch) that can manage the aggregation. That's why only the backup adapter feature from the EtherChannel is used.

Figure 5-2 shows the scenario 2 configuration.

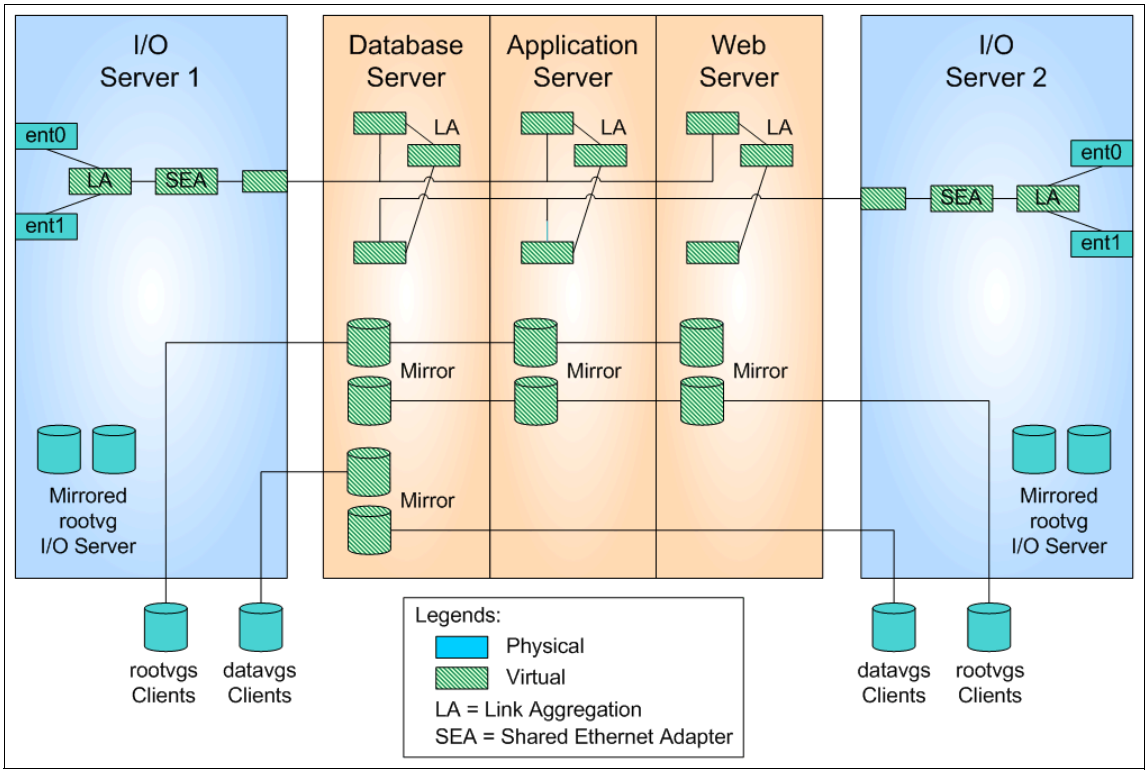

Figure 5-2 Configuration scenario 2

With scenario 2, the configuration is protected against failure of:

- One physical Ethernet adapter through the use of Link Aggregation (EtherChannel) of the Virtual I/O Server
- Loss of complete Ethernet connection of a single Virtual I/O Server through the use of Link Aggregation inside the client partitions

- Storage connecting to a single Virtual I/O Server by mirroring the volume groups inside the client partitions
- A critical Virtual I/O Server failure by mirroring and Link Aggregation as described previously

5.3 Scenario 1: Basic configuration

The following sections take you through the steps required to create your partitions, install the virtual I/O server, and configure partitions to use Virtual I/O Server resources.

5.3.1 Creating the Virtual I/O Server partition

This section shows you how to create the partition for the Virtual I/O Server for scenario 1 named I/O_Server_1 on the HMC.

It is recommended to dedicate a processor when optimal performance is a requirement; however, in this section we use a shared processor to make the best use of the resources on our test system. It also provides a useful example of creating a shared processor partition.

The following steps describe the configuration of processors, memory, and physical devices. The virtual devices are configured in a separate step.

Figure 5-3 shows the HMC of an unconfigured POWER5 system named p5Server with no partitions defined.

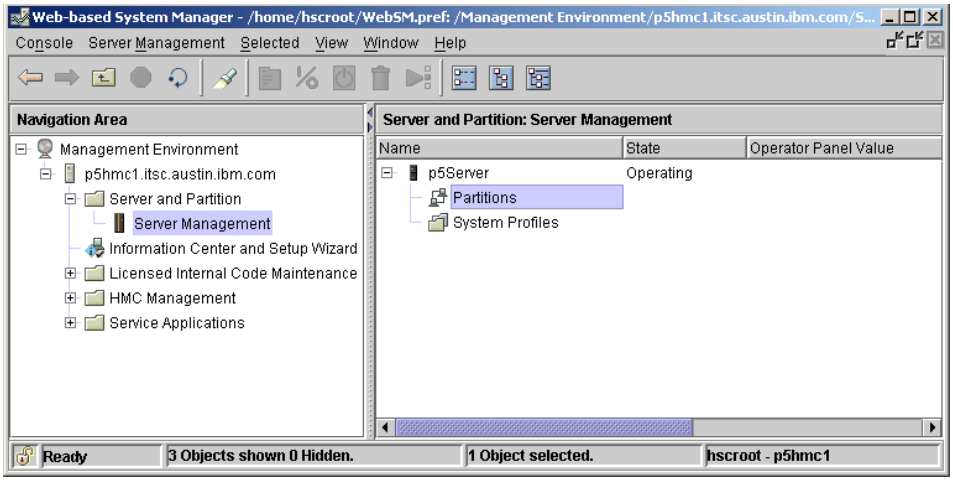

Figure 5-3 Unconfigured POWER5 system

1. Right click **Partitions**, then select **Create** → **Logical Partition** as shown in Figure 5-4 to start the Create Logical Partition Wizard.

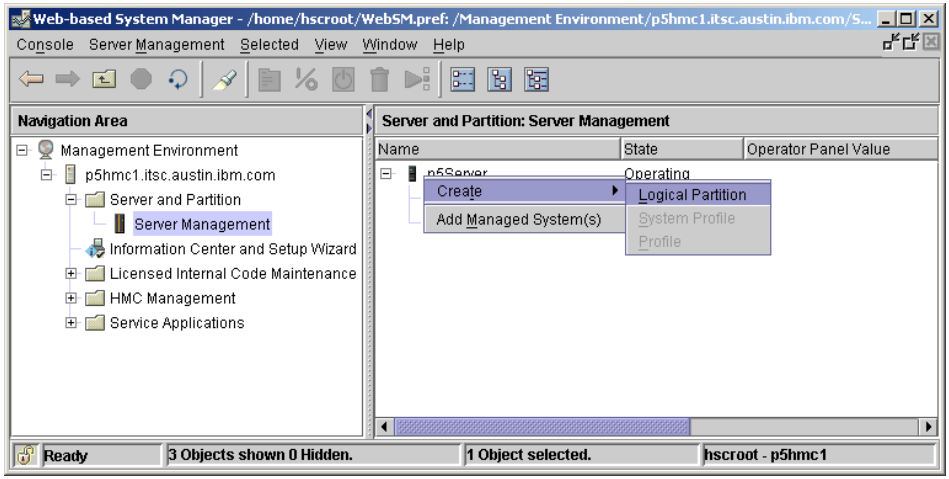

Figure 5-4 Starting Create Logical Partition Wizard

2. Enter the partition name and the partition ID, and select the **Virtual I/O server** checkbox, as shown in Figure 5-5. During testing, it was discovered that the Virtual I/O Server could be installed in an AIX or Linux partition. If this is the case with later versions of HMC software, then make sure this does not happen. If you select AIX and Linux for the Virtual I/O Sever, you will only be able to define client virtual adapters.

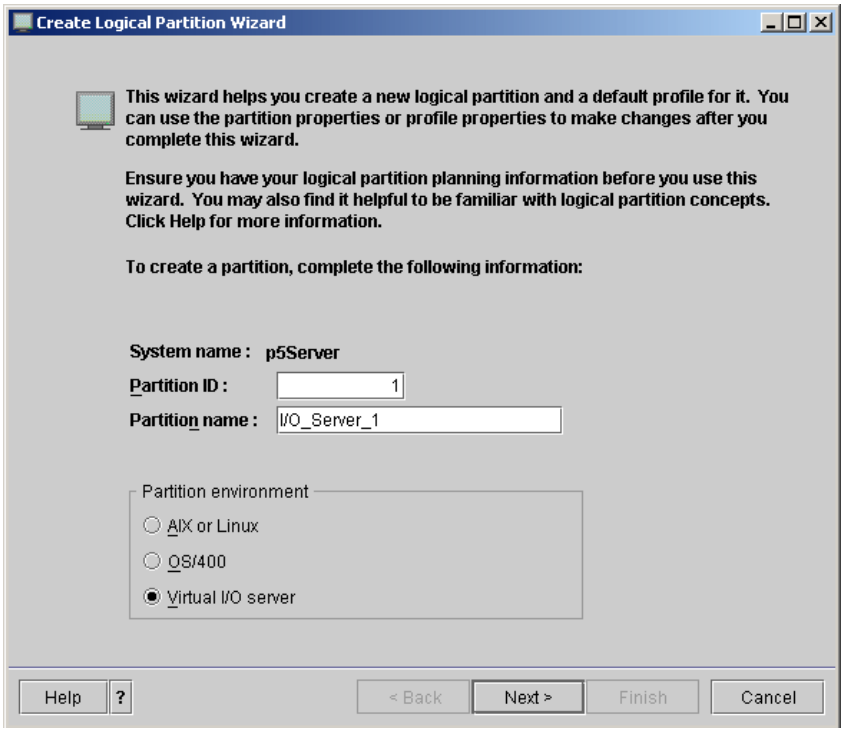

Figure 5-5 Defining partition name and ID

3. Skip the definition of a workload management group by selecting the **No** checkbox, as shown in Figure 5-6.

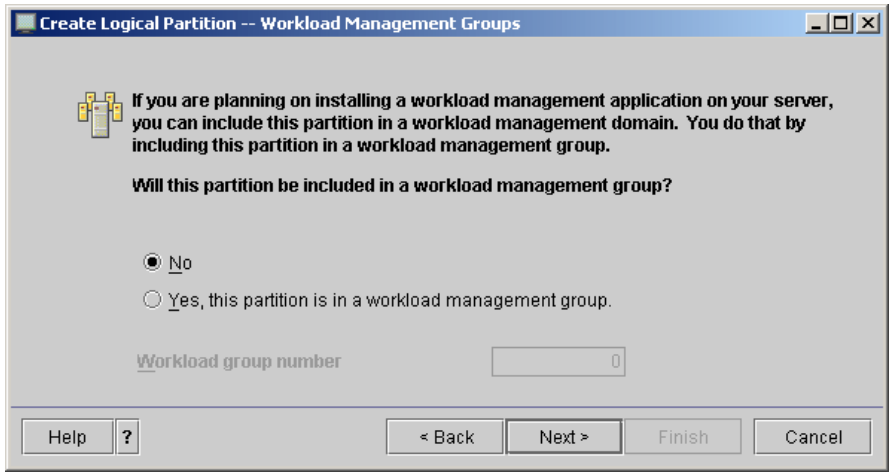

Figure 5-6 Skipping workload management group

4. Use `default` as the name for the partition's profile, as shown in Figure 5-7.

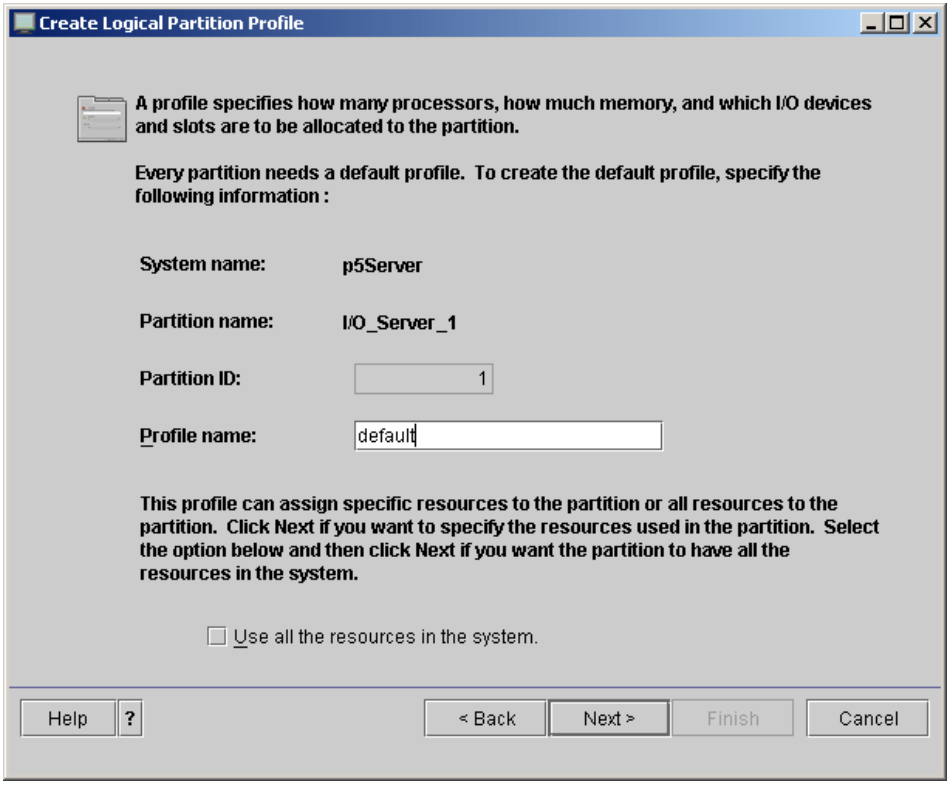

Figure 5-7 Naming the partitions profile

Note: If the checkbox "Use all the resources in the system" is activated this partition acts like a full partition POWER4 system, but compared to the POWER4 full partition mode, on POWER5 the POWER Hypervisior will always be active.

5. Choose the memory settings, as shown in Figure 5-8.

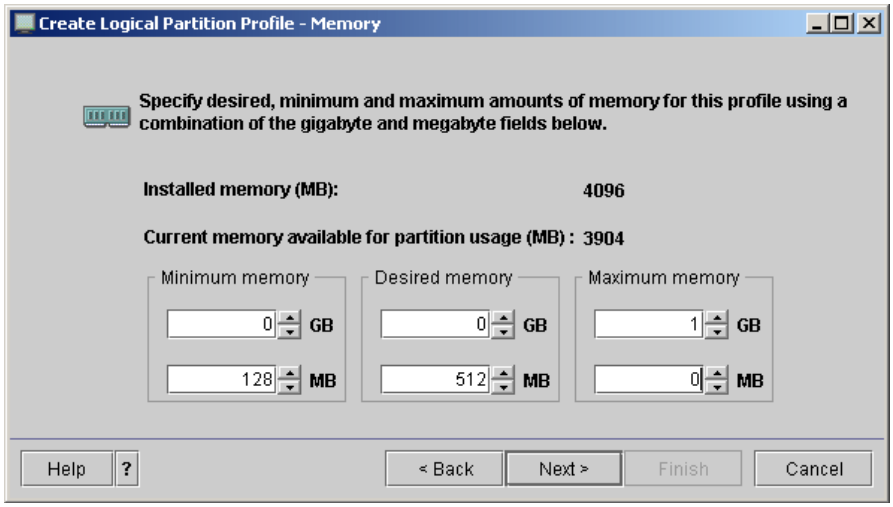

Figure 5-8 Partitions memory settings

Restriction: The following are key restrictions used during this step:

- If the system cannot provide the defined minimum amount of memory, the partition will not start.
- You cannot increase dynamically the amount of memory to more than the defined maximum amount.

6. Select the **Shared** checkbox for processor allocation, as shown in Figure 5-9.

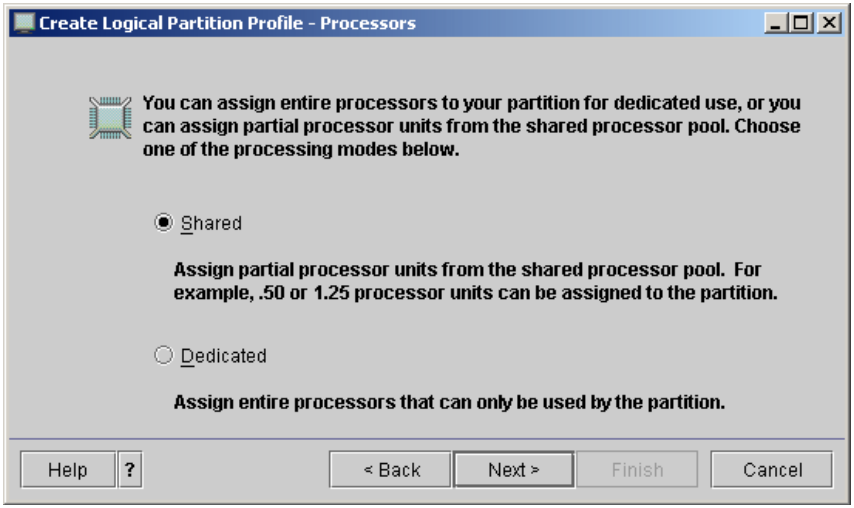

Figure 5-9 Using shared processor allocation

Note: Although it is recommended to use at least one dedicated processor for Virtual I/O Server partition, we used shared processor allocation due to the limited amount of processors (two) in our hardware environment.

7. Choose the shared processor settings, as shown inFigure 5-10.

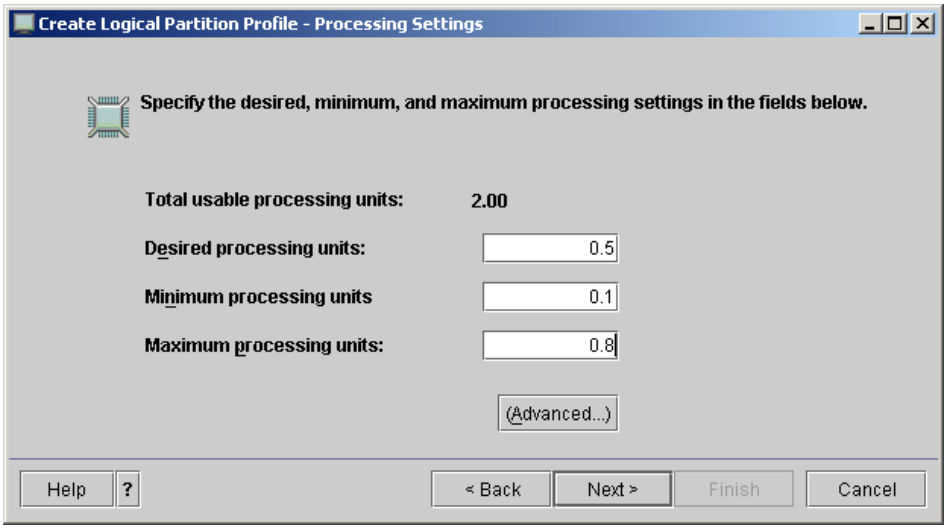

Figure 5-10 Shared processor settings

Restriction: The following restrictions apply to the processor settings:

- The partition will not start if the system cannot provide the defined minimum amount of processors.
- You cannot increase dynamically the amount of processors to more than the defined maximum amount allowed, unless CUoD features are available.

8. Specify the processing sharing mode and the virtual processor settings as shown in Figure 5-11 by clicking the **(Advanced...)** button. Click **Next** when you have completed the settings.

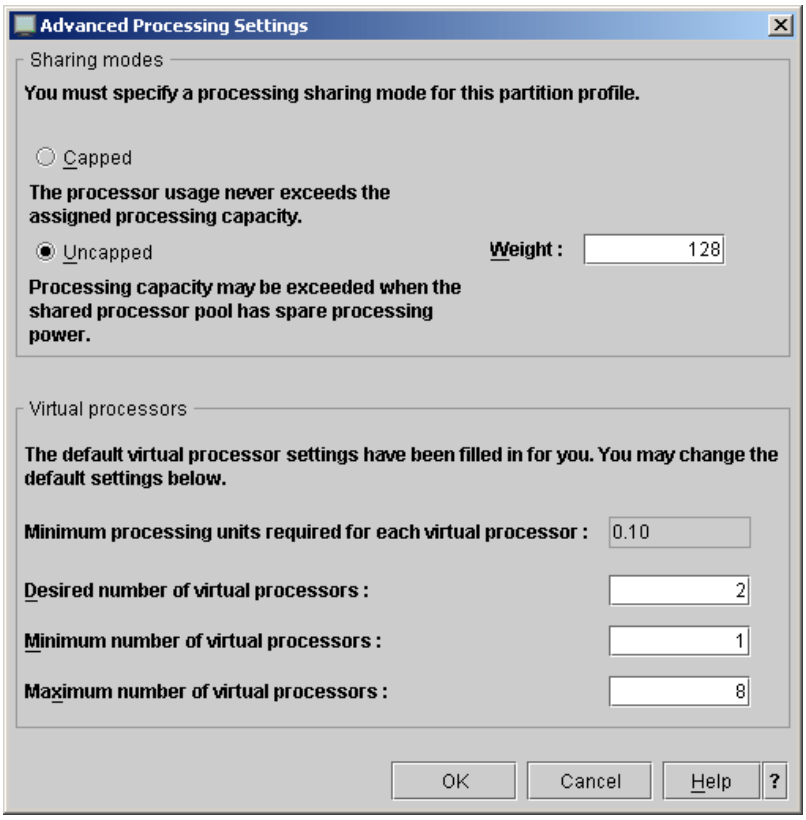

Figure 5-11 Processing sharing mode and the virtual processor settings

Note: Refer to 3.3, “Micro-Partitioning introduction” on page 54 for more information about processing units, capped and uncapped mode, as well as virtual processors.

9. Select at least one storage controller with local disks, one CD drive, and either one two port or two single port Ethernet adapters from your physical I/O components. Figure 5-12 shows the selections from our environment. Choose the matching Required settings and click **Next** when you have completed all of your selections.

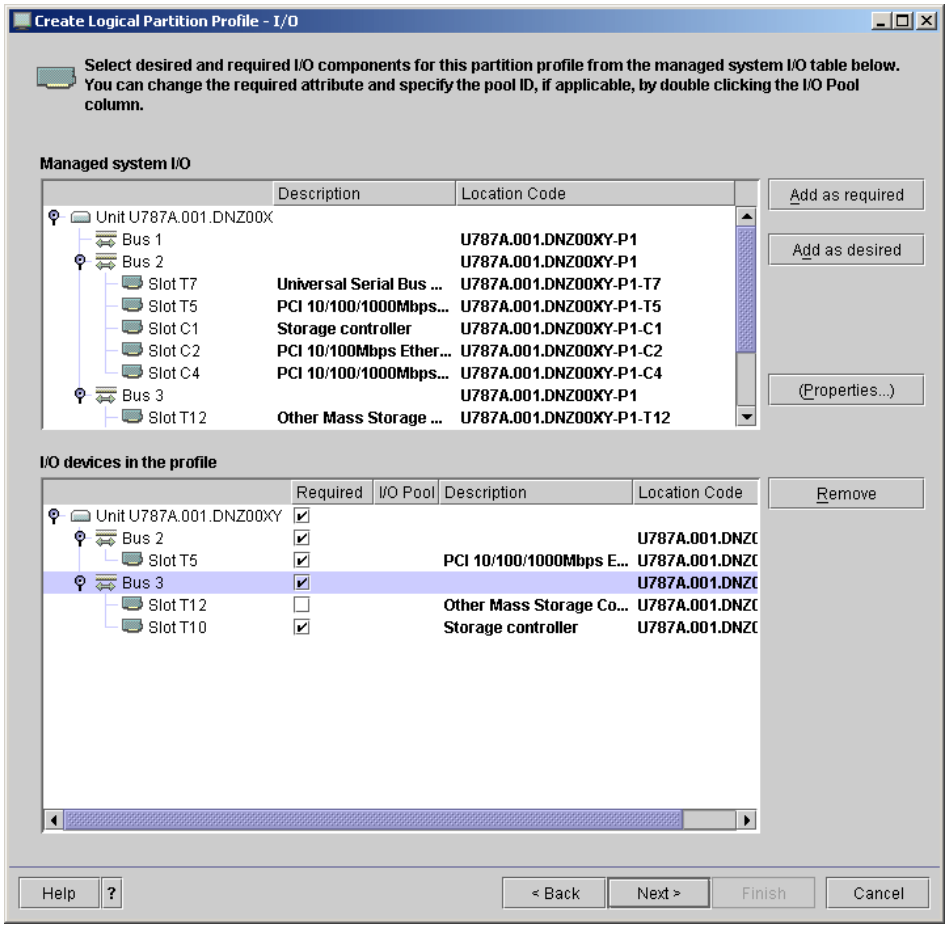

Figure 5-12 Physical I/O component selection

10. Skip the settings of I/O Pools as shown in Figure 5-13 clicking **Next**.

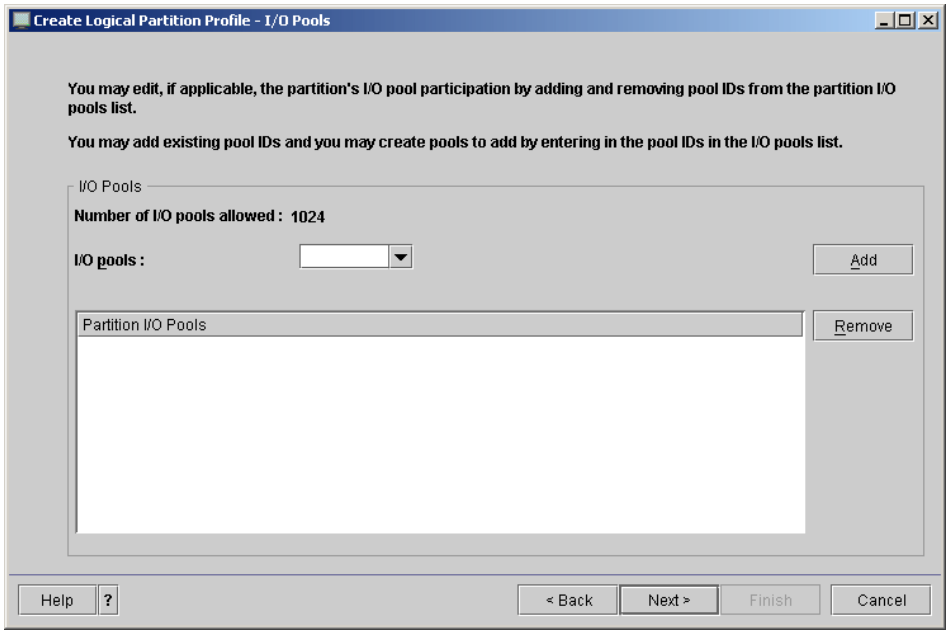

Figure 5-13 I/O Pool settings

11. Skip specifying virtual I/O adapters by selecting the **No** checkbox. As mentioned previously, this will be done in a later section.

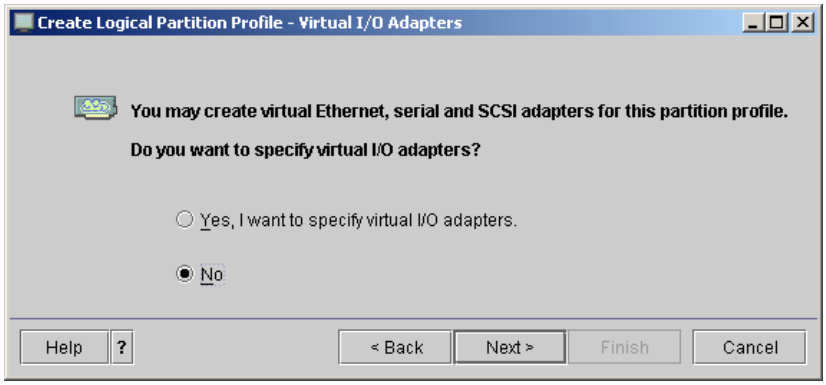

Figure 5-14 Skipping virtual I/O adapter definitions

12. Skip the settings for power controlling partitions shown in Figure 5-15 by clicking **Next**.

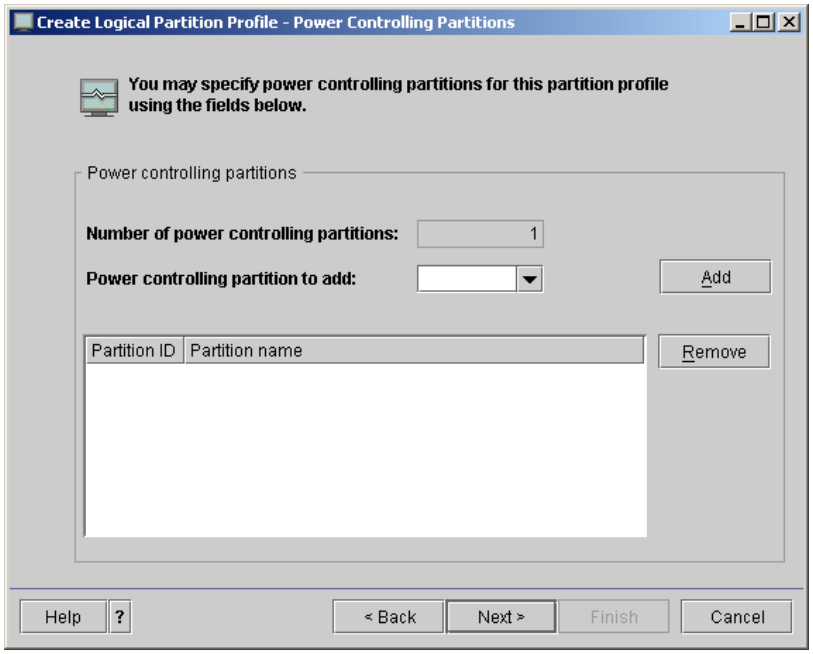

Figure 5-15 Skipping settings for power controlling partitions

13. Select **Normal** for your boot mode setting, as shown in Figure 5-16.

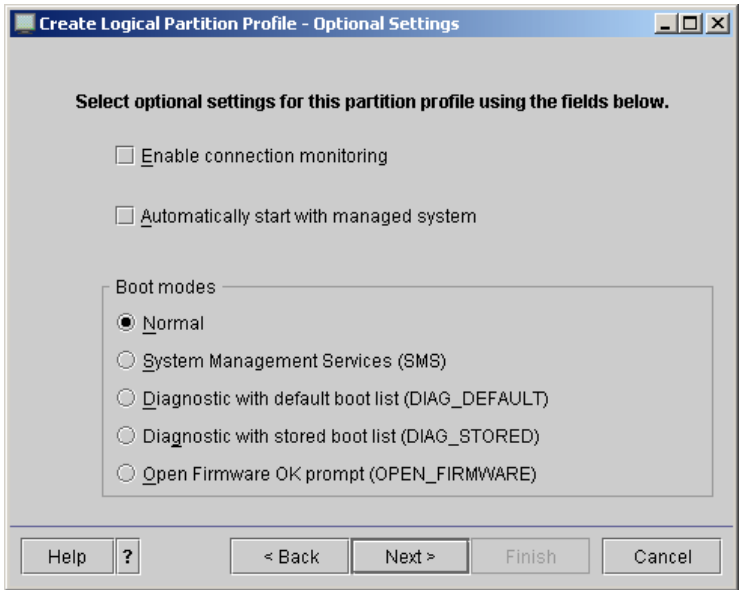

Figure 5-16 Boot mode settings

14. After checking the settings as shown in Figure 5-17, launch the partition creating process by clicking **Finish**.

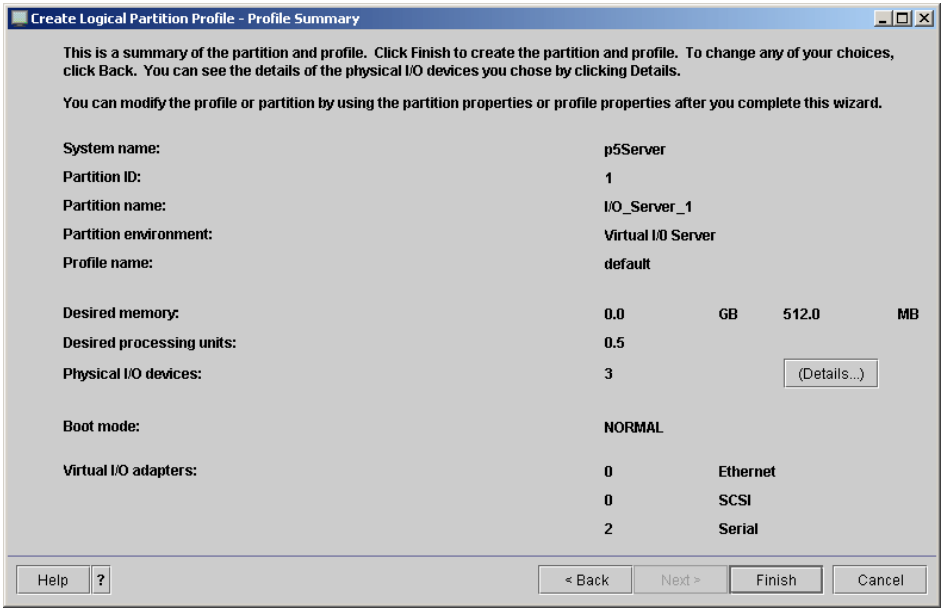

Figure 5-17 Overview of the partition settings.

The Working window shown in Figure 5-18 is displayed during the partition creation process.

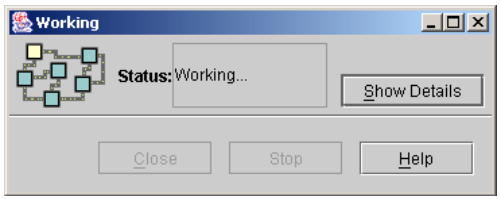

Figure 5-18 Working window

After processing, the new partition appears within the **Server Management** window, as shown in Figure 5-19.

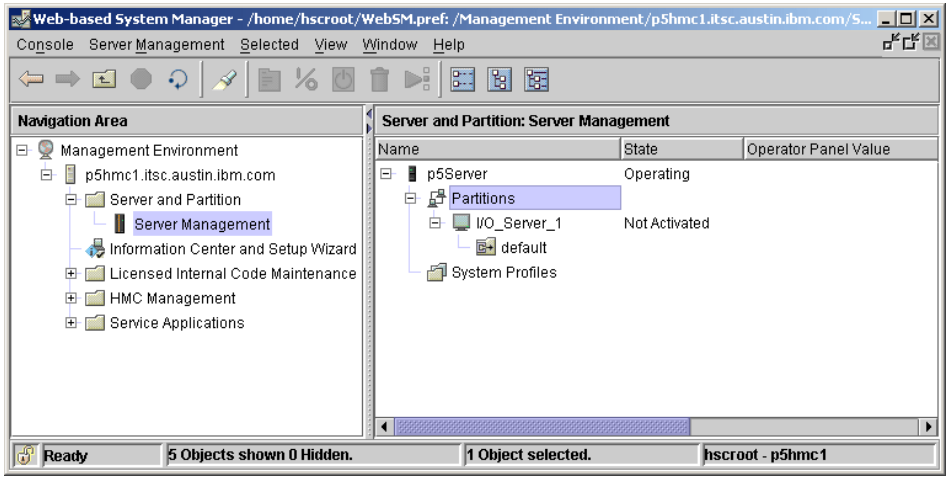

Figure 5-19 Managed POWER5 system with new I/O_Server_1 partition

The I/O_Server_1 partition is now created with its basic settings.

5.3.2 Creating client partitions

This section shows you how to create the three client partitions for scenario 1 on the HMC: DB_Server, Apps_Server, and Web_Server. Similar to the creation of the I/O_Server_1 partition, only the configuration of processors and memory is done at this time. The virtual devices are configured in a separate step.

1. Restart the Create Logical Partition Wizard, as you did at the beginning of the previous process (5.3.1, “Creating the Virtual I/O Server partition” on page 140).

2. Select the checkbox **AIX or Linux**, and enter a partition ID and partition name as shown in Figure 5-20.

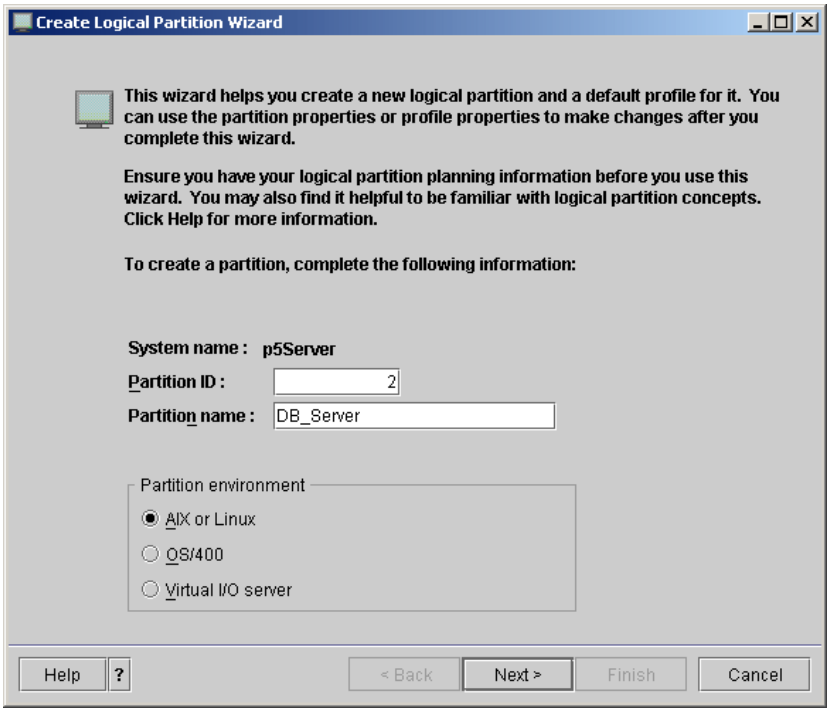

Figure 5-20 Creating DB_Server partition

3. Repeat steps 3 to 14 of the previous procedure, with the following exceptions:
   a. Skip step 9 by clicking **Next** instead of making a physical I/O component selection.
   b. Use 0.3/0.1/0.5 for desired/minimum/maximum processing units in step 7.
   c. Use 2/1/5 for desired/minimum/maximum number of virtual processors in step 8.

4. Figure 5-21 shows the result when you have finished creating the DB_Server partition.

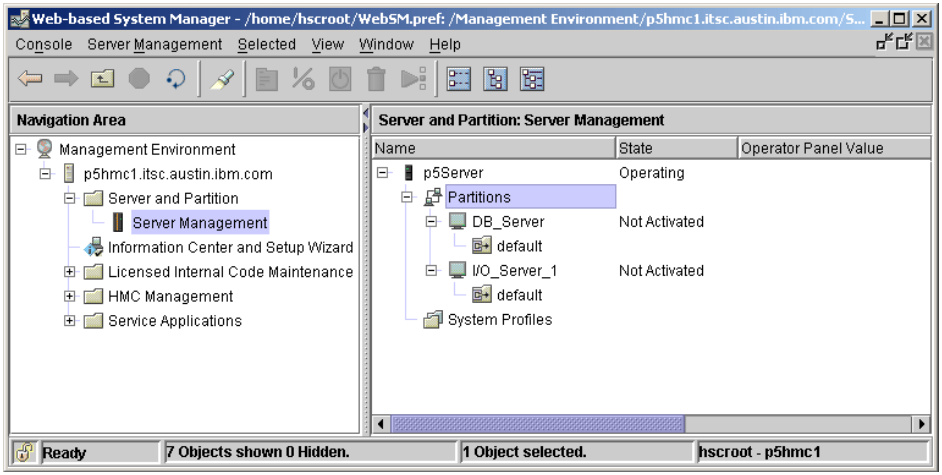

Figure 5-21 Managed system with DB_Server partition

5. Create the Apps_Server and Web_Server partition in the same way as the DB_Server partition.

   See Figure 5-22 and Figure 5-23 for the results of creating all client partitions.

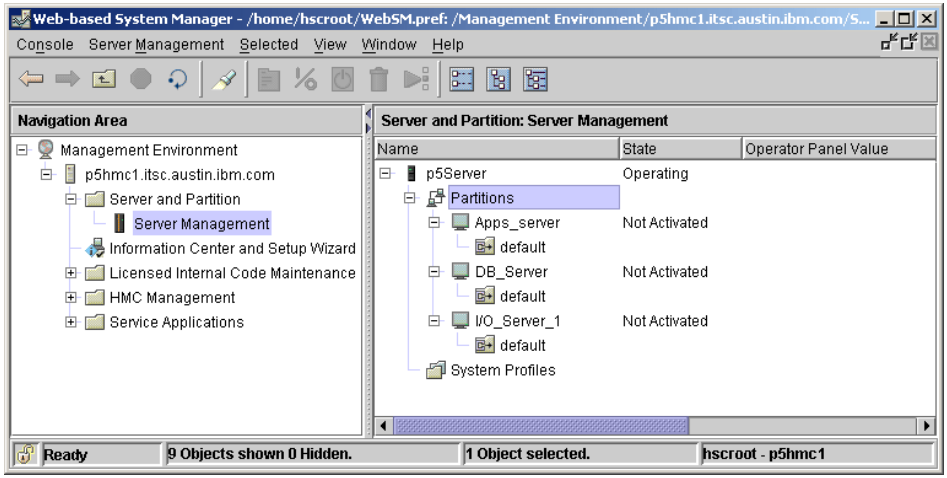

Figure 5-22 Managed system with Apps_Server partition

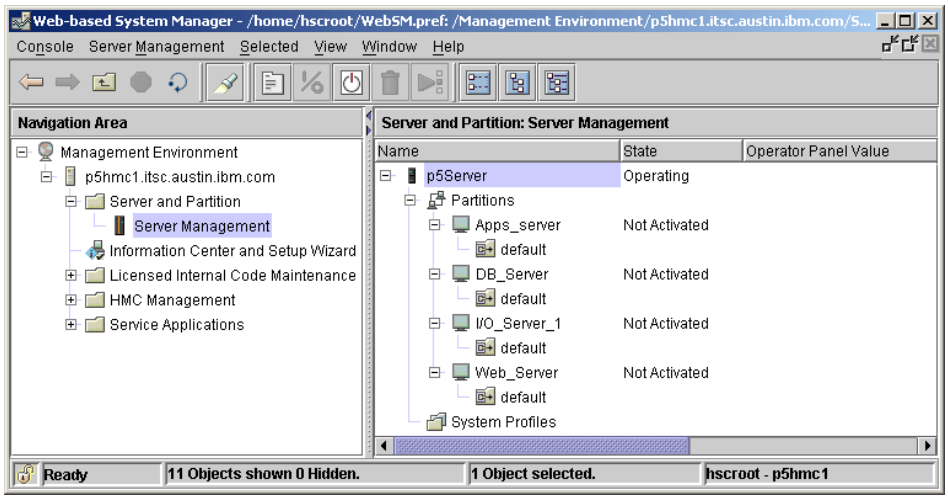

Figure 5-23 Managed system with all partitions of scenario 1

Now the basic creation of all partitions is done. The next section discusses the installation of the Virtual I/O Server.

5.3.3 Virtual I/O Server software installation

This section describes the installation of the Virtual I/O Server software to the formerly created partition I/O_Server_1. We assume that you are familiar with the tasks of a basic AIX installation. As shown in Figure 5-12 on page 149, the I/O_Server_1 partition owns the CD device since you selected **Other Mass Storage Controller** as a physical I/O component.

The following steps show the installation using the CD install device.

1. Activate the I/O_Server_1 partition by right-clicking the partition name and selecting the **Activate** bar as shown in Figure 5-24. Select the default profile you used to create this server.

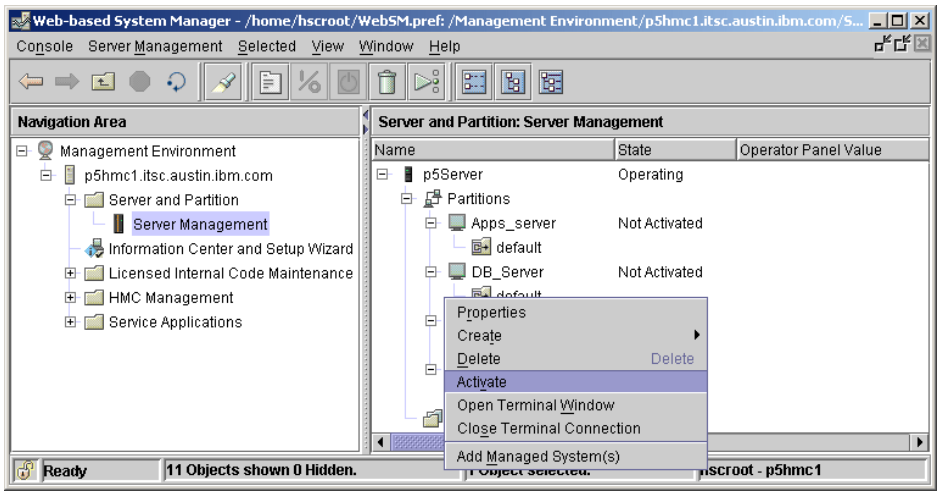

Figure 5-24 Activate I/O_Server_1 partition

2. Select the default profile, activate the **Open a terminal window or console session** checkbox, and click **(Advanced...)**, as shown in Figure 5-25.

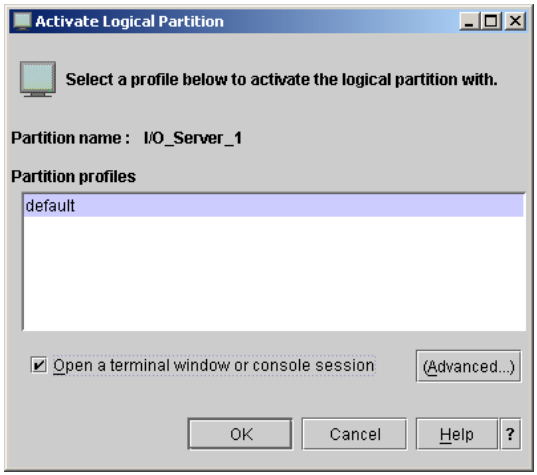

Figure 5-25 Selecting the profile

3. Choose the SMS boot mode as shown in Figure 5-26, and click **OK** to return to the previous window. When at the previous window, click **OK** to activate the partition and launch a terminal window.

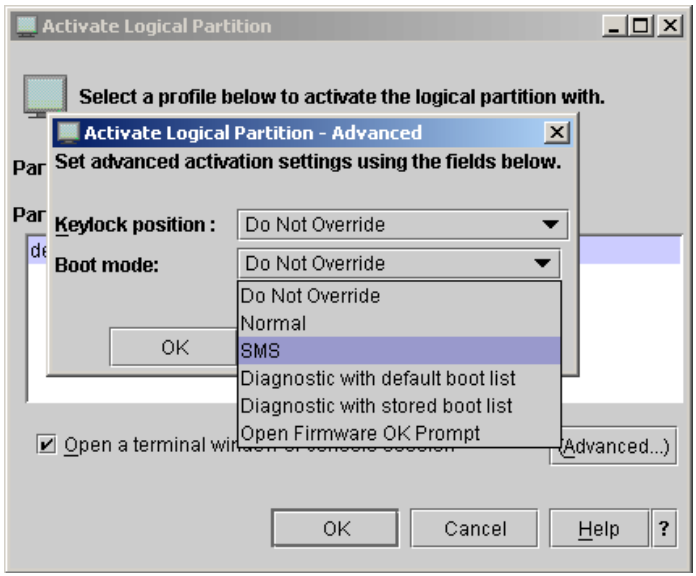

Figure 5-26 Choosing SMS boot mode

4. Figure 5-27 shows you an IBM @server pSeries SMS menu. Proceed through the installation process like any other AIX installation, choosing CD as your installation device.

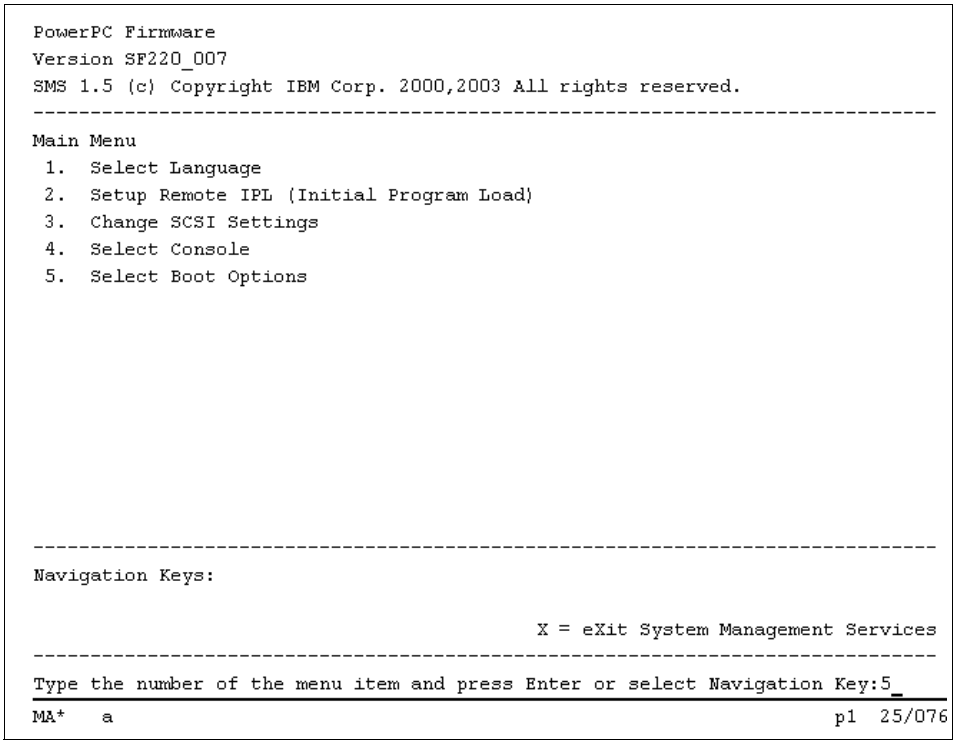

Figure 5-27 SMS menu

5. When the installation procedure has finished, use `padmin` for your username on the login prompt, choose a new password, and accept the license using the **`licence`** command at the **$** prompt, as shown in Figure 5-28. Now, for example, you can use the **`lspv`** command to show the available disks.

```
0513-059 The aixmibd Subsystem has been started. Subsystem PID is 655468.
0513-059 The muxatmd Subsystem has been started. Subsystem PID is 659536.
Finished starting tcpip daemons.
Starting NFS services:
0513-059 The biod Subsystem has been started. Subsystem PID is 622644.
0513-059 The rpc.lockd Subsystem has been started. Subsystem PID is 663688.
Completed NFS services.
Virtual I/O Server
login: 0513-059 The ctrmc Subsystem has been started. Subsystem PID is 356530.

Virtual I/O Server
login: padmin
[compat]: 3004-610 You are required to change your password.
        Please choose a new one.

padmin's New password:
Enter the new password again:

$ license -accept
$ lspv
hdisk0          00cddedc00efe72a                    rootvg          active
hdisk1          00cddedc15d81aaf                    None
hdisk2          00cddedc003416f4                    None
hdisk3          00cddedc04a9aa04                    None
$ _
MA*    a                                                              p1  25/003
```

Figure 5-28 Finished Virtual I/O Server installation

With this step you have finished the installation of the Virtual I/O Server. The I/O_Server_1 partition is now ready for further configuration.

5.3.4 Defining resources for the Virtual I/O Server

The following sections describe how to set up the virtual resources needed for Scenario 1, such as disks and Ethernet adapters on the Virtual I/O Server. This process shows, step by step, the resource configuration process based on the installation of the Virtual I/O Server described in the previous section.

Creating virtual Ethernet adapter for Virtual I/O Server

A virtual Ethernet adapter is a logical adapter description that emulates the function of a physical I/O adapter on a logical partition. Virtual I/O adapters

enable connections to other logical partitions on the same managed system without using real hardware and cables.

The first step is to create the virtual Ethernet adapter for the Virtual I/O Server (I/O_Server_1). To create a virtual Ethernet, perform the following steps at the HMC:

1. Right-click the profile name of the partition (I/O_Server_1) and go to the **Properties** menu as shown on Figure 5-29.

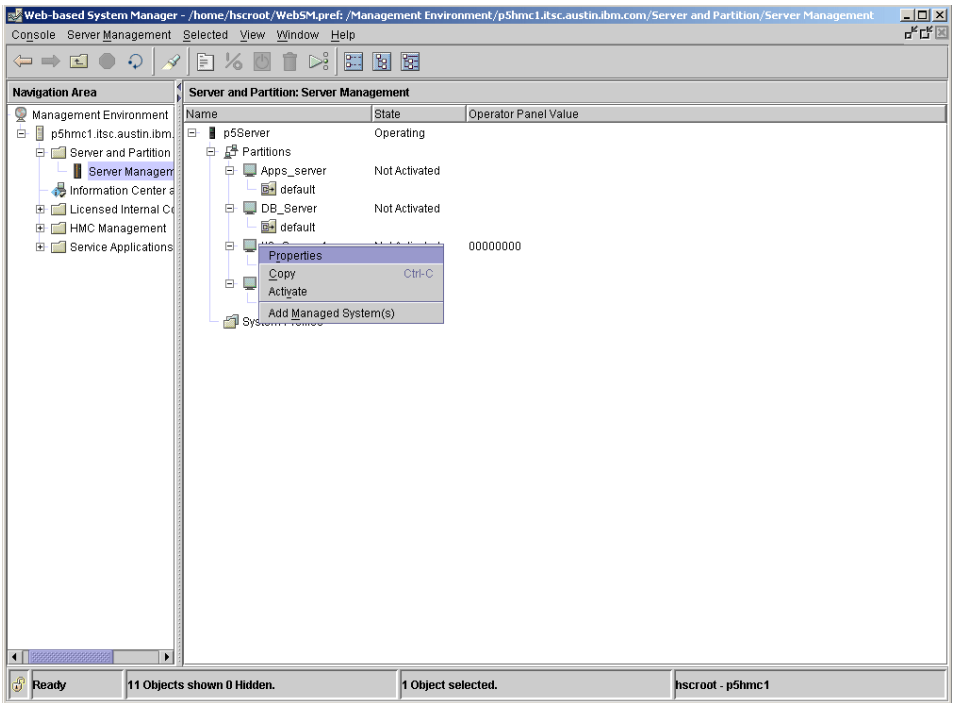

Figure 5-29 Profile properties menu

2. Click the Virtual I/O tab and add a virtual Ethernet adapter by choosing the **Ethernet** radio button in the Create Adapters area and clicking **Create**.
3. On the Virtual Ethernet Adapter Properties tab, choose the slot number for the virtual adapter and virtual LAN ID, then select the **Trunk Adapter** check box to use this adapter as a gateway between VLANs and an external network. This Ethernet adapter will be configured as a shared Ethernet adapter.

4. Select the **IEEE 802.1Q compatible adapter** check box and add the additional virtual LAN IDs for other partitions you want to communicate with using an external network, as shown on Figure 5-30.

Note: Selecting the “Trunk Adapter” flag makes sense only for a Virtual I/O Server partition. Do not select this flag when configuring the client partition. Do not create more than one Ethernet adapter with trunk flag on within one VLAN.

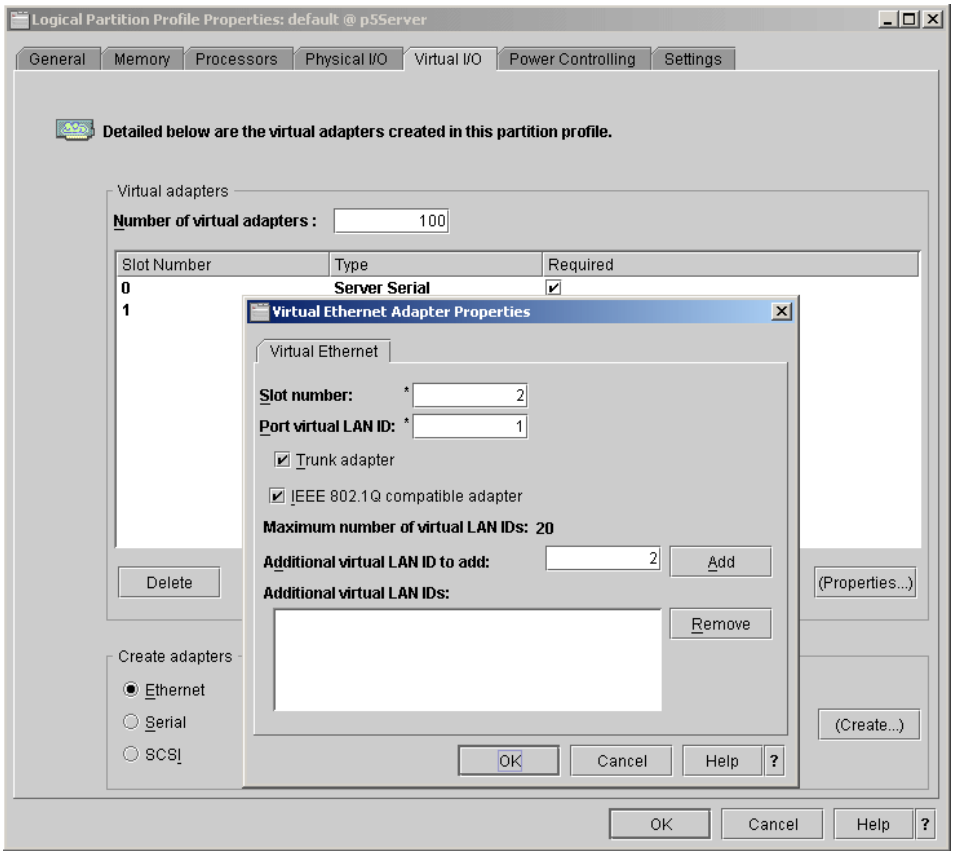

Figure 5-30 Virtual Ethernet Adapter Properties tab

5. Click **OK** and the virtual Ethernet adapter is ready for configuration from the Command Line Interface (CLI) of the Virtual I/O Server.

Creating Virtual SCSI server adapters

This section describes how to define the virtual SCSI server adapter for client partitions (DB_Server, Apps_Server, and Web_Server) using the Hardware Management Console.

Use the following steps to create virtual SCSI disks for the client partitions, then repeat this procedure for each virtual disk you want to create for your partitions.

In this scenario, you have to modify the Virtual I/O Server partition profile in the following way:

1. Right click the I/O_Server_1 partition's profile.
2. Choose the **Properties** menu.
3. On the Virtual I/O tab select the **SCSI** radio button and click **Create** as shown on Figure 5-31.

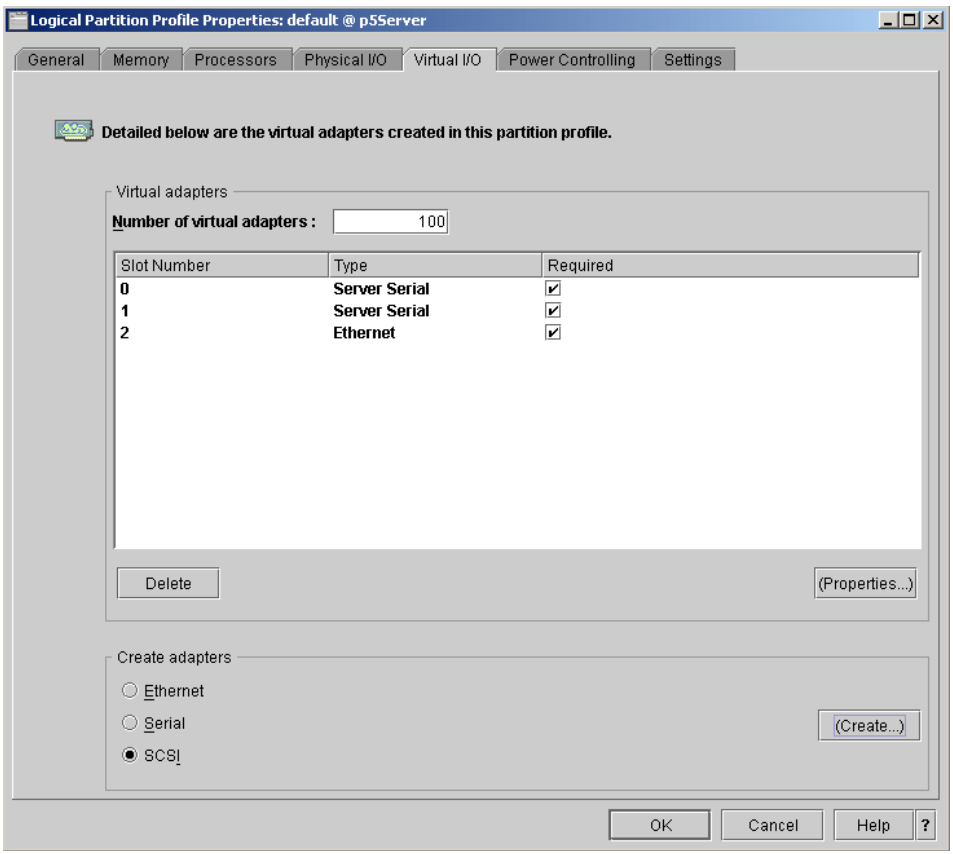

Figure 5-31 Virtual SCSI adapter properties

4. Define the following required information as shown on Figure 5-32:
   - Specify the slot number used for this virtual SCSI server adapter.
   - Choose the **Server** radio button in the Adapter Type field.
   - Choose the Remote partition name (**DB_Server**) from the drop-down list.

   Click **OK** to accept these changes.

Note: If all partitions are not defined yet, you can enter a unique Partition ID in the Remote partition field. Use this Partition ID while creating the client partition. HMC will automatically assign this partition as a client for virtual SCSI resources.

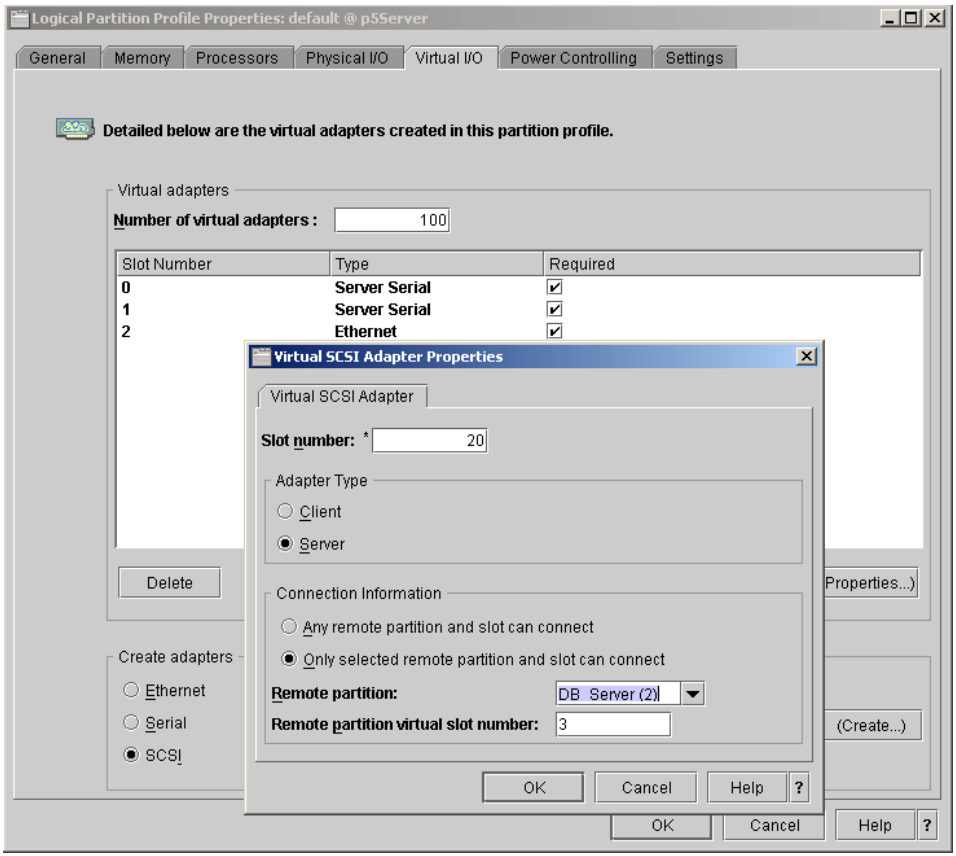

Figure 5-32 Virtual I/O properties

5. Repeat these steps for the other partitions (Web_server and Apps_Server) with slot numbers 30 and 40.

When all of the steps are finished for all of your partitions, you should see virtual SCSI resources defined in the Virtual I/O tab as shown on Figure 5-33.

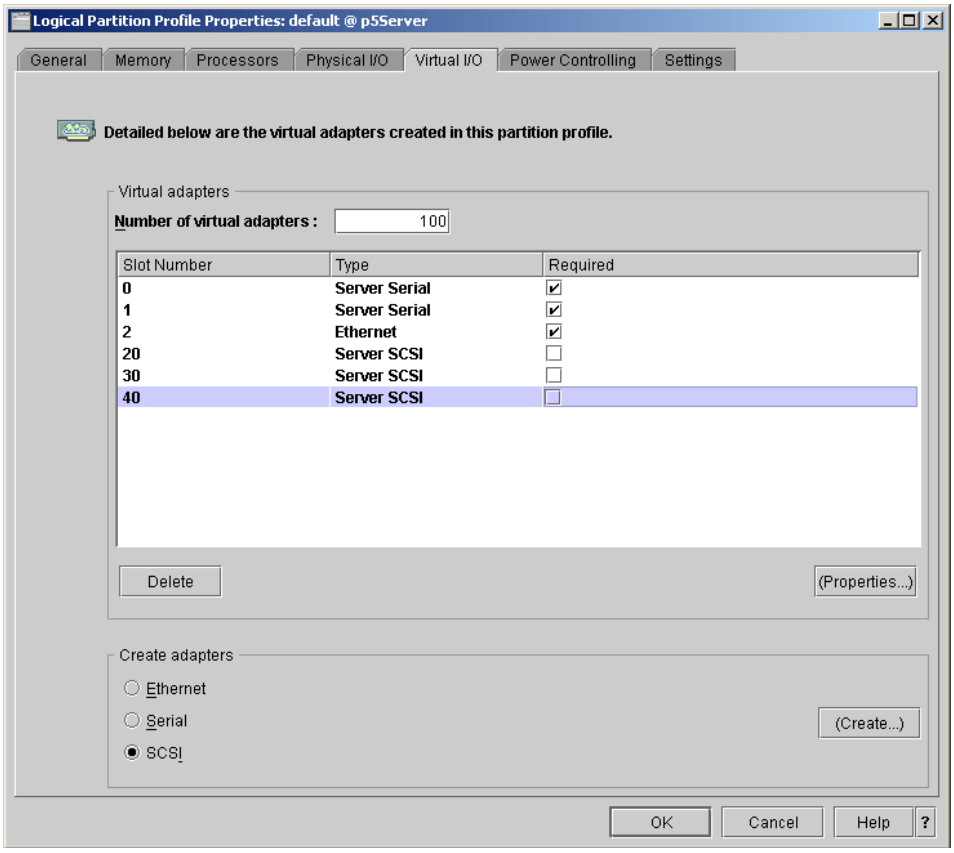

Figure 5-33 Defined virtual SCSI resource.

Note: If you want to dynamically move your resources between partitions, or dynamically remove your resources, do not mark the **Required** check box when defining them.

5.3.5 Defining Virtual I/O resources for the clients

This section discusses the creation of Virtual I/O resources for client partitions using the HMC.

Defining the virtual Ethernet adapters for the client partitions

A virtual Ethernet adapter is an adapter description that emulates the function of a physical I/O adapter on a logical partition. Virtual I/O adapters allow partitions to connect to other logical partitions on the same managed system without using real hardware and cables.

The following steps describe how to create the virtual Ethernet adapter for each of your virtual client partitions (DB_Server, Apps_Server, and Web_Server) using the HMC:

These steps are exactly the same steps as creating virtual Ethernet adapter for the I/O_Server_1, but *do not* select the **Trunk** flag.

1. Right-click the profile name of the client partition (DB_Server) and go to the Properties menu.
2. Add a virtual Ethernet adapter by selecting the Virtual I/O tab, choosing the **Ethernet** radio button in the Create Adapters field, and clicking **Create**.
3. On the Virtual Ethernet Adapter Properties tab choose the slot number for the virtual adapter and virtual LAN ID; do not select the Trunk Adapter.

4. Select the **IEEE 802.1Q compatible adapter** check box. Add the additional virtual LAN IDs for the other partitions that communicate with external networks, as shown on Figure 5-34.

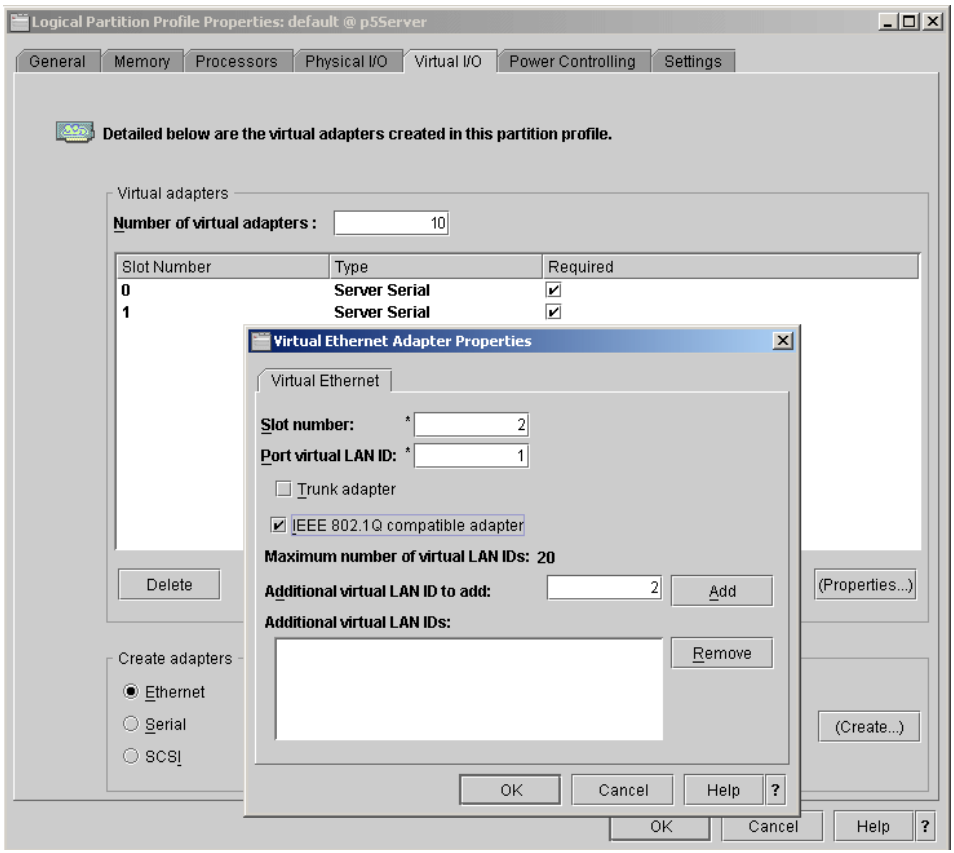

Figure 5-34 Creating the virtual Ethernet adapter for the client partition.

5. Click **OK** to save your changes.
6. Repeat this procedure for each client partition (Apps_Server and Web_Server).

Defining the SCSI client adapters for OS install

Use the following steps to add previously defined resources to a partition using HMC:

1. Right-click the client partition (DB_Server) profile and choose the **Properties** menu.

2. On the Virtual I/O Properties tab select the slot number used for this virtual SCSI adapter.
3. Choose the Remote Partition name or ID that shares the disk with the partition (in our scenario 1 this is the I/O_Server_1) and Remote Partition Virtual Slot Number, which you defined before for your client partition as shown on Figure 5-35.

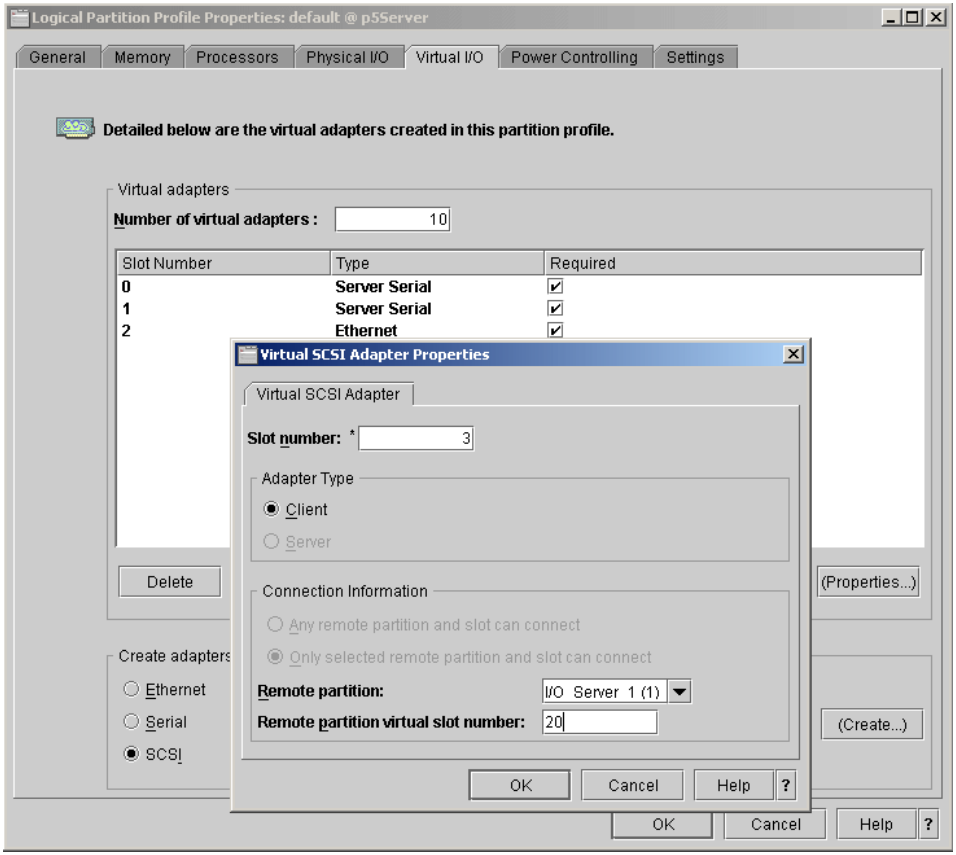

Figure 5-35 Client side SCSI properties

4. Activate this partition by right-clicking the profile name of the partition. Select the **Activate** menu, the **Open a Terminal Window** checkbox, and the **Boot to SMS** option from the Advanced menu option.
5. Repeat these steps for all of your client partitions.

You are now able to start the configuration of the Virtual I/O operating system to make virtual disks available for the partitions so you can install their OS on them.

5.3.6 Virtual I/O Server configuration with mirroring

This section discusses some of the steps required to increase the availability of the Virtual I/O Server (I/O_Server_1 in this scenario); and steps to configure both physical and virtual resources, then assign these resources to the client partitions. The topics included are:

- Mirroring the Virtual I/O servers rootvg volume group onto separate physical disks
- Creating additional volume groups and logical volumes
- Creating a Shared Ethernet Adapter to enable the inter-partition VLAN to communicate with external networks
- Creating Virtual SCSI mapping of logical volumes for client partitions

The Command Line Interface (CLI) of the Virtual I/O Server allows you to configure and assign previously defined resources to the client partitions. Use the following steps to configure your resources:

1. Right click the I/O_Server_1 partition profile, select the **Open a terminal window** checkbox and click **OK**, as shown on Figure 5-36, to activate the I/O_Server_1 partition and launch a terminal.

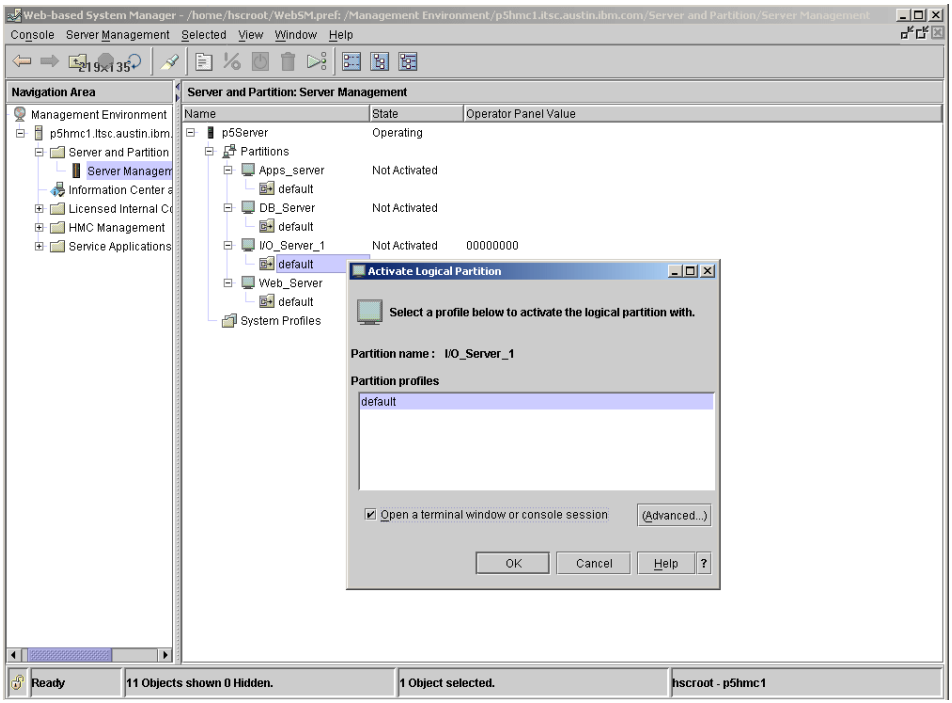

Figure 5-36 Activating the Virtual I/O Partition.

2. Log in with user `padmin` and the password you chose during the installation to enter the Virtual I/O Servers restricted shell that supports uniquely defined commands to execute the following tasks.

Mirroring the rootvg volume group

Once the installation of the Virtual I/O Server is complete, the following commands can be used to mirror the rootvg volume group to a second physical volume.

1. Run the **`extendvg`** command to extend rootvg to include a new physical volume. Make sure that the new hdisk is not part of another volume group.

```
$ extendvg -f rootvg hdisk1
0516-1254 extendvg: Changing the PVID in the ODM.
```

2. Use the **`lspv`** command to confirm that rootvg has been extended to include hdisk1

```
$ lspv
hdisk0          00cddedc6f6cf54b                    rootvg          active
hdisk1          00cddedc74c16ee3                    rootvg          active
hdisk2          none                                None
hdisk3          none                                None
```

3. Run the **`mirrorios`** command to perform the following tasks:
   - MIrror the rootvg to a second physical volume.
   - Synchronize the Physical Partitions (PP) to the mirrored physical volume.
   - Update the boot image file.
   - Update the bootlist to include both physical volumes.

```
$ mirrorios -f hdisk1
```

4. Run the **`lsvg`** command to verify that the process was successful and complete.

```
$ lsvg rootvg
VOLUME GROUP:       rootvg                  VG IDENTIFIER:  00cddedc00004c00006
VG STATE:           active                  PP SIZE:        64 megabyte(s)
VG PERMISSION:      read/write              TOTAL PPs:      1084 (69376 megaby)
MAX LVs:            256                     FREE PPs:       620 (39680 megabyt)
LVs:                11                      USED PPs:       464 (29696 megabyt)
OPEN LVs:           10                      QUORUM:         1
TOTAL PVs:          2                       VG DESCRIPTORS: 3
STALE PVs:          0                       STALE PPs:      0
ACTIVE PVs:         2                       AUTO ON:        yes
MAX PPs per VG:     32512
MAX PPs per PV:     1016                    MAX PVs:        32
LTG size (Dynamic): 256 kilobyte(s)         AUTO SYNC:      no
HOT SPARE:          no                      BB POLICY:      relocatable
```

Note: Confirm that there are no stale PPs in rootvg. Stale PPs suggest the PPs are not synchronized.

5. Use the following command to confirm that the number of PPs for each Logical Volume is twice the number of Logical Partitions (LPs). The only exception is lv00 dumpspace, which is not mirrored.

```
$ lsvg -lv rootvg
rootvg:
LV NAME            TYPE       LPs   PPs   PVs  LV STATE      MOUNT POINT
hd5                boot       1     2     2    closed/syncd  N/A
hd6                paging     8     16    2    open/syncd    N/A
paging00           paging     16    32    2    open/syncd    N/A
hd8                jfs2log    1     2     2    open/syncd    N/A
hd4                jfs2       3     6     2    open/syncd    /
hd2                jfs2       9     18    2    open/syncd    /usr
hd9var             jfs2       9     18    2    open/syncd    /var
hd3                jfs2       16    32    2    open/syncd    /tmp
hd1                jfs2       160   320   2    open/syncd    /home
hd10opt            jfs2       1     2     2    open/syncd    /opt
lv00               sysdump    16    16    1    open/syncd    N/A
```

6. Add the disk to the bootlist with the **bootlist -mode normal hdisk0 cd0 hdisk1** command. Issue the **bootlist** command to confirm that both Physical Volumes are included in the bootlist.

```
$ bootlist -mode normal -ls
hdisk0
cd0
hdisk1
```

Creating volume groups and logical volumes

In the following steps, you will build the logical volumes required to create the virtual disks for the clients install images.

1. Run the **cfgdev** command to rebuild the list of visible devices used by the Virtual I/O Server.

   The virtual SCSI server adapters are now available to the Virtual I/O Server. The name of this adapter will be `vhostX` where X is a number assigned by the system.

2. Run the **lsdev -virtual** command to make sure that your new virtual SCSI adapter is available as shown in the following:

```
$ lsdev -virtual
name            status     description

ent2            Available  Virtual I/O Ethernet Adapter (l-lan)
vhost0          Available  Virtual SCSI Server Adapter
vhost1          Available  Virtual SCSI Server Adapter
vhost2          Available  Virtual SCSI Server Adapter
vsa0            Available  LPAR Virtual Serial Adapter
```

   If the devices are not active, then there was a problem in defining them. You can use the command **rmdev -dev vhost0 -recursive** for each device, and then restart the Virtual I/O server if needed. Upon reboot, the configuration manager of the system will detect the hardware and re-create the vhost devices.

   In Scenario 1 you have to create a volume group called rootvg_clients with a disk hdisk2 inside (hdisk0 and hdisk1 will be part of the rootvg volume group as system disks for your Virtual I/O Server).

3. Create a volume group and assign disk to this volume group using the **mkvg** command as follows:

```
$ mkvg -f -vg rootvg_clients hdisk2
rootvg_clients
```

4. Define the logical volume which will be visible as a disk to the client partition. The size of this logical volumes will act as the size of disks which will be available to the client partition. Use the **mklv** command to a create 2 GB size logical volume called rootvg_dbsrv as shown in the following:

```
$ mklv -lv  rootvg_dbsrv rootvg_clients 2G
rootvg_dbsrv
```

5. Repeat this procedure to create the logical volume for each partition.

Creating virtual SCSI mapping

The SCSI mappings allow you to create the virtual target device and map this device to the logical volume.

At times it may be valuable to determine the physical location code of a virtual device, and use that to make sure you are working with the one you think you are. In the previous steps, you determined you had three vhost devices, but to determine which belong to what client, enter the following command:

```
$ lsdev -vpd | grep vhost
  vhost2          U9111.520.10DDEEC-V2-C40         Virtual SCSI Server Adapter
  vhost1          U9111.520.10DDEEC-V2-C30         Virtual SCSI Server Adapter
  vhost0          U9111.520.10DDEEC-V2-C20         Virtual SCSI Server Adapter
```

Or, enter the following command if you want to check a specific vhost:

```
$ lsdev -dev vhost0 -vpd
  vhost0           U9111.520.10DDEEC-V2-C20  Virtual SCSI Server Adapter

        Device Specific.(YL)........U9111.520.10DDEEC-V2-C20


  PLATFORM SPECIFIC

  Name:  v-scsi-host
    Node:  v-scsi-host@30000014
    Physical Location: U9111.520.10DDEEC-V2-C20
```

At this time, you can proceed with your configuration using the following steps:

1. Create a virtual target device, which maps the newly created virtual SCSI server adapters to a logical volume, by running the **mkvdev** command as shown below:

   ```
   $ mkvdev -vdev rootvg_dbsrv -vadapter vhost0 -dev vdbsrv
   vdbsrv Available
   ```

   where `rootvg_dbsrv` is a logical volume you have created before, `vhost0` is your new virtual SCSI adapter and `vdbsrv` is the name of the new target device which will be available to the client partition.

2. Repeat this step for all previously created logical volumes and virtual SCSI adapters, specifying the different name for each target device.

The following steps show a way to check that all previous commands have been finished correctly as planned:

1. Use the **lsdev** command to ensure that all virtual target devices are created as shown in the following:

   ```
   $ lsdev -virtual
   name            status     description

   ent2            Available  Virtual I/O Ethernet Adapter (l-lan)
   vhost0          Available  Virtual SCSI Server Adapter
   vhost1          Available  Virtual SCSI Server Adapter
   vhost2          Available  Virtual SCSI Server Adapter
   vsa0            Available  LPAR Virtual Serial Adapter
   vappssrv        Available  Virtual Target Device - Logical Volume
   vdbsrv          Available  Virtual Target Device - Logical Volume
   vwebsrv         Available  Virtual Target Device - Logical Volume
   ```

2. Use the **lsmap** command to ensure that all logical connections between newly created devices are correct as shown in the following:

```
$ lsmap -vadapter vhost0
SVSA            Physloc                                      Client PartitionID
--------------- -------------------------------------------- ------------------
vhost0          U9111.520.10DDEEC-V1-C20                     0x00000000

VTD                   vdbsrv
LUN                   0x8100000000000000
Backing device        rootvg_dbsrv
Physloc
```

When all of these procedures are finished, you can attach the virtual SCSI disks to the client partition and install the operating system on the client partition using this logical volume as a virtual SCSI disk.

Creating additional virtual SCSI disks

If the operating system is already installed on the client partition you can add additional disk to the client partition at this time.

From the Virtual I/O Server partition, repeat the procedure described in “Creating Virtual SCSI server adapters” on page 164 to create another logical volume, then create another virtual target device, which maps the virtual SCSI server adapter to a logical volume.

1. On the Virtual I/O Server, create another volume group called datavg_clients as follows:

```
$ mkvg -f -vg datavg_clients hdisk3
```

2. Create another logical volume called datavg_dbsrv as follows:

```
$ mklv -lv  datavg_dbsrv datavg_clients 2G
```

3. Map this logical volume to the virtual server adapter using the **mkvdev** command as follows:

```
$ mkvdev -vdev datavg_dbsrv -vadapter vhost0 -dev vdbsrvdata
```

The datavg_dbsrv logical volume will be visible from the client partition as a disk.

The rest of this configuration should be done using terminal window or SMIT on the client partition you want to access shared disk from. In this scenario you have do this from partition called DB_Server as follows:

1. Log on as root to the DB_Server partition using terminal client or open the terminal window from the HMC.

2. Use the **cfgmgr** command to refresh the device list. Two new devices should be visible using the **lsdev** command. In our scenario the new devices are: vscsi1 and hdisk1, as shown in the following:

```
# cfgmgr
# lsdev |grep Virtual
ent0           Available       Virtual I/O Ethernet Adapter (l-lan)
hdisk0         Available       Virtual SCSI Disk Drive
hdisk1         Available       Virtual SCSI Disk Drive
vio0           Available       Virtual I/O Bus
vsa0           Available       LPAR Virtual Serial Adapter
vscsi0         Available       Virtual SCSI Client Adapter
```

3. Assign the new hdisk1 to the new volume group using the **mkvg** command as follows:

```
# mkvg -f -y datavg hdisk1
```

4. Verify how much free space is on this newly added disk using the **lspv** command as follows:

```
# lsvg datavg
VOLUME GROUP:       datavg                   VG IDENTIFIER:  00cddeec00004c00000000fd7d331ec8
VG STATE:           active                   PP SIZE:        4 megabyte(s)
VG PERMISSION:      read/write               TOTAL PPs:      511 (2044 megabytes)
MAX LVs:            256                      FREE PPs:       511 (2044 megabytes)
LVs:                0                        USED PPs:       0 (0 megabytes)
OPEN LVs:           0                        QUORUM:         2
TOTAL PVs:          1                        VG DESCRIPTORS: 2
STALE PVs:          0                        STALE PPs:      0
ACTIVE PVs:         1                        AUTO ON:        yes
MAX PPs per VG:     32512
MAX PPs per PV:     1016                     MAX PVs:        32
LTG size (Dynamic): 128 kilobyte(s)          AUTO SYNC:      no
HOT SPARE:          no                       BB POLICY:      relocatable
```

You previously created the logical volume with 2 GB of free space. This free space is visible from the client side as a 2 GB disk.

5. Make a logical volume in this volume group using the **mklv** command as follows:

```
# mklv -y data1lv -t jfs2 datavg 500M
data1lv
```

This command creates a jfs2 type logical volume named data1lv with size 500 MB in volume group datavg.

6. Make the file system on the previously created logical volume using the **crfs** command as follows:

```
# crfs -v jfs2 -d data1lv -m /data
File system created successfully.
511780 kilobytes total disk space.
New File System size is 1024000
```

7. Mount the file system using the **mount** command as follows:

```
# mount /data
# df -k
Filesystem    1024-blocks      Free %Used    Iused %Iused Mounted on
/dev/hd4            16384      5944   64%     1402    47% /
/dev/hd2           516096     20496   97%    15768    72% /usr
/dev/hd9var         16384     11244   32%      302    11% /var
/dev/hd3            28672     28280    2%       23     1% /tmp
/dev/hd1            16384     16032    3%        5     1% /home
/proc                   -         -    -        -     -  /proc
/dev/hd10opt        36864     14856   60%      474    13% /opt
/dev/data1lv       512000    511592    1%        4     1% /data
```

All of the virtual disks as well as physical disks can be used in a native way, as you are accustomed to doing on AIX 5L systems.

Creating a shared Ethernet adapter

To create a shared Ethernet adapter, perform the following steps.

1. Run the following command in I/O_Server_1 to verify that the Virtual Ethernet trunk adapter is available.

```
$ lsdev -virtual
name            status     description

ent2            Available  Virtual I/O Ethernet Adapter (l-lan)
vhost0          Available  Virtual SCSI Server Adapter
vhost1          Available  Virtual SCSI Server Adapter
vhost2          Available  Virtual SCSI Server Adapter
vsa0            Available  LPAR Virtual Serial Adapter
vappssrv        Available  Virtual Target Device - Logical Volume
vdbsrv          Available  Virtual Target Device - Logical Volume
vwebsrv         Available  Virtual Target Device - Logical Volume
```

2. Select the appropriate physical Ethernet adapter which will be used to create the Shared Ethernet Adapter. The **lsdev** command will show a list of available physical adapters.

```
$ lsdev -type adapter
name            status     description

ent0            Available  2-Port 10/100/1000 Base-TX PCI-X Adapter
(14108902)
ent1            Available  2-Port 10/100/1000 Base-TX PCI-X Adapter
(14108902)
ent2           Available  Virtual I/O Ethernet Adapter (l-lan)
ide0           Available  ATA/IDE Controller Device
sisscsia0      Available  PCI-X Dual Channel Ultra320 SCSI Adapter
vhost0         Available  Virtual SCSI Server Adapter
```

```
vhost1          Available  Virtual SCSI Server Adapter
vhost2          Available  Virtual SCSI Server Adapter
vsa0             Available  LPAR Virtual Serial Adapter
```

3. Use the **mkvdev** command to create a new ent3 device as the Shared Ethernet Adapter. ent0 will be used as the physical Ethernet adapter and ent2 as the Virtual Ethernet adapter.

```
$ mkvdev -sea ent0 -vadapter ent2 -default ent2 -defaultid 1
ent3 Available
en3
et3
```

4. Confirm that the newly created Shared Ethernet Adapter is available.

```
$ lsdev -virtual
name            status     description

ent2            Available  Virtual I/O Ethernet Adapter (l-lan)
vhost0          Available  Virtual SCSI Server Adapter
vhost1          Available  Virtual SCSI Server Adapter
vhost2          Available  Virtual SCSI Server Adapter
vsa0            Available  LPAR Virtual Serial Adapter
vappssrv        Available  Virtual Target Device - Logical Volume
vdbsrv          Available  Virtual Target Device - Logical Volume
vwebsrv         Available  Virtual Target Device - Logical Volume
ent3            Available  Shared Ethernet Adapter
```

This Shared Ethernet Adapter will form a bridge allowing communication between the inter-partition VLAN and the external network.

You do not need to configure an IP address for the Shared Ethernet adapter. But in our scenario it is the only physical network connection and we need an IP interface so we can use the dynamic reconfiguration function to add virtual adapters later on in scenario 2.

For our scenario we used the following network settings:

hostname	p5_2ios1
IP-address	9.3.5.150
netmask	255.255.255.0
gateway	9.3.5.41

Use the **mktcpip** command to configure the newly created Shared Ethernet interface ent3.

```
mktcpip -hostname p5_2ios1 -inetaddr 9.3.5.150 -interface en3 -netmask
255.255.255.0 -gateway 9.3.5.41
```

5.3.7 Client partition installation

This section describes the various methods available to install AIX Version 5.3 onto a previously defined client partition. Any of the following methods can also be used to install the DB_Server, Web_Server, and Apps_Server partitions. In this scenario, installation methods for the defined clients are used in the following sequence:

1. DB_Server partition is installed using NIM.
2. Apps_Server partition is installed using Alternate disk installation.
3. Web_Server partition is installed using **mksysb** installation.

Network Installation Manager (NIM) with virtual devices

The installation of AIX on virtual devices using NIM is no different than installation on physical devices. Refer to *The Complete Partitioning Guide for IBM eServer pSeries Servers*, SG24-7039 for detailed information on installing AIX using NIM.

Assuming that a NIM master is configured, the following are the basic steps required to perform an AIX installation using NIM:

1. Set up the DB_Server client partition as a NIM client.
2. Define resources to the NIM client partition.
3. Initiate the installation process on the NIM master.
4. Configure the DB_Server client partition to proceed with the installation using the client partition's SMS menus. Activate this partition by right-clicking the profile name and selecting Activate with the Open a Terminal Window checkbox. To boot into SMS use the Advanced menu option, as shown in Figure 5-37.

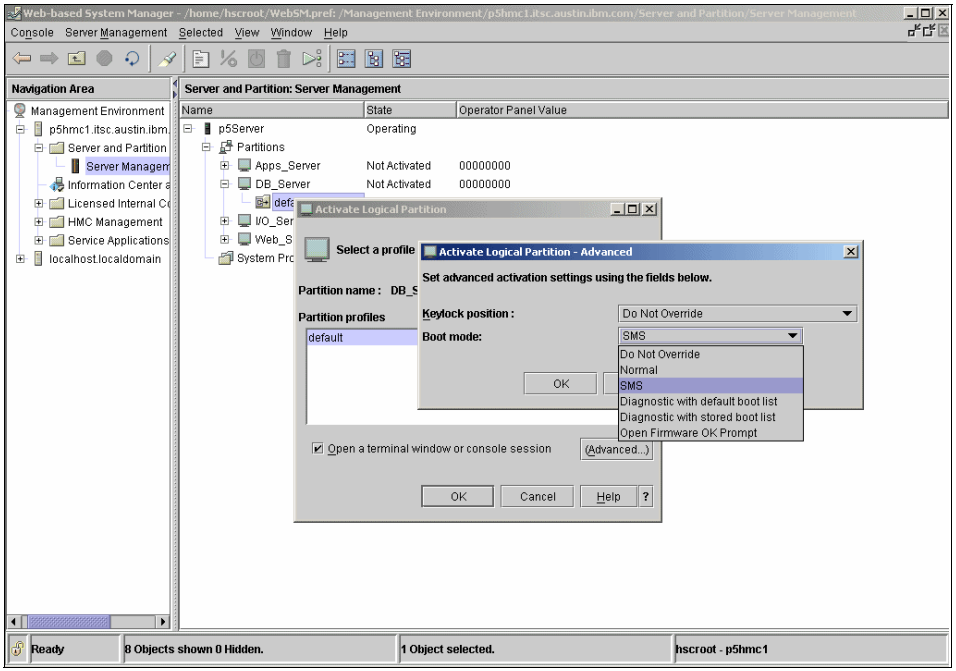

Figure 5-37 Profile activation menu

5. Use the Virtual Ethernet adapter to initiate the installation of the client partition as shown in Figure 5-38.

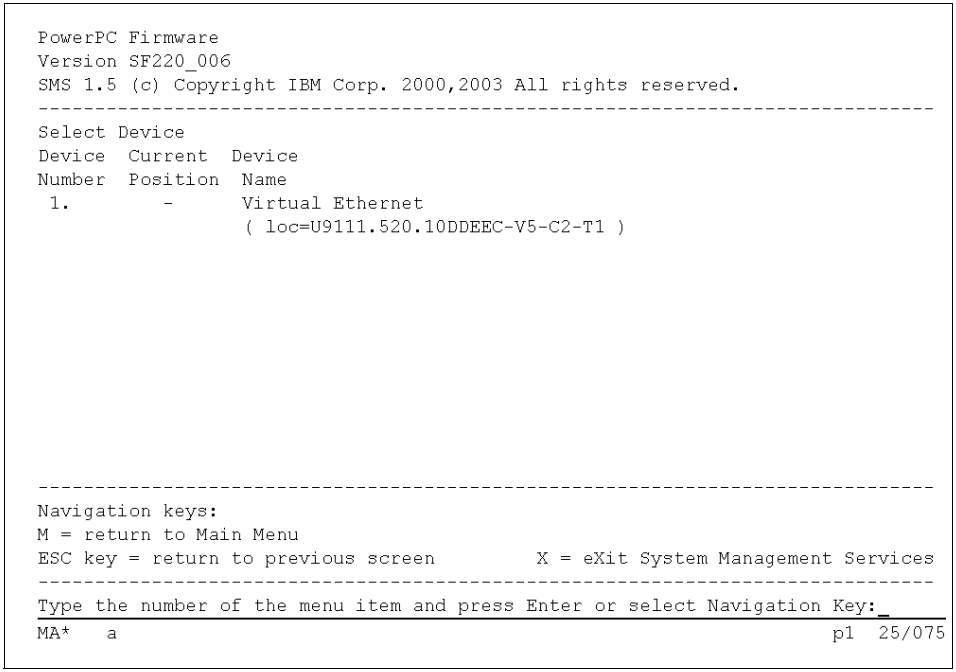

Figure 5-38 Bootable Virtual Ethernet adapter in the SMS menu

6. Use the Virtual disk assigned to the DB_Server (hdisk0) to install the base operating system as shown in Figure 5-39.

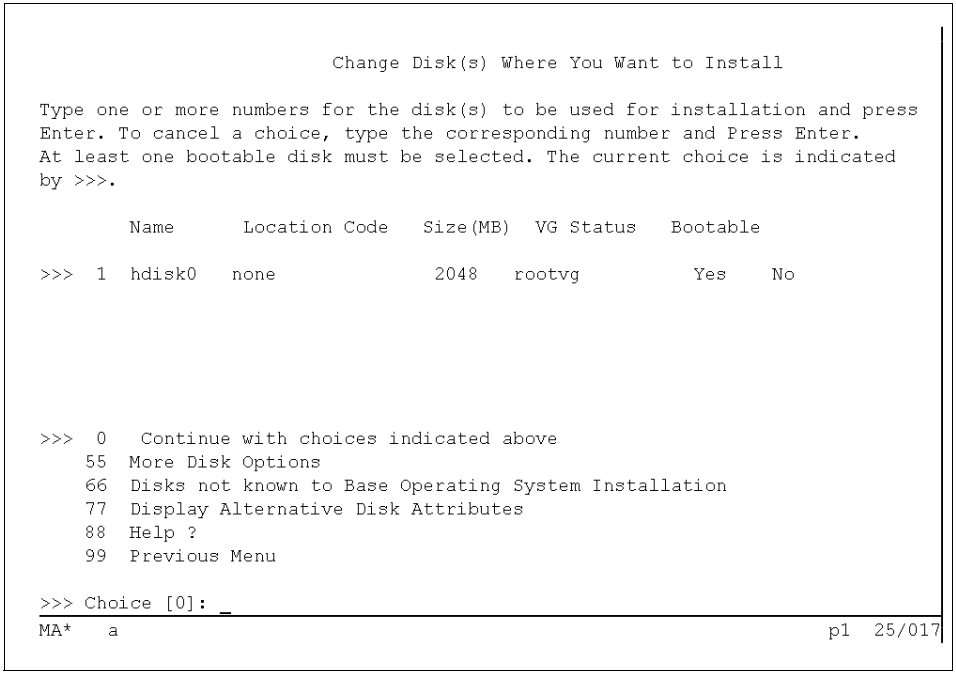

Figure 5-39 Bootable Virtual SCSI disk in the AIX OS installation menu

Alternate disk installation with virtual devices

This section discusses how to install AIX using the alternate disk installation method onto a virtual disk on the preciously installed DB_Server partition. This disk is then redefined to the Apps_Server partition to be used as its rootvg volume group. For more information on alternate disk installation methods, refer to the following literature:

- *The Complete Partitioning Guide for IBM eServer pSeries Servers*, SG24-7039
- *IBM AIX 5L Version 5.3 Installation Guide* included with the AIX product documentation

Alternate disk rootvg cloning

The first step is to configure a new virtual disk to a running server partition. In this scenario, the DB_Server partition will be used to clone the alternate rootvg. This disk will later be redefined to the Apps_Server partition.

An vappssrv2 virtual target device is created in the Virtual I/O Server using the vhost0 virtual SCSI server adapter and a rootvg_appssrv2 logical volume. Follow the steps described in “Creating additional virtual SCSI disks” on page 175 to create a new virtual disk to a running partition.

1. Check that a free disk is available and is not assigned to a volume group using the **lspv** command.

```
# lspv
hdisk0          00cddeec0a3b6ce0                    rootvg          active
hdisk1          00cddeec6b9177b9                    datavg          active
hdisk2          none                                None
```

2. Run the **alt_disk_install** command on the DB_Server partition. The **-C** option causes the current rootvg to be cloned to the target disk hdisk2.

   This command creates the alternate rootvg volume group on hdisk2 as a cloned image from the rootvg on hdisk0. The produced alternate rootvg has the name altinst_rootvg. The file system and logical volume names of the altinst_rootvg have the prefix alt_.

Note: The **-O** option removes the device definition from the ODM in altinst_rootvg. Using this option, the phantom devices will not appear on the target partition. The phantom devices are those that currently do not exist in the system and appear as defined in the **lsdev -C** command output.

```
# alt_disk_install -C -O hdisk2
+-----------------------------------------------------------------------------+
ATTENTION: calling new module /usr/sbin/alt_disk_copy. Please see the
alt_disk_copy man page and documentation for more details.
Executing command: /usr/sbin/alt_disk_copy -O -d "hdisk2"
+-----------------------------------------------------------------------------+
Calling mkszfile to create new /image.data file.
Expanding /tmp.
Filesystem size changed to 57344
Checking disk sizes.
Creating cloned rootvg volume group and associated logical volumes.
Creating logical volume alt_hd5.
Creating logical volume alt_hd6.
Creating logical volume alt_hd8.
Creating logical volume alt_hd4.
Creating logical volume alt_hd2.
Creating logical volume alt_hd9var.
Creating logical volume alt_hd3.
Creating logical volume alt_hd1.
Creating logical volume alt_hd10opt.
Creating /alt_inst/ file system.
Creating /alt_inst/home file system.
Creating /alt_inst/opt file system.
```

```
Creating /alt_inst/tmp file system.
Creating /alt_inst/usr file system.
Creating /alt_inst/var file system.
Generating a list of files
for backup and restore into the alternate file system...
Backing-up the rootvg files and restoring them to the alternate file
system...
Modifying ODM on cloned disk.
Building boot image on cloned disk.
Resetting all device attributes.
NOTE: The first boot from altinst_rootvg will prompt to define the new
system console.
Resetting all device attributes.
NOTE: The first boot from altinst_rootvg will prompt to define the new
system console.
forced unmount of /alt_inst/var
forced unmount of /alt_inst/usr
forced unmount of /alt_inst/tmp
forced unmount of /alt_inst/opt
forced unmount of /alt_inst/home
forced unmount of /alt_inst
forced unmount of /alt_inst
Changing logical volume names in volume group descriptor area.
Fixing LV control blocks...
Fixing file system superblocks...
Bootlist is set to the boot disk: hdisk2
```

At completion of this command, the volume group altinst_rootvg is automatically created as a clone of rootvg.

3. Use the **lspv** command to verify the created alt_rootvg volume group:

```
# lspv
hdisk0          00cddedc03a34381                    rootvg          active
hdisk1          00cddedcb48b82cf                    datavg          active
hdisk2          00cddedcb48d7c43                    altinst_rootvg
```

4. Remove the information created by the **alt_disk_install** command from the ODM on the DB_Server partition. This command also puts the boot device back to hdisk0:

```
# alt_disk_install -X
+-----------------------------------------------------------------------+
ATTENTION: calling new module /usr/sbin/alt_rootvg_op. Please see the
alt_rootvg_op man page and documentation for more details.
Executing command: /usr/sbin/alt_rootvg_op -X
+-----------------------------------------------------------------------+
Bootlist is set to the boot disk: hdisk0
```

5. Use the **lspv** command to confirm the removal of alt_rootvg:

```
# lspv
hdisk0          00cddedc03a34381                    rootvg          active
hdisk1          00cddedcb48b82cf                    datavg          active
hdisk2          00cddedcb48d7c43                    None
```

   This completes the cloning of rootvg using the **alt_disk_install** command.

6. The next step is to log in to the Virtual I/O Server, and redefine the rootvg_appssrv2 logical volume to the Apps_Server partition. List the virtual devices within the Virtual I/O server:

```
$ lsdev -virtual
name            status     description

ent2            Available  Virtual I/O Ethernet Adapter (l-lan)
vhost0          Available  Virtual SCSI Server Adapter
vhost1          Available  Virtual SCSI Server Adapter
vhost2          Available  Virtual SCSI Server Adapter
vsa0            Available  LPAR Virtual Serial Adapter
vappssrv        Available  Virtual Target Device - Logical Volume
vappssrv2       Available  Virtual Target Device - Logical Volume
vdatavg_dbsrv   Available  Virtual Target Device - Logical Volume
vdbsrv          Available  Virtual Target Device - Logical Volume
vwebsrv         Available  Virtual Target Device - Logical Volume
ent3            Available  Shared Ethernet Adapter
```

7. Remove the device named vappssrv2 as follows:

```
$ rmdev -dev vappssrv2
vappssrv2 deleted
```

   Create a new virtual target device (vappssrv2 in this example) using the vhost1 virtual SCSI server adapter (which is assigned to the Apps_Server partition) and the rootvg_appssrv2 logical volume.

8. List all the mapped Virtual SCSI devices.

```
$ lsmap -all
SVSA            Physloc                                     Client Partition ID
--------------- ------------------------------------------- ------------------
vhost0          U9111.520.10DDEDC-V1-C20                    0x00000002

VTD                   vdatavg_dbsrv
LUN                   0x8300000000000000
Backing device        datavg_dbsrv
Physloc

VTD                   vdbsrv
LUN                   0x8100000000000000
Backing device        rootvg_dbsrv
Physloc
```

```
SVSA            Physloc                                     Client Partition ID
--------------- ------------------------------------------- ------------------
vhost1          U9111.520.10DDEDC-V1-C30                    0x00000000

VTD                   vappssrv
LUN                   0x8100000000000000
Backing device        rootvg_appssrv
Physloc

SVSA            Physloc                                     Client Partition ID
--------------- ------------------------------------------- ------------------
vhost2          U9111.520.10DDEDC-V1-C40                    0x00000000

VTD                   vwebsrv
LUN                   0x8100000000000000
Backing device        rootvg_websrv
Physloc
```

9. Create the virtual SCSI target device using the **mkvdev** command.

```
$ mkvdev -vdev rootvg_appssrv2 -vadapter vhost1 -dev vappssrv2
vappssrv2 Available
```

10. Use the **lsdev** command to verify that the new redefine virtual SCSI device is now available and mapped to the Apps_Server partition.

```
$ lsdev -virtual
name            status     description

ent2            Available  Virtual I/O Ethernet Adapter (l-lan)
vhost0          Available  Virtual SCSI Server Adapter
vhost1          Available  Virtual SCSI Server Adapter
vhost2          Available  Virtual SCSI Server Adapter
vsa0            Available  LPAR Virtual Serial Adapter
vappssrv        Available  Virtual Target Device - Logical Volume
vappssrv2       Available  Virtual Target Device - Logical Volume
vdatavg_dbsrv   Available  Virtual Target Device - Logical Volume
vdbsrv          Available  Virtual Target Device - Logical Volume
vwebsrv         Available  Virtual Target Device - Logical Volume
ent3            Available  Shared Ethernet Adapter
```

11. Boot up the Apps_Server partition using the HMC. The alt_rootvg volume group created on the disk will be renamed rootvg and function as the root volume group for the Apps_Server partition.

This completes the cloning of rootvg using the Alternate disk installation method.

Installing on virtual disks using CD media

The section describes a procedure to install AIX in the Web server partition using the AIX CD media.

1. Make sure that the appropriate media devices are allocated to the Web_Server client partition. If configured correctly, this can be done dynamically using the HMC.
2. Activate the Web_Server partition into the SMS menu as shown on Figure 5-37 on page 180.
3. Once in the SMS menu, select the CD ROM as the installation device.
4. After booting from product media, follow the appropriate steps when the Welcome to the Base Operating System Installation and Maintenance screen is displayed. Be sure to select the virtual SCSI device to install (In this case hdisk0).

This completes the AIX CD media installation onto a Virtual SCSI device.

5.4 Scenario 2: Enhanced availability virtualization

This section expands on scenario 1 to provide the steps required to set up and configure a highly available environment which includes both physical adapter and Virtual I/O Server redundancy. The creation of scenario 2 includes:

- Installing and configuring I/O_Server_2
- Creating a Link Aggregation (EtherChannel) device on the Virtual I/O Servers
- Creating a Shared Ethernet Adapter using Link Aggregation
- Creating Link Aggregation (EtherChannel) on the client partitions
- Mirroring rootvg on the client partitions

All discussions and commands shown in this section are conducted on the I/O_Server_2 and DB_Server partitions. All tasks should also be replicated on the I/O_Server_1, Apps_Server, and Web_Server as appropriate, but we do not present the details for each procedure here.

5.4.1 Installation and configuration of a second Virtual I/O Server

I/O_Server_2 is configured and installed in the same manner as I/O_server_1 (refer to 5.3.3, “Virtual I/O Server software installation” on page 157). It is assigned the same amount of physical and virtual resources as I/O_server_1. In this scenario, the Virtual Ethernet trunk adapter is the only virtual resource configured prior to the initial startup of I/O_Server_2.

The Virtual SCSI server adapters are created dynamically later in this section.

Note: The Virtual Ethernet adapter used for this Shared Ethernet Adapter is assigned to VLAN 2 using the PVID as shown in Figure 5-40 on page 188. Separate VLANs are used to isolate network traffic between the two Virtual I/O Servers.

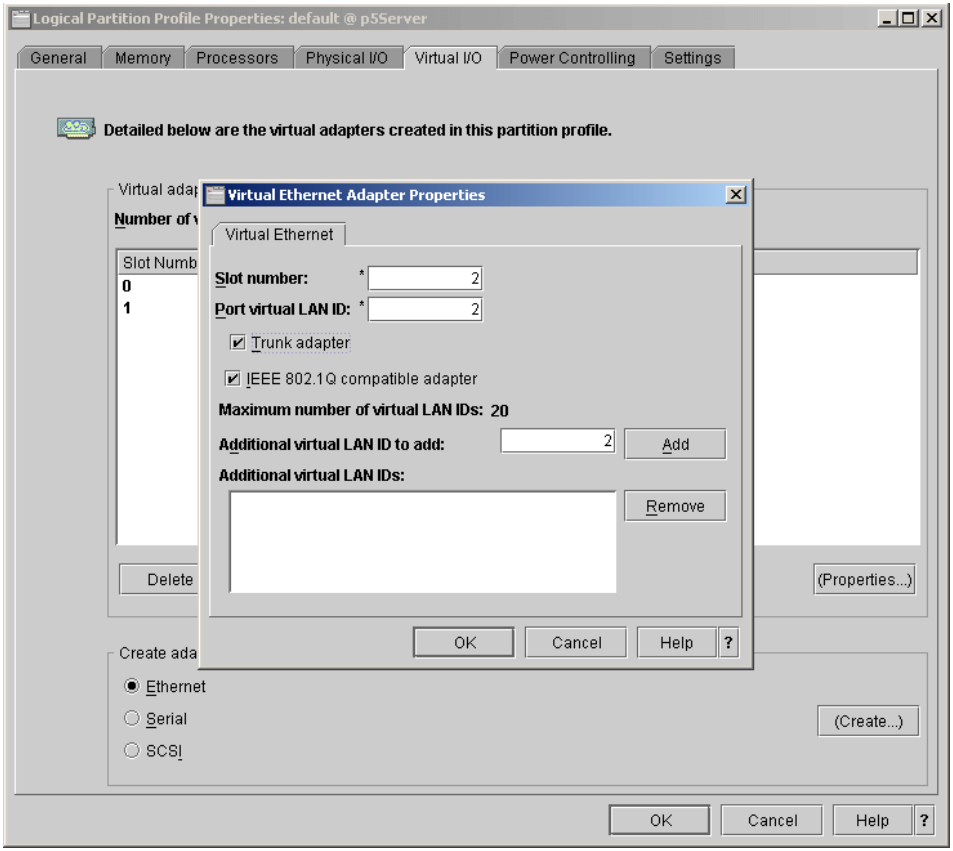

Figure 5-40 Virtual Ethernet adapter properties

5.4.2 Creating Link Aggregation on the Virtual I/O Server

In this task, Link Aggregation (EtherChannel) devices are used on both Virtual I/O Servers to provide redundant and load-balanced physical network connections. Each EtherChannel device is configured with two physical adapters.

1. List the network adapters which are available on I/O_Server_2 using the **lsdev** command:

```
$ lsdev -dev ent*
name status    description

ent0 Available 2-Port 10/100/1000 Base-TX PCI-X Adapter (14108902)
ent1 Available 2-Port 10/100/1000 Base-TX PCI-X Adapter (14108902)
ent2 Available Virtual I/O Ethernet Adapter (l-lan)
```

2. To create the EtherChannel device, adapters ent0 and ent1 are be used. Use the **mkvdev -lnagg** command to define the EtherChannel on I/O_Server_2.

```
$ mkvdev -lnagg ent0 ent1
ent3 Available
en3
et3
$
```

3. Use the **lsdev** command to show the detailed attributes of the newly created EtherChannel device.

```
$ lsdev -dev ent3 -attr
attribute       value          description                                user_se
ttable

adapter_names   ent0,ent1      EtherChannel Adapters                      True
alt_addr        0x000000000000 Alternate EtherChannel Address             True
backup_adapter  NONE           Adapter used when whole channel fails      True
hash_mode       default        Determines how outgoing adapter is chosen True
mode            standard       EtherChannel mode of operation             True
netaddr                        Address to ping                            True
num_retries     3              Times to retry ping before failing         True
retry_time      1              Wait time (in seconds) between pings       True
use_alt_addr    no             Enable Alternate EtherChannel Address      True
use_jumbo_frame no             Enable Gigabit Ethernet Jumbo Frames       True
$
```

4. Repeat these steps on I/O_Server_1.

Note: The physical adapters that will be assigned to the EtherChannel device *must* be unconfigured in either a down or detached state prior to configuring the EtherChannel device.

This completes the creation of the EtherChannel device on the Virtual I/O Servers.

5.4.3 Creating Shared Ethernet Adapters using Link Aggregation

A Shared Ethernet Adapter can now be defined using the EtherChannel device created in the previous section.

1. Run the **lsdev** command to show the available network adapters in I/O_Server_2.

```
$ lsdev -dev ent*
name status    description

ent0 Available 2-Port 10/100/1000 Base-TX PCI-X Adapter (14108902)
ent1 Available 2-Port 10/100/1000 Base-TX PCI-X Adapter (14108902)
ent2 Available Virtual I/O Ethernet Adapter (l-lan)
ent3 Available EtherChannel / IEEE 802.3ad Link Aggregation
```

Note: The ent3 EtherChannel and the virtual Ethernet adapter ent2 are used to create the Shared Ethernet Adapter. Separate VLANs are used to isolate network traffic between the two Virtual I/O Servers. Traffic to I/O_Server_1 will be sent on VLAN 1 while traffic to I/O_server_2 is sent on VLAN 2.

2. Execute the following command to create the Shared Ethernet Adapter on I/O_Server_2:

```
$ mkvdev -sea ent3 -vadapter ent2 -default ent2 -defaultid 2
ent4 Available
en4
et4
$
```

3. To display the attributes of the newly created adapter use the following command:

```
$ lsdev -dev ent4 -attr
attribute     value description                                          user_settable

pvid          1     PVID to use for the SEA device                                True
pvid_adapter  ent2  Default virtual adapter to use for non-VLAN-tagged packets    True
real_adapter  ent3  Physical adapter associated with the SEA                      True
virt_adapters ent2  List of virtual adapters associated with the SEA (comma separated) True
```

4. Repeat these steps on I/O_Server_1.

This completes the creation of the Shared Ethernet Adapter using EtherChannel device.

5.4.4 Creating Link Aggregation in client partitions

In this scenario, the Virtual Ethernet adapters ent0 and ent1 are used to create a Link Aggregation (Etherchannel) device running in Network Interface Backup mode.

The primary adapter is configured as an EtherChannel in standard mode with one adapter and the backup adapter is configured as a backup of that EtherChannel. The DB_Server is used to show this function.

1. Add a second Virtual Ethernet adapter to the DB_Server using the HMC as shown in Figure 5-41.

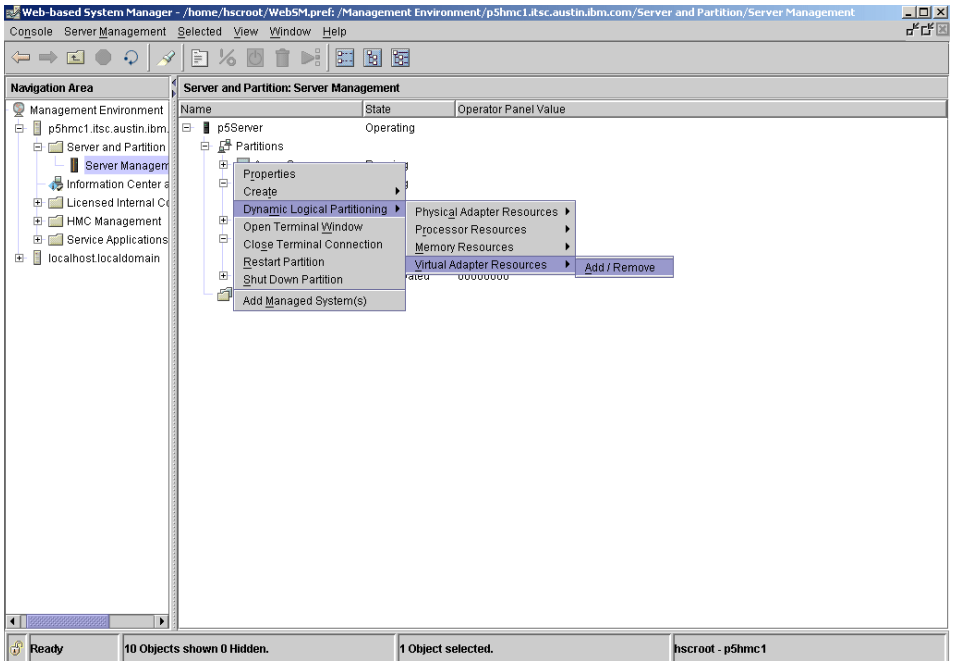

Figure 5-41 Dynamic virtual adapter resource menu

2. Create a Virtual Ethernet adapter. Be sure to set the Port Virtual LAN ID to 2 as shown in Figure 5-42.

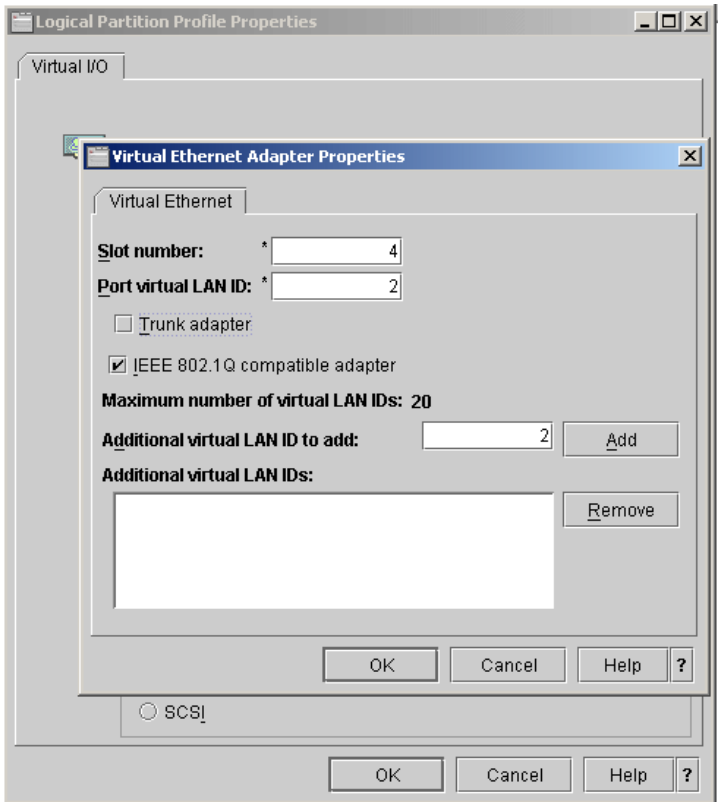

Figure 5-42 Virtual Ethernet adapter properties

Important: This is a dynamic function which enables the use of the adapter without requiring a reboot. The adapter *must* also be added to each of the partition profiles in order for the device to be available to the partition on reboot. Failure to do this will result in the adapter definition being removed on reboot.

3. Run **`cfgmgr`** on the DB_Server partition to pick up the newly created Ethernet device.

4. The **lscfg** commands shows the two Virtual Ethernet adapters.

```
# lscfg -v | grep ent
  Model Implementation: Multiple Processor, PCI bus
  ent2           U9111.520.10DDEDC-V2-C4-T1           Virtual I/O Ethernet Adapter (l-lan)
  vscsi0         U9111.520.10DDEDC-V2-C3-T1           Virtual SCSI Client Adapter
  ent0           U9111.520.10DDEDC-V2-C2-T1           Virtual I/O Ethernet Adapter (l-lan)
```

5. Detach the ent0 device using the following command.

```
# chdev -l en0 -a state=detach
en0 changed
```

6. Use **smitty etherchannel** to Add An EtherChannel / Link Aggregation and select the appropriate settings to create the EtherChannel device as shown in Figure 5-43.

```
                        Add an EtherChannel / Link Aggregation

Type or select values in entry fields.
Press Enter AFTER making all desired changes.

                                                        [Entry Fields]
  EtherChannel / Link Aggregation Adapters               ent0                +
  Enable Alternate Address                              no                  +
  Alternate Address                                     []                  +
  Enable Gigabit Ethernet Jumbo Frames                  no                  +
  Mode                                                  standard            +
  Hash Mode                                             default             +
  Backup Adapter                                        ent1                +
  Internet Address to Ping                              [9.3.5.41]
  Number of Retries                                     []                  +#
  Retry Timeout (sec)                                   []                  +#



F1=Help              F2=Refresh           F3=Cancel            F4=List
Esc+5=Reset          F6=Command           F7=Edit              F8=Image
F9=Shell             F10=Exit             Enter=Do
```

Figure 5-43 Add EtherChannel using SMIT

Note: In this example, ent0 is used as the primary device and ent1 as the backup device. The default gateway was also used as the Internet Address to ping.

7. Use the **lsattr** command to see the attributes of the newly created EtherChannel:

```
# lsattr -El ent2
adapter_names   ent0           EtherChannel Adapters                     True
alt_addr        0x000000000000 Alternate EtherChannel Address            True
backup_adapter  ent1           Adapter used when whole channel fails     True
hash_mode       default        Determines how outgoing adapter is chosen True
mode            standard       EtherChannel mode of operation            True
netaddr         9.3.5.41       Address to ping                           True
num_retries     3              Times to retry ping before failing        True
retry_time      1              Wait time (in seconds) between pings      True
use_alt_addr    no             Enable Alternate EtherChannel Address     True
use_jumbo_frame no             Enable Gigabit Ethernet Jumbo Frames      True
```

8. Configure the new en2 device using **smitty tcpip**, as shown in Figure 5-44.

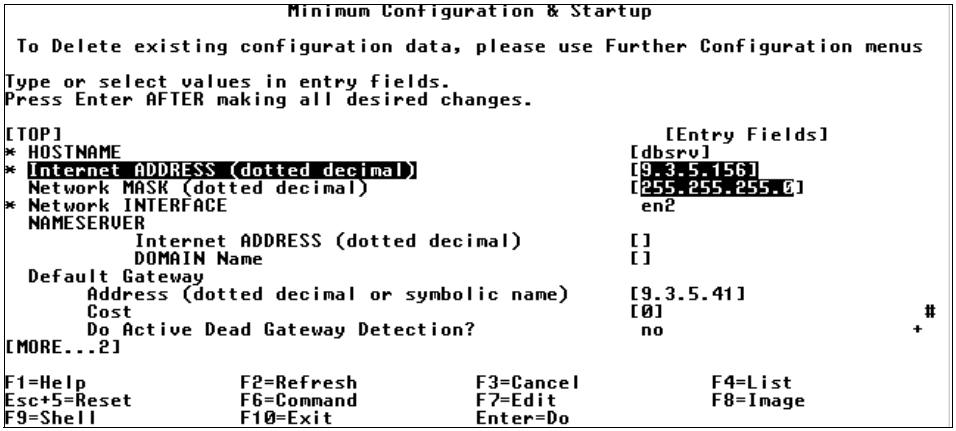

Figure 5-44 Network configuration and startup using smitty

9. When activating the EtherChannel, the following entries appear in the errorlog:

```
# errpt
IDENTIFIER TIMESTAMP  T C RESOURCE_NAME  DESCRIPTION
5561971C   0712155304 P S ent2           UNSUPPORTED IOCTL IN DEVICE DRIVER
5561971C   0712155304 P S ent2           UNSUPPORTED IOCTL IN DEVICE DRIVER
5561971C   0712155104 P S ent2           UNSUPPORTED IOCTL IN DEVICE DRIVER
5561971C   0712155104 P S ent2           UNSUPPORTED IOCTL IN DEVICE DRIVER
F89FB899   0712150004 P O dumpcheck      The copy directory is too small.
A6DF45AA   0712125304 I O RMCdaemon      The daemon is started.
2BFA76F6   0101111870 T S SYSPROC        SYSTEM SHUTDOWN BY USER
9DBCFDEE   0101112170 T O errdemon       ERROR LOGGING TURNED ON
```

Note: These errors occur because EtherChannel is not fully supported for virtual Ethernet. However, as this example shows, it is possible to configure a single network adapter as an EtherChannel device in standard mode using the backup function only since this does not require switch support from the POWER Hypervisor.

10. Repeat these steps on the Apps_Server and Web_Server partitions.

This completes the creation of an EtherChannel device using Virtual Ethernet adapters.

5.4.5 Adding a new disk to a running client

1. On the HMC, right-click the I/O_Server_2 partition and select **Dynamic Logical Partitioning** → **Virtual Adapter Resources** → **Add/Remove** as shown in Figure 5-45.

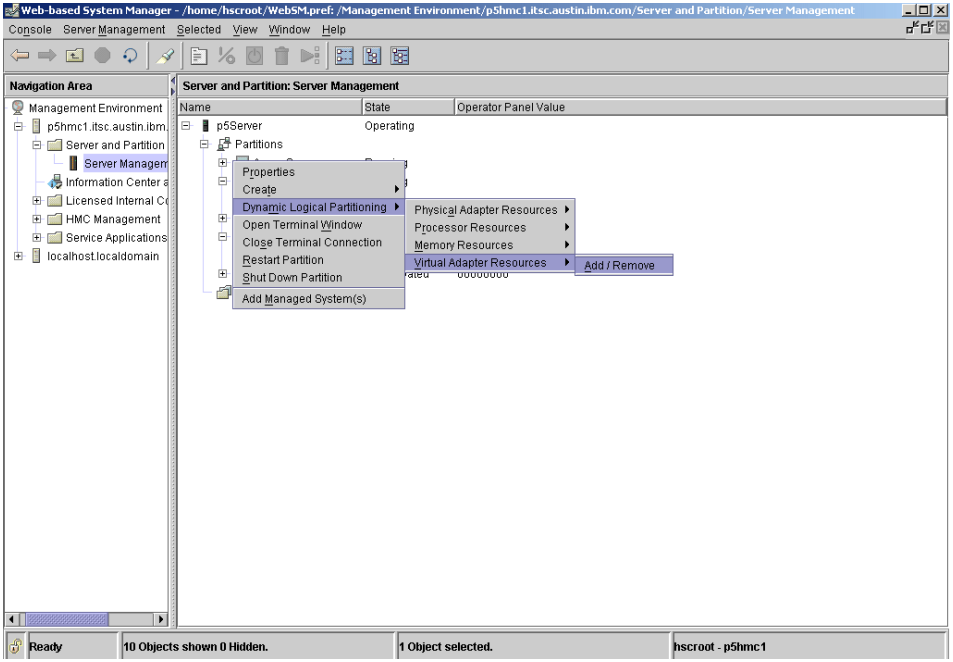

Figure 5-45 Dynamic virtual adapter resource menu

2. Select the **SCSI** radio button in the create adapters section and click **Create**, as shown in Figure 5-46.

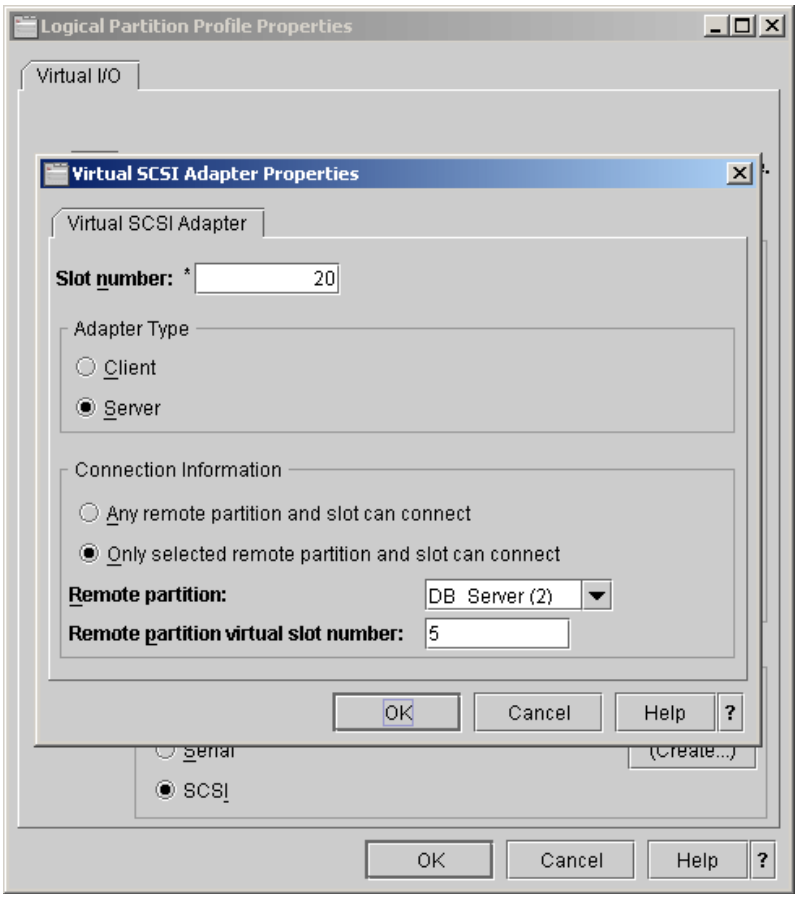

Figure 5-46 Virtual SCSI Server adapter properties

3. Select the Slot number, Adapter Type (Server in this case), and choose the Remote partition and virtual slot number of the client adapters as shown in Figure 5-46.

Important: This is a dynamic function which enables the use of the adapter without requiring a reboot. The adapter *must* also be added to each of the partition profiles in order for the device to be available to the partition on reboot. Failure to do this will result in the adapter definition being removed on reboot.

4. Follow the same procedure on the DB_Server partition. This time, however, create a client SCSI adapter as shown in Figure 5-47.

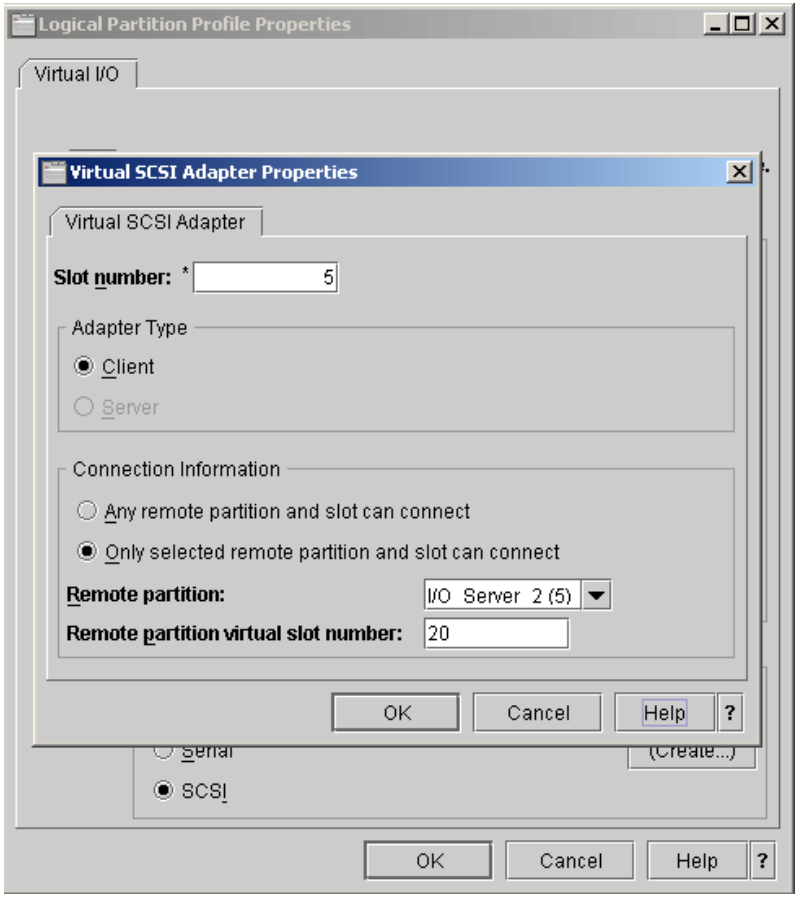

Figure 5-47 Virtual SCSI client adapter properties

5. Repeat this on the Apps_Server and Web_Server partitions.
6. On the I/O_Server_2, perform the following tasks. Refer to “Creating volume groups and logical volumes” on page 172 for the detailed procedure.
   a. Create a rootvg_clients volume group.
   b. Create a logical volume for each client partition as follows:
      - db_srv
      - apps_srv
      - web_srv

7. Map these logical volumes to the appropriate Virtual SCSI Server adapter to create a Virtual Target device. This will result in the following:
   - vdbsrv target device for DB_Server
   - vappssrv target device for Apps_Server
   - vwebsrv target device for Web_Server
8. Run the **cfgmgr** command on each client partition. A new hdisk will be available on the client partition.

```
# lspv
hdisk0          00cddedc03a34381                    rootvg          active
hdisk1          00cddedcb48b82cf                    datavg          active
hdisk2          none                                None
```

This completes the addition of a disk to a running client partition.

5.4.6 Mirroring rootvg on the client partition

Each client partition will have a newly created available hdisk served by I/O_Server_2. This disk will be used to mirror rootvg to provide physical disk and Virtual I/O Server redundancy. The procedure to perform this task on DB_Server is as follows:

1. List the available hdisks in the DB_Server partition.

```
# lspv
hdisk0          00cddedc03a34381                    rootvg          active
hdisk1          00cddedcb48b82cf                    datavg          active
hdisk2          none                                None
```

   The hdisk2 will be used to mirror rootvg in this scenario.

2. Extend the rootvg volume group as follows:

```
# extendvg -f rootvg hdisk2
0516-1254 extendvg: Changing the PVID in the ODM.
# lspv
hdisk0          00cddedc03a34381                    rootvg          active
hdisk1          00cddedcb48b82cf                    datavg          active
hdisk2          00cddedcb53b6d98                    rootvg          active
```

3. Mirror the volume group using the **mirrorvg** command.

Note: There are many different flags which can be used with this command. The following will mirror the rootvg volume group while synchronizing the disks in the background.

```
# mirrorvg -S rootvg hdisk2
......
0516-1124 mirrorvg: Quorum requirement turned off, reboot system for this
       to take effect for rootvg.
0516-1126 mirrorvg: rootvg successfully mirrored, user should perform
        bosboot of system to initialize boot records.  Then, user must
modify
        bootlist to include:  hdisk0 hdisk2.
```

4. Update the boot image file using the **bosboot** command.

```
# bosboot -a

bosboot: Boot image is 22307 512 byte blocks.
```

5. Include hdisk2 as part of the bootlist as follows:

```
# bootlist -m normal hdisk0 hdisk2
# bootlist -m normal -o
hdisk0
hdisk2
```

Note: LVM mirroring virtual disks within a client partition is no different than any other type of assigned disk.

6. Reboot the client partition as previously indicated by the **mirrorvg** command.
7. Perform these tasks on the Apps_Server and Web_Server partitions.

This completes the steps required to mirror the rootvg volume group using Virtual SCSI disk drives.

You have now completed the steps to enhance the availability of your configuration as required by scenario 2.

6

System management

This chapter provides a discussion of the following topics:

- A method to back up and restore the Virtual I/O Server
- Rebuilding the Virtual I/O Server
- Hardware Management Console changes
- Basic Partition Load Manager configuration

6.1 Backup and restore of the Virtual I/O Server

This section describes a method to back up and restore the Virtual I/O Server.

6.1.1 Backing up the Virtual I/O Server

The Virtual I/O Server command line interface provides the **backupios** command to create an installable image of the root volume group onto either a bootable tape or a multi-volume CD/DVD. The creation of an installable NIM image on a file system is provided as well. Additionally, the system's partition configuration information, including the virtual I/O devices, should be backed up on the HMC. The client data should be backed up from the client system to ensure the consistency of the data.

The **backupios** command supports the following backup devices:

- Tape
- File system
- CD
- DVD

For this example, we backed up the Virtual I/O Server using tape and file system.

Examples for CD or DVD backup are not included.

6.1.2 Backing up on tape

The following command shows the output from a backup on tape drive.

```
$ backupios -tape /dev/rmt0

Creating information file (/image.data) for rootvg..

Creating tape boot image.....

Creating list of files to back up.
Backing up 23622 files.....................
23622 of 23622 files (100%)
0512-038 mksysb: Backup Completed Successfully.

bosboot: Boot image is 26916 512 byte blocks.

bosboot: Boot image is 26916 512 byte blocks.
$
```

The result of this command is a bootable tape which allows an easy restore of the Virtual I/O Server, as shown in 6.2.1, “Restoring from tape”.

6.1.3 Backing up on a file system

The result of the **backupios** command is a backup image in **tar** format. This file will be stored in the directory specified by the **-file** flag. The output that follows shows the creation of a subdirectory called backup_loc, and the **backupios** command with its output:

```
$ mkdir /home/padmin/backup_loc
$ backupios -file /home/padmin/backup_loc
Backup in progress. This command may take several hours.

cp: /tmp/bootpkg_270554/temp-bosinst.data: A file or directory in the path name
does not exist.
$
```

The **ls** command shows that the backup was creating a **tar** file successfully.

```
$ ls /home/padmin/backup_loc
nim_resources.tar
$
```

Note: At the time of writing, development of the **backupios** command was not complete, and the error shown occurs on **cp** command. The work around is described in the restore section. Also, backup on an NFS mounted file system has not completed development.

6.2 Restoring the Virtual I/O Server

The following sections describe the restore of the Virtual I/O Server depending on your chosen backup format.

6.2.1 Restoring from tape

To restore your Virtual I/O Server from tape, boot the Virtual I/O Server partition to the SMS menu and select the tape drive as the install device. Then continue like a normal AIX installation. Figure 6-1 on page 204 shows the selection of a tape drive for restoring a previously made tape backup.

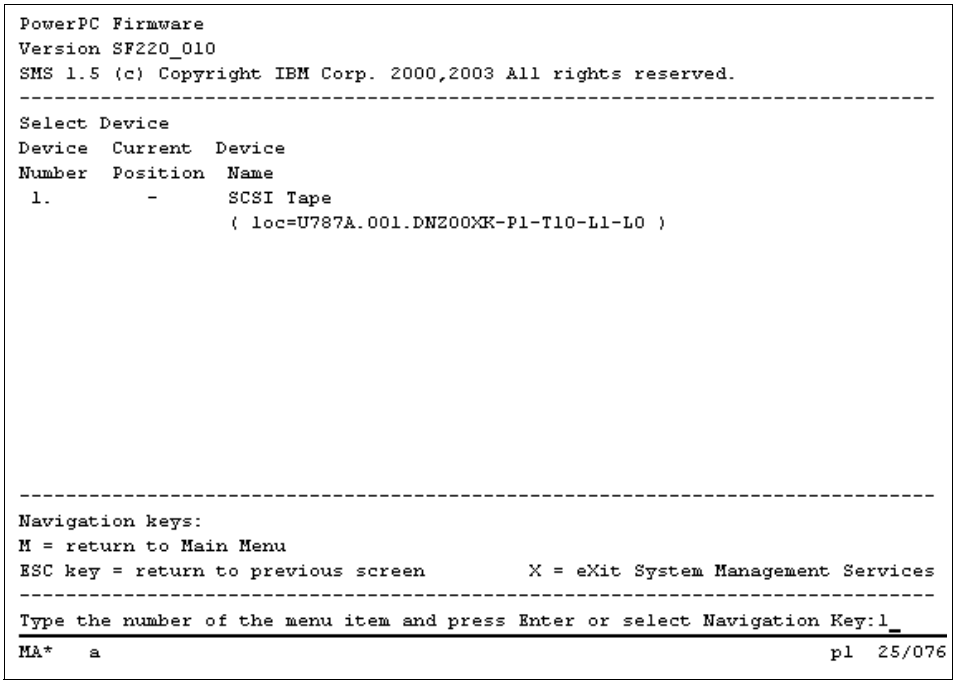

Figure 6-1 Selecting tape drive for restore of Virtual I/O Server

6.2.2 Restoring from file system

The restore of a Virtual I/O Server file system backup is done by the **installios** command of the HMC. For restore, the tar-file must be located either on the HMC or on an NFS-accessible directory. To make the tar file created with the **backupios** command accessible for restore we performed the following steps:

1. Create a directory named backup:

```
$ mkdir /home/padmin/backup
$ ls -l /home/padmin
total 8
drwxr-xr-x   2 padmin   system          256 Jul 14 09:10 backup
drwxr-xr-x   2 padmin   system          256 Jul 13 12:18 backup_loc
-rw-r--r--   1 root     system         1736 Jul 14 09:10 ioscli.log
$
```

2. Check the NFS server:

```
$ showmount server1
export list for server1:
/export/mksysb_ios (everyone)
$
```

3. Mount this directory to NFS server, server1:

```
$ mount server1:/export/mksysb_ios /home/padmin/backup
$ mount
  node       mounted        mounted over    vfs       date        options
-------- ---------------  ---------------  ------ ------------ ---------------
         /dev/hd4         /                jfs2   Jul 13 11:34 rw,log=/dev/hd8
         /dev/hd2         /usr             jfs2   Jul 13 11:34 rw,log=/dev/hd8
         /dev/hd9var      /var             jfs2   Jul 13 11:34 rw,log=/dev/hd8
         /dev/hd3         /tmp             jfs2   Jul 13 11:34 rw,log=/dev/hd8
         /dev/hd1         /home            jfs2   Jul 13 11:34 rw,log=/dev/hd8
         /proc            /proc            procfs Jul 13 11:34 rw
         /dev/hd10opt     /opt             jfs2   Jul 13 11:34 rw,log=/dev/hd8
server1  /export/mksysb_ios /home/padmin/backup nfs3   Jul 14 09:08
$
```

4. Copy the **tar** file created in “Backing up on a file system” on page 203 to the NFS mounted directory:

```
$ cp /home/padmin/backup_loc/nim_resources.tar /home/padmin/backup
$
```

Note: Because development of the backup process was not complete at the time of this writing, we unpacked the **tar** file, edited the bosinst.data file changing PROMPT= yes to PROMPT = no, and packed it again.

At this stage, the backup is ready to be restored to the Virtual I/O Server partition using the **installios** command on HMC. The restore procedure will shut down the Virtual I/O Server partition if it is still running. The following example shows the output from the restore session (several pages of output follow):

```
hscroot@p5hmc2:~> installios -s p5Server -p I/O_Server_2 -i 9.3.5.151 -S
255.255.255.0 -d 9.3.5.194:/export/mksysb_ios -m 00:0d:60:0a:59:26 -r default
-g 9.3.5.41 -P 10 -D half
nimol_config MESSAGE: No NIMOL server hostname specified, using p5hmc2 as the
default.
nimol_config MESSAGE: Added "REMOTE_ACCESS_METHOD /usr/bin/rsh" to the file
"/etc/nimol.conf"
nimol_config MESSAGE: Removed "disable = yes" from the file
"/etc/xinetd.d/tftp"
nimol_config MESSAGE: Added "disable = no" to the file "/etc/xinetd.d/tftp"
Shutting down xinetd:
done
Starting INET services. (xinetd)
done
nimol_config MESSAGE: Removed "SYSLOGD_PARAMS=" from the file
"/etc/sysconfig/syslog"
nimol_config MESSAGE: Added "SYSLOGD_PARAMS=-r " to the file
"/etc/sysconfig/syslog"
nimol_config MESSAGE: Removed "local2,local3.*  -/var/log/localmessages" from
the file "/etc/syslog.conf"
```

```
nimol_config MESSAGE: Added "local3.*  -/var/log/localmessages" to the file
"/etc/syslog.conf"
nimol_config MESSAGE: Added "local2.* /var/log/nimol.log" to the file
"/etc/syslog.conf"
Shutting down syslog services
done
Starting syslog services
done
nimol_config MESSAGE: Created /etc/dhcpd.conf.
Starting DHCP server [chroot]
done
nimol_config MESSAGE: Executed /etc/init.d/dhcpd start.
nimol_config MESSAGE: Successfully configured NIMOL.
nimol_config MESSAGE: target directory: /dump/ioserver_res
nimol_config MESSAGE: Executed /usr/sbin/iptables -I INPUT 1 -s 9.3.5.194 -j
ACCEPT.
nimol_config MESSAGE: source directory: /mnt/nimol
nimol_config MESSAGE: Checking /mnt/nimol/nim_resources.tar for existing
resources.
nimol_config MESSAGE: Executed /usr/sbin/iptables -D INPUT -s 9.3.5.194 -j
ACCEPT.
nimol_config MESSAGE: Added "/dump/ioserver_res *(rw,insecure,no_root_squash)"
to the file "/etc/exports"
nimol_config MESSAGE: Successfully created "ioserver_res".
nimol_install WARNING: The hostname "ioserver" is in use.
nimol_install MESSAGE: Added "CLIENT ioserver" to the file "/etc/nimol.conf"
nimol_install MESSAGE: Added "host ioserver" to the file "/etc/dhcpd.conf"
Shutting down DHCP server
done
Starting DHCP server [chroot]
done
nimol_install MESSAGE: Created /tftpboot/ioserver.
nimol_install MESSAGE: Executed /sbin/arp -s ioserver 00:0d:60:0a:59:26 -t
ether.
nimol_install MESSAGE: Executed /usr/sbin/iptables -I INPUT 1 -s ioserver -j
ACCEPT.
nimol_install MESSAGE: Created /dump/ioserver_res/scripts/ioserver.script.
nimol_install MESSAGE: Created /tftpboot/ioserver.info.
nimol_install MESSAGE: Successfully setup ioserver for a NIMOL install
# Connecting to I/O_Server_2.
# Checking for power off.
# Power off the node.
# Wait for power off.
# Power off complete.
# Power on I/O_Server_2 to Open Firmware.
# Power on complete.
# Network booting install adapter.
# bootp sent over network.
# Network boot proceeding, lpar_netboot is exiting.
```

```
# Finished.
Tue Jul 13 10:21:14 2004
-----------/var/log/nimol.log :---------------
Jul 13 10:18:03 ioserver nimol:,-S,shutdown,ioserver,

Tue Jul 13 10:21:24 2004
-----------/var/log/nimol.log :---------------
Jul 13 10:21:24 ioserver nimol:,info=LED 610: mount  -r
p5hmc2:/dump/ioserver_res/SPOT/usr /SPOT/usr,

Tue Jul 13 10:21:24 2004
-----------/var/log/nimol.log :---------------
Jul 13 10:21:24 ioserver nimol:,info=,

Tue Jul 13 10:21:24 2004
-----------/var/log/nimol.log :---------------
Jul 13 10:21:24 ioserver nimol:,-S,booting,ioserver,

Tue Jul 13 10:21:24 2004
-----------/var/log/nimol.log :---------------
Jul 13 10:21:24 ioserver nimol:,info=LED 610: mount
p5hmc2:/dump/ioserver_res/mksysb /NIM_BOS_IMAGE,

Tue Jul 13 10:21:24 2004
-----------/var/log/nimol.log :---------------
Jul 13 10:21:24 ioserver nimol:,info=LED 610: mount
p5hmc2:/dump/ioserver_res/bosinst.data /NIM_BOSINST_DATA,

Tue Jul 13 10:21:24 2004
-----------/var/log/nimol.log :---------------
Jul 13 10:21:24 ioserver nimol:,info=,

Tue Jul 13 10:21:46 2004
-----------/var/log/nimol.log :---------------
Jul 13 10:21:46 ioserver nimol:,-R,success,ioserver,

Tue Jul 13 10:21:46 2004
-----------/var/log/nimol.log :---------------
Jul 13 10:21:46 ioserver nimol:,info=extract_data_files,

Tue Jul 13 10:21:47 2004
-----------/var/log/nimol.log :---------------
Jul 13 10:21:47 ioserver nimol:,info=query_disks,

Tue Jul 13 10:21:47 2004
-----------/var/log/nimol.log :---------------
Jul 13 10:21:47 ioserver nimol:,info=extract_diskette_data,

Tue Jul 13 10:21:48 2004
```

```
-----------/var/log/nimol.log :---------------
Jul 13 10:21:48 ioserver nimol:,info=setting_console,

Tue Jul 13 10:21:48 2004
-----------/var/log/nimol.log :---------------
Jul 13 10:21:48 ioserver nimol:,info=initialization,

Tue Jul 13 10:21:52 2004
-----------/var/log/nimol.log :---------------
Jul 13 10:21:52 ioserver nimol:,info=verifying_data_files,

Tue Jul 13 10:21:58 2004
-----------/var/log/nimol.log :---------------
Jul 13 10:21:58 ioserver nimol:,info=,

Tue Jul 13 10:22:01 2004
-----------/var/log/nimol.log :---------------
Jul 13 10:21:59 ioserver nimol:,info=BOS install 1% complete : Making boot
logical volume.,

Tue Jul 13 10:22:01 2004
-----------/var/log/nimol.log :---------------
Jul 13 10:22:00 ioserver nimol:,info=BOS install 2% complete : Making paging
logical volumes.,

Tue Jul 13 10:22:03 2004
-----------/var/log/nimol.log :---------------
Jul 13 10:22:02 ioserver nimol:,info=BOS install 3% complete : Making logical
volumes.,

Tue Jul 13 10:22:10 2004
-----------/var/log/nimol.log :---------------
Jul 13 10:22:10 ioserver nimol:,info=BOS install 4% complete : Forming the jfs
log.,

Tue Jul 13 10:22:12 2004
-----------/var/log/nimol.log :---------------
Jul 13 10:22:12 ioserver nimol:,info=BOS install 5% complete : Making file
systems.,

Tue Jul 13 10:22:13 2004
-----------/var/log/nimol.log :---------------
Jul 13 10:22:13 ioserver nimol:,info=BOS install 6% complete : Mounting file
systems.,

Tue Jul 13 10:22:14 2004
-----------/var/log/nimol.log :---------------
Jul 13 10:22:14 ioserver nimol:,info=BOS install 7% complete,
```

```
Tue Jul 13 10:22:14 2004
-----------/var/log/nimol.log :---------------
Jul 13 10:22:14 ioserver nimol:,info=BOS install 7% complete : Restoring base
operating system.,

Tue Jul 13 10:22:19 2004
-----------/var/log/nimol.log :---------------
Jul 13 10:22:19 ioserver nimol:,info=BOS install 8% complete : 2% of mksysb
data restored.,
...
...
...
Tue Jul 13 10:31:22 2004
-----------/var/log/nimol.log :---------------
Jul 13 10:31:22 ioserver nimol:,info=BOS install 79% complete : 96% of mksysb
data restored.,

Tue Jul 13 10:31:23 2004
-----------/var/log/nimol.log :---------------
Jul 13 10:31:23 ioserver nimol:,info=BOS install 82% complete,

Tue Jul 13 10:31:23 2004
-----------/var/log/nimol.log :---------------
Jul 13 10:31:23 ioserver nimol:,info=BOS install 82% complete : Initializing
disk environment.,

Tue Jul 13 10:31:31 2004
-----------/var/log/nimol.log :---------------
Jul 13 10:31:31 ioserver nimol:,info=BOS install 83% complete : Over mounting
/.,

Tue Jul 13 10:31:51 2004
-----------/var/log/nimol.log :---------------
Jul 13 10:31:51 ioserver nimol:,info=BOS install 84% complete,

Tue Jul 13 10:31:51 2004
-----------/var/log/nimol.log :---------------
Jul 13 10:31:51 ioserver nimol:,info=BOS install 85% complete : Copying Cu* to
disk.,

Tue Jul 13 10:31:51 2004
-----------/var/log/nimol.log :---------------
Jul 13 10:31:51 ioserver nimol:,info=BOS install 86% complete,

Tue Jul 13 10:31:57 2004
-----------/var/log/nimol.log :---------------
Jul 13 10:31:56 ioserver nimol:,info=BOS install 86% complete : Initializing
dump device.,
```

```
Tue Jul 13 10:31:59 2004
-----------/var/log/nimol.log :---------------
Jul 13 10:31:58 ioserver nimol:,info=recover_device_attributes,

Tue Jul 13 10:31:59 2004
-----------/var/log/nimol.log :---------------
Jul 13 10:31:58 ioserver nimol:,info=bosboot,

Tue Jul 13 10:31:59 2004
-----------/var/log/nimol.log :---------------
Jul 13 10:31:58 ioserver nimol:,info=BOS install 87% complete : Creating boot
image.,

Tue Jul 13 10:32:15 2004
-----------/var/log/nimol.log :---------------
Jul 13 10:32:15 ioserver nimol:,info=BOS install 100% complete,

nimol_install MESSAGE: Removed "host ioserver" from the file "/etc/dhcpd.conf"
Shutting down DHCP server
done
Starting DHCP server [chroot]
done
nimol_install MESSAGE: Removed /tftpboot/ioserver.
nimol_install MESSAGE: Executed /sbin/arp -d ioserver.
nimol_install MESSAGE: Executed /usr/sbin/iptables -D INPUT -s ioserver -j
ACCEPT.
nimol_install MESSAGE: Removed /tftpboot/ioserver.info.
nimol_install MESSAGE: Removed /dump/ioserver_res/scripts/ioserver.script.
nimol_install MESSAGE: Removed "CLIENT ioserver" from the file
"/etc/nimol.conf"
nimol_config MESSAGE: Removed "/dump/ioserver_res
*(rw,insecure,no_root_squash)" from the file "/etc/exports"
nimol_config MESSAGE: Executed /usr/sbin/exportfs -ua.
nimol_config MESSAGE: Executed /usr/sbin/exportfs -a.
nimol_config MESSAGE: Removed /tftpboot/ioserver_res.chrp.mp.ent.
nimol_config MESSAGE: Removed /dump/ioserver_res.
nimol_config MESSAGE: Removed "LABEL ioserver_res" from the file
"/etc/nimol.conf"
nimol_config MESSAGE: Unconfiguring the NIMOL server...
nimol_config MESSAGE: Removed "disable = no" from the file "/etc/xinetd.d/tftp"
nimol_config MESSAGE: Added "disable = yes" to the file "/etc/xinetd.d/tftp"
Shutting down xinetd:
done
Starting INET services. (xinetd)
done
nimol_config MESSAGE: Removed "SYSLOGD_PARAMS="-r "" from the file
"/etc/sysconfig/syslog"
nimol_config MESSAGE: Added "SYSLOGD_PARAMS="" to the file
"/etc/sysconfig/syslog"
```

```
nimol_config MESSAGE: Removed "local2.*   /var/log/nimol.log" from the file
"/etc/syslog.conf"
nimol_config MESSAGE: Removed "local3.*  -/var/log/localmessages" from the file
"/etc/syslog.conf"
nimol_config MESSAGE: Added "local2,local3.*  -/var/log/localmessages" to the
file "/etc/syslog.conf"
Shutting down syslog services
done
Starting syslog services
done
nimol_config MESSAGE: Removed /etc/dhcpd.conf.
Shutting down DHCP server
done
nimol_config MESSAGE: Executed /etc/init.d/dhcpd stop.
nimol_config MESSAGE: Removed /var/tmp/nimol_original.
nimol_config MESSAGE: Removed /etc/nimol.conf.
nimol_config MESSAGE: Successfully unconfigured NIMOL.
hscroot@p5hmc2:~>
```

The Virtual I/O Server partition will reboot to finish the restore procedure.

6.3 Rebuilding the Virtual I/O Server

This section describes what to do if there are no valid backup devices or backup images. In this case you must install a new Virtual I/O Server.

In the following discussion, we assume that the partition definitions of the Virtual I/O Server and of all clients on the HMC are still available. We describe how we rebuilt our configuration of network and SCSI.

In addition to the regular backups using the **backupios** command, we recommend documenting the configuration of:

- Network settings
  Commands: **netstat -state, netstat -routinfo,**
  netstat -dev *Device* -attr
- All physical and logical volumes SCSI devices
  Commands: **lspv, lsvg, lsvg -lv *VolumeGroup***
- All physical and logical adapters
  Commands: **lsdev -type adapter**
- The mapping between physical and logical devices and virtual devices
  Commands: **lsmap -all, lsmap -all -net**

With this information you are able to reconfigure your Virtual I/O Server manually. In the following sections we describe the commands we needed to get the

necessary information and the commands that rebuilt the configuration. The important information from the command outputs is highlighted. Depending on your environment, the commands may differ from those shown as examples.

To start rebuilding the Virtual I/O Server, you must know which disks are used for the Virtual I/O Server itself and for any assigned volume groups for virtual I/O.

The **lspv** command shows us that the Virtual I/O Server was installed on hdisk0. The first step is to install the new Virtual I/O Server from the installation media onto disk hdisk0. This command can be run from diag, or another AIX environment.

```
$ lspv
hdisk0          00cddedc01300ed3                    rootvg          active
hdisk1          00cddedc143815fb                    None
hdisk2          00cddedc4d209163                    client_disks    active
hdisk3          00cddedc4d2091f8                    datavg          active
```

See 5.3.3, “Virtual I/O Server software installation” for the installation procedure. The further rebuild of the Virtual I/O Server is done in two steps:

1. Rebuild the SCSI configuration.
2. Rebuild the network configuration.

6.3.1 Rebuild the SCSI configuration

The **lspv** command also shows us that there are two additional volume groups located on the Virtual I/O Server (client_disks and datavg).

```
$ lspv
hdisk0          00cddedc01300ed3                    rootvg          active
hdisk1          00cddedc143815fb                    None
hdisk2          00cddedc4d209163                    client_disks    active
hdisk3          00cddedc4d2091f8                    datavg          active
```

The following command imports this information into the new Virtual I/O Server system’s ODM:

```
importvg -vg client_disks hdisk2
importvg -vg datavg hdisk3
```

With the **lsmap -all** command we look to the mapping between the logical and physical volumes and the Virtual SCSI Server adapters:

```
$ lsmap -all
SVSA            Physloc                                     Client Partition
ID
--------------- ------------------------------------------- ------------------
vhost0          U9111.520.10DDEDC-V4-C30                    0x00000000
```

```
VTD                   vtscsi2
LUN                   0x8100000000000000
Backing device        datalv
Physloc

SVSA            Physloc                                     Client Partition
ID
--------------- ------------------------------------------- ------------------
vhost2          U9111.520.10DDEDC-V4-C10                    0x00000006

VTD                   vtscsi0
LUN                   0x8100000000000000
Backing device        rootvg_ztest0
Physloc

VTD                   vtscsi1
LUN                   0x8200000000000000
Backing device        hdisk1
Physloc               U787A.001.DNZ00XY-P1-T10-L4-L0
```

Virtual SCSI Server Adapter vhost0 (defined on slot 30 in HMC) is mapped to Logical Volume datalv by Virtual Target Device vtscsi2.

Virtual SCSI Server Adapter vhost2 has two Virtual Target Devices, vtscsi0 and vtscsi1. They are mapping Logical Volume rootvg_ztest0 and Physical Volume hdisk1 to vhost2 (is defined on slot 10 in HMC).

The following commands are used to create our needed Virtual Target Devices:

```
mkvdev -vdev datalv -vadapter vhost0
mkvdev -vdev rootvg_ztest0 -vadapter vhost2
mkvdev -vdev hdisk1 -vadapter vhost2
```

Note: The names of the Virtual Target Devices are generated automatically, except when you define a name using the **-dev** flag of **mkvdev** command.

6.3.2 Rebuild network configuration

After successfully rebuilding the SCSI configuration we now are going to rebuild the network configuration:

The **netstat -state** command shows us that en2 is the only active network adapter.

```
$ netstat -state
Name  Mtu   Network     Address            Ipkts Ierrs    Opkts Oerrs  Coll
en2   1500  link#2      0.d.60.a.58.a4        2477     0      777     0     0
```

```
en2   1500  9.3.5       9.3.5.147             2477     0      777      0      0
lo0   16896 link#1                             153      0      158      0      0
lo0   16896 127         127.0.0.1             153      0      158      0      0
lo0   16896 ::1                               153      0      158      0      0
```

With the **lsmap -all -net** command, we determine that ent2 is defined as a shared Ethernet adapter mapping physical adapter ent0 to virtual adapter ent1.

```
$ lsmap -all -net
SVEA   Physloc
------ --------------------------------------------
ent1   U9111.520.10DDEDC-V4-C2-T1

SEA                 ent2
Backing device      ent0
Physloc             U787A.001.DNZ00XY-P1-C2-T1
```

The information for the default gateway address is provided by the **netstat -routinfo** command.

```
$ netstat -routinfo
Routing tables
Destination      Gateway           Flags     Wt  Policy  If   Cost Config_Cost

Route Tree for Protocol Family 2 (Internet):
default          9.3.5.41          UG         1    -     en2     0         0
9.3.5.0          9.3.5.147         UHSb       1    -     en2     0         0 =>
9.3.5/24         9.3.5.147         U          1    -     en2     0         0
9.3.5.147        127.0.0.1         UGHS       1    -     lo0     0         0
9.3.5.255        9.3.5.147         UHSb       1    -     en2     0         0
127/8            127.0.0.1         U          1    -     0       0         0
```

To list the subnet mask, we use the **lsdev -dev en2 -attr** command.

```
$ lsdev -dev en2 -attr
attribute     value         description
user_sett
able

alias4                      IPv4 Alias including Subnet Mask           True
alias6                      IPv6 Alias including Prefix Length         True
arp           on            Address Resolution Protocol (ARP)          True
authority                   Authorized Users                           True
broadcast                   Broadcast Address                          True
mtu           1500          Maximum IP Packet Size for This Device     True
netaddr       9.3.5.147     Internet Address                           True
netaddr6                    IPv6 Internet Address                      True
netmask       255.255.255.0 Subnet Mask                                True
prefixlen                   Prefix Length for IPv6 Internet Address    True
```

```
remmtu          576           Maximum IP Packet Size for REMOTE Networks True
rfc1323                       Enable/Disable TCP RFC 1323 Window Scaling True
security        none           Security Level                             True
state           up             Current Interface Status                   True
tcp_mssdflt                    Set TCP Maximum Segment Size               True
tcp_nodelay                    Enable/Disable TCP_NODELAY Option          True
tcp_recvspace                  Set Socket Buffer Space for Receiving      True
tcp_sendspace                  Set Socket Buffer Space for Sending        True
$
```

The last information we need is the default virtual adapter and the default PVID for the shared Ethernet adapter. This is shown by the **lsdev -dev ent2 -attr** command.

```
$ lsdev -dev ent2 -attr
attribute     value description                                              user_settable

pvid          1     PVID to use for the SEA device                                    True
pvid_adapter  ent1  Default virtual adapter to use for non-VLAN-tagged packets       True
real_adapter  ent0  Physical adapter associated with the SEA                         True
virt_adapters ent1  List of virtual adapters associated with the SEA (comma separated) True
$
```

The following commands re-created our network configuration.

```
mkvdev -sea ent0 -vadapter ent1 -default ent1 -defaultid 1
mktcpip -hostname p51iosrv2 -inetaddr 9.3.5.147 -interface en2 -start -netmask
255.255.255.0 -gateway 9.3.5.41
```

These steps complete the basic rebuilding of the Virtual I/O Server.

6.4 Basic Partition Load Manager configuration

This section provides a short overview of the installation and configuration of the Partition Load Manager (PLM) and shows an example of how PLM adjusts capacity entitlement. Only basic tuning values and management of CPU capacity for one shared processor partition group is provided.

An introduction to PLM can be found in 3.7, “Partition Load Manager introduction” .

The scenario used in Chapter 5, “AIX and Virtual I/O Server configuration scenarios” on page 137 is used again in this section. Partition Load Manager is set up on the NIM server with the hostname server1. The NIM server is an external system and runs with the latest level of AIX 5L Version 5.3.

6.4.1 Installation and configuration of the Partition Load Manager

Setting up Partiton Load Manager consists of the following steps:

- Install SSH and PLM software.
- Configure SSH for communication between the PLM and the HMC without prompting for a password.
- Create a policy file.
- Set up the client partition to communicate with the PLM over RMC.

Installation of SSH and PLM software

For the installation of the SSH software, we used the following filesets:

- SSL filesets selected as prerequisites using Rpm installation procedure:
  - openssl-0.9.6m-1.aix5.1.ppc.rpm
  - openssl-devel-0.9.6m-1.aix5.1.ppc.rpm
  - openssl-doc-0.9.6m-1.aix5.1.ppc.rpm
- OpenSSH filesets used for the installation using SMIT:
  - openssh.base
  - openssh.license
  - openssh.man.man

For the installation of PLM, use the following filesets and SMIT:

- plm.license
- plm.server.rte
- plm.sysmgt.websm

Configuration of SSH

PLM must be able to remotely execute commands securely on the HMC without being prompted for a password.

It is assumed that the filesets discussed in the previous section are already installed.

Use the **ssh-keygen** command to generate a public key. It then must be copied to the .ssh directory of the HMC user. In our example we use the hscroot user.

On the PLM server command line enter the following command:

```
# ssh-keygen -t rsa
Generating public/private rsa key pair.
Enter file in which to save the key (//.ssh/id_rsa):
Created directory '//.ssh'.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in //.ssh/id_rsa.
Your public key has been saved in //.ssh/id_rsa.pub.
The key fingerprint is:
14:98:50:3f:15:a1:fb:9f:3a:0b:0b:18:1a:eb:c8:33 root@applsrv
```

After using the **ssh-keygen** command, you find the id_rsa.pub file in the .ssh directory. The contents of this directory are the following:

```
# ls -al
total 40
drwx------   2 root     system          512 Jun 28 16:54 .
drwxr-xr-x  22 root     system         1024 Jun 28 16:53 ..
-rw-------   1 root     system          883 Jun 28 16:54 id_rsa
-rw-r--r--   1 root     system          222 Jun 28 16:54 id_rsa.pub
-rw-r--r--   1 root     system          226 Jun 28 16:53 known_hosts
```

Then add the content of the id_rsa.pub file to the authorized_keys2 file on the HMC.

Because of the restricted shell on the HMC, we are not allowed to redirect the output to the authorized_keys2 file. Therefore, we copy an existing authorized_keys2 file from the HMC using the **scp**, as shown in the following:

```
# scp hscroot@p5hmc2:.ssh/authorized_keys2 /.ssh/authorized_keys2
hscroot@p5hmc2's password:
authorized_keys2                              100%  222     0.2KB/s   00:00
```

Add the public key using the **cat** command, as shown in the following:

```
# cat id_rsa.pub >> authorized_keys2
# ls -al
total 48
drwx------   2 root     system          512 Jul 03 16:50 .
drwxr-xr-x  25 root     system         1024 Jul 02 15:33 ..
-rw-r--r--   1 root     system          444 Jul 03 16:52 authorized_keys2
-rw-------   1 root     system          883 Jun 28 16:54 id_rsa
-rw-r--r--   1 root     system          222 Jun 28 16:54 id_rsa.pub
-rw-r--r--   1 root     system          452 Jun 30 13:53 known_hosts
```

After adding the key to the file, use the **scp** command to copy the authorized_keys2 file back to the HMC.

```
# scp /.ssh/authorized_keys2 hscroot@p5hmc2:.ssh/authorized_keys2
hscroot@p5hmc2's password:
authorized_keys2                              100%  444     0.4KB/s   00:00
```

To verify the SSH configuration, test whether you receive the output of the **ls** command without being prompted for a password. On the PLM system enter:

```
# ssh hscroot@p5hmc2 ls
WebSM.pref
websm.log
websm.script
```

If SSH is not working, ensure that you checked the “Enable remote command execution” on the HMC configuration screen.

You also have to check whether the new HMC firewall function allows incoming SSH traffic. On the Web-based System Manager, select HMC Configuration in the HMC Management menu and choose Customize Network Settings.

Choose the **LAN Adapters** tab on the Customize Network Settings panel shown in Figure 6-2.

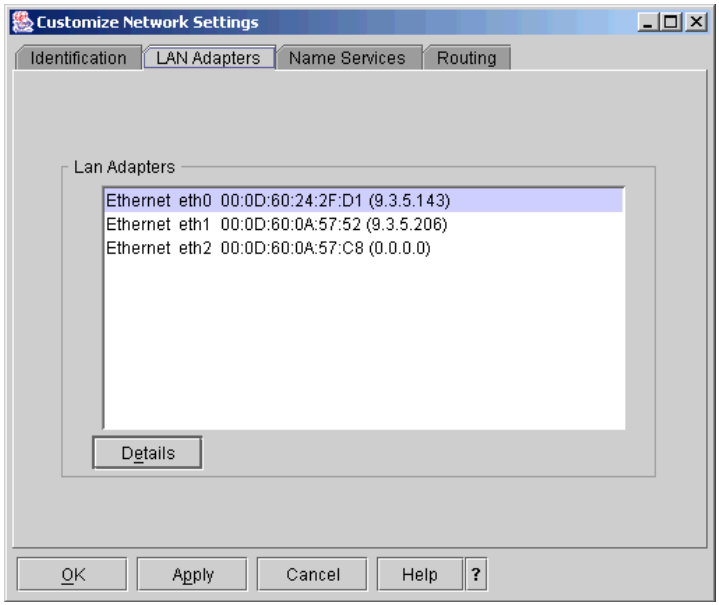

Figure 6-2 Customize Network Settings panel

Select the configured network adapter and click **Details**. Choose the **Firewall** tab on the LAN Adapter Details panel shown in Figure 6-3.

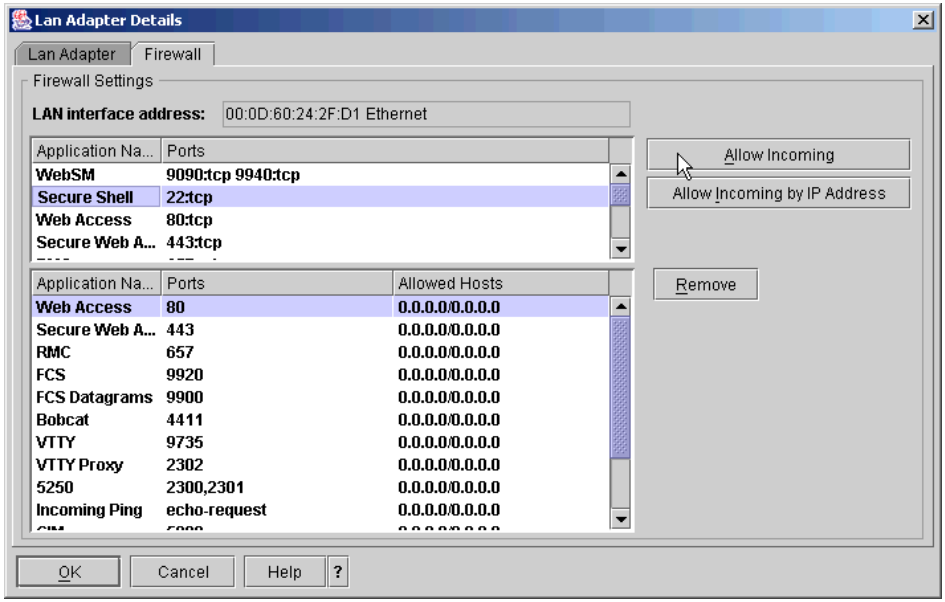

Figure 6-3 Lan Adapter Details panel for changing Firewall settings

Select the Secure Shell entry and click the **Allow Incoming** button to allow ssh access for all incoming connections or click the **Allow Incoming by IP Address** button to specify dedicated hosts to connect using the SSH connection.

Creating a policy file

The next task is to create a policy file and set up management of the logical partition (client partitions) using the Web-based System Manager, shown using an Xwindows connection.

Figure 6-4 shows the initial configuration screen of the Partition Load Manager.

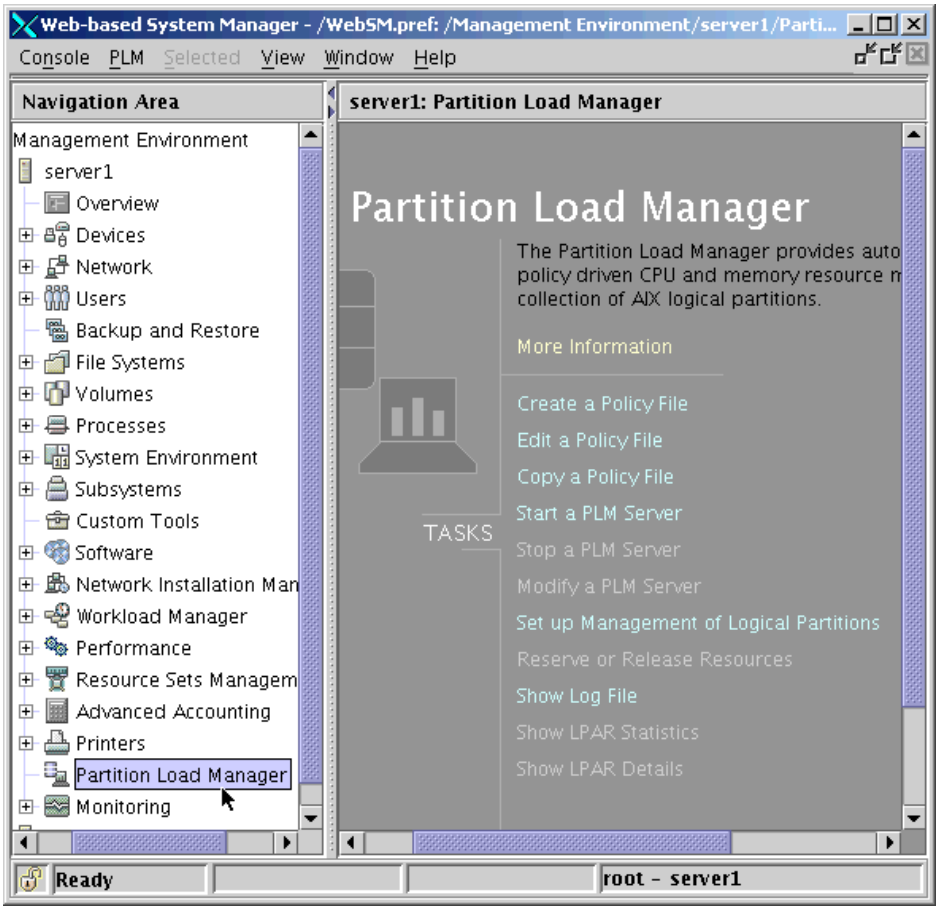

Figure 6-4 Partition Load Manager interface

We start by creating the policy file. You can set up the management of the client partitions using the policy file.

Figure 6-5 shows the **General** tab of the Create Policy File panel. Here you can choose the name and the location of the policy file. By default it is stored under /etc/plm/policies.

In this example, the author had not completed entering the file name, just the path. Make sure you complete the entry with a fully qualified file name, otherwise the policy file cannot be created.

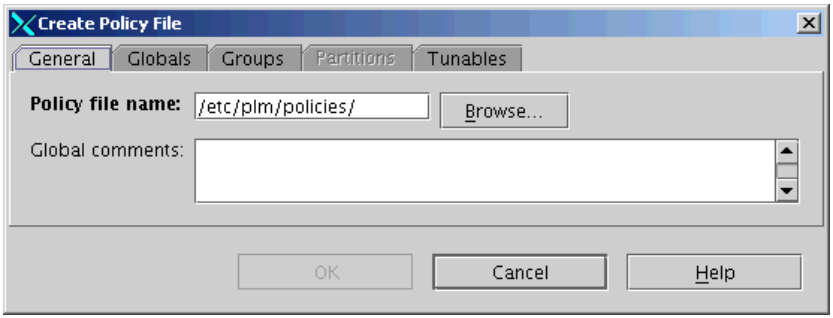

Figure 6-5 General tab of the Create Policy File panel

In the **Global** tab shown in Figure 6-6, fill in the hostname of your HMC, the HMC user name, and the name of your managed system.

You can use the **`lssyscfg -r sys -F name`** command on the HMC command line to find out the name of your managed system.

The HMC command wait field is used to adjust the time out value of an HMC command. The default value is 5 minutes.

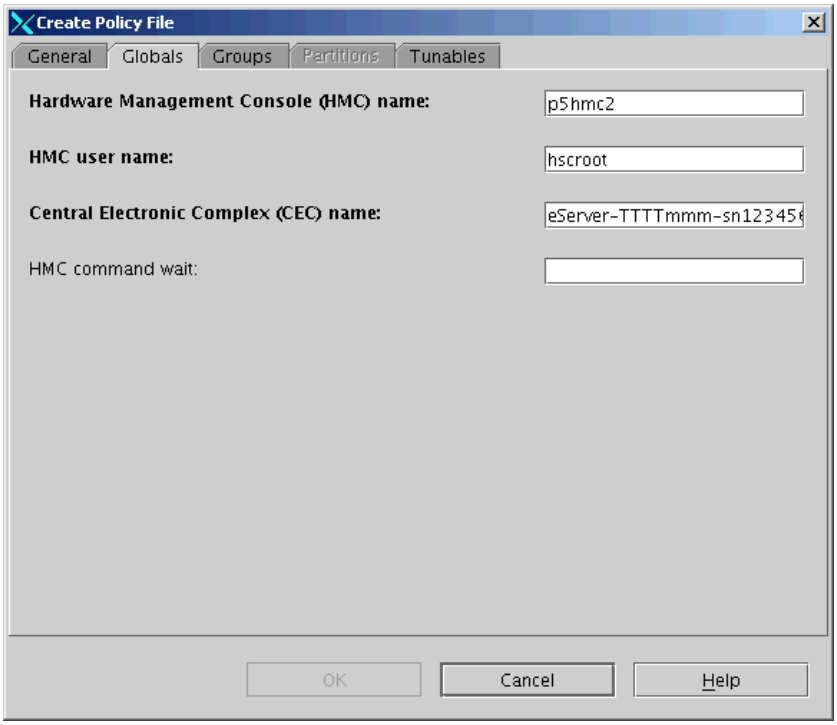

Figure 6-6 Global tab of the Create Policy File panel

Next, define the PLM groups. The **Group** tab allows you to create multiple groups of partitions which are managed independently. At least one group must be defined. All partitions in a group must have the same processor type.

In this example, all partitions are in the same group and use the shared processor type.

We are only interested in the management of the processor capacity so we only checked the CPU management check box. Figure 6-7 shows the definition of the created group. We set the Maximum CPU capacity to 1.3 to restrict the processor resources for this partition group.

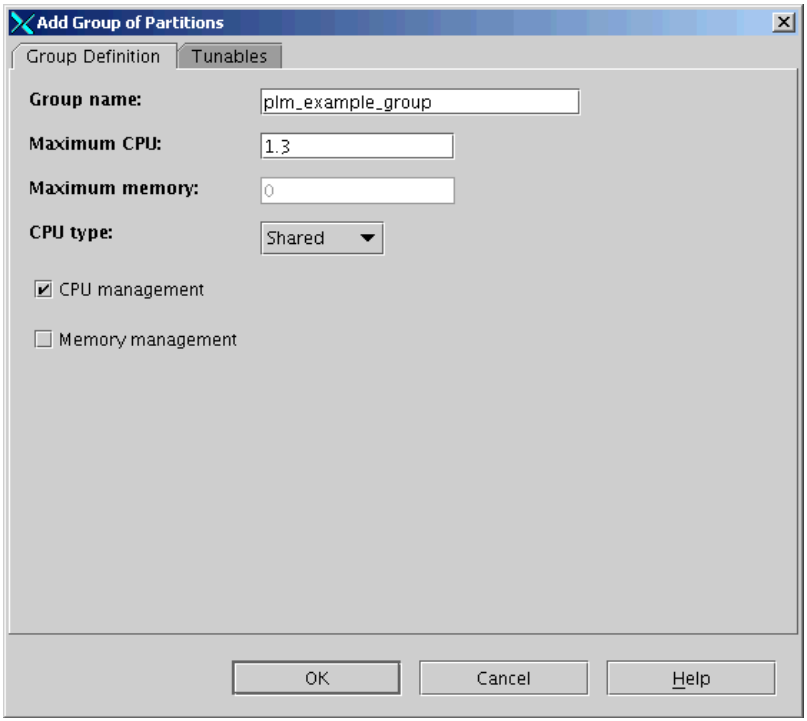

Figure 6-7 Add Group of Partitions panel

On the **Tunable** tab, set the tunable parameters to be applied to all members of the group. We chose to define the tunable values on a per partition basis so this panel is not needed. The default values are chosen in this case.

After defining at least one group, define partitions by selecting the **Partition** tab of the Create Poilcy File panel. In the Partition Definition panel shown in Figure 6-8, define the Partition name and select the group to which it belongs.

Important: In the Partition name field you have to provide the hostname of the partition, not the partition name defined on the HMC. Use the hostname the way it is provided in the RMC communication (fully qualified hostname versus short hostname).

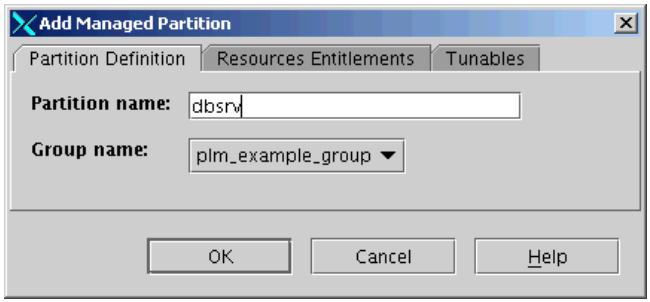

Figure 6-8 Add Managed Partition

On the next tab, specify the **Resource Entitlements** of your partition. When the values are not specified as shown in Figure 6-9 on page 226 they are obtained from the minimum, desired, and maximum HMC partition definition. These minimum and maximum values are strictly enforced at the HMC and cannot change while the partition profile is active.

You can also specify different Minimum, Guaranteed, and Maximum CPU values for each partition in the PLM policy, but these must be within the boundaries specified in the active profile on the HMC. If the minimum and maximum values specified in the PLM policy do not represent a subset of the range specified by the active profile, PLM will use the intersection of the two as the managed range. If the intersection is empty for any partition, that partition will not be managed.

For example, define the following on the HMC partition profile:

- Maximum processing units 2.0
- Desired processing units 0.8
- Minimum processing units 0.3

In the PLM policy file you define a Maximum CPU of 1.2. PLM then takes the maximum boundary of 1.2. The benefit from this definition is that you can control the maximum value without shutting down and reactivating your partition. If you need to set the maximum up to 2.0 you just change the PLM profile.

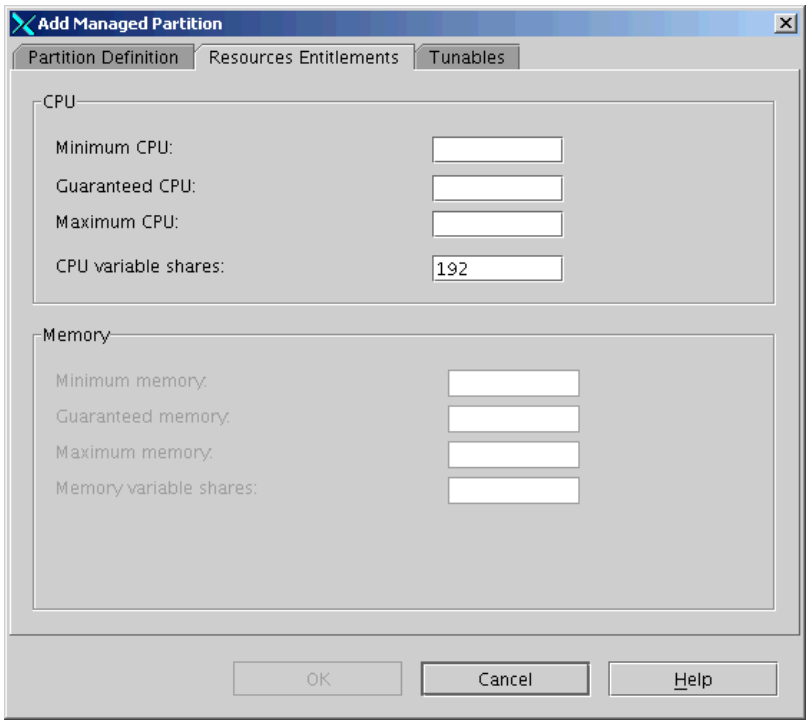

Figure 6-9 Resources Entitlement tab in the Add Managed Partition panel

You can also specify the CPU variable shares. If you do not specify this value, the default value is 1. This is the PLM managed weight for the partition and is analogous to the HMC defined weight for uncapped partitions. If specified for an uncapped partition, PLM also sets the HMC defined weight to this value.

Figure 6-10 shows an overview of all defined partitions and their specified CPU variable shares.

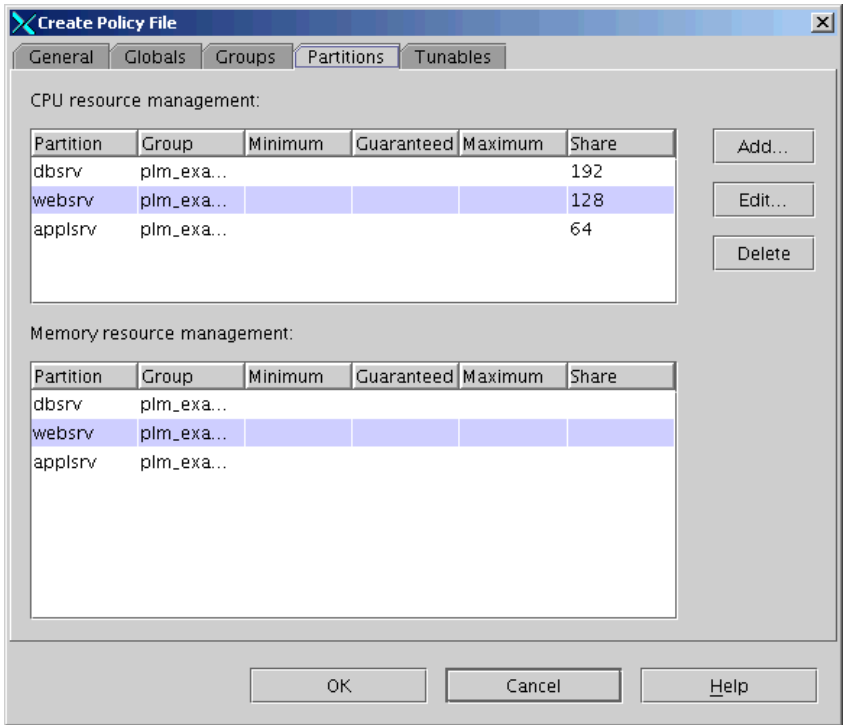

Figure 6-10 Overview of all defined partitions

The **Tunables** tab, shown in Figure 6-11, shows the tunables you can define per partition. You also see the default and group tunables you can specify. All tunables have default values which are used if no others are specified. In the policy, the tunables can be specified:

- Globally (apply to all partitions)
- At the group level (apply to all partitions in the group)
- At the partition level

The value specified at the lowest level takes precedence.

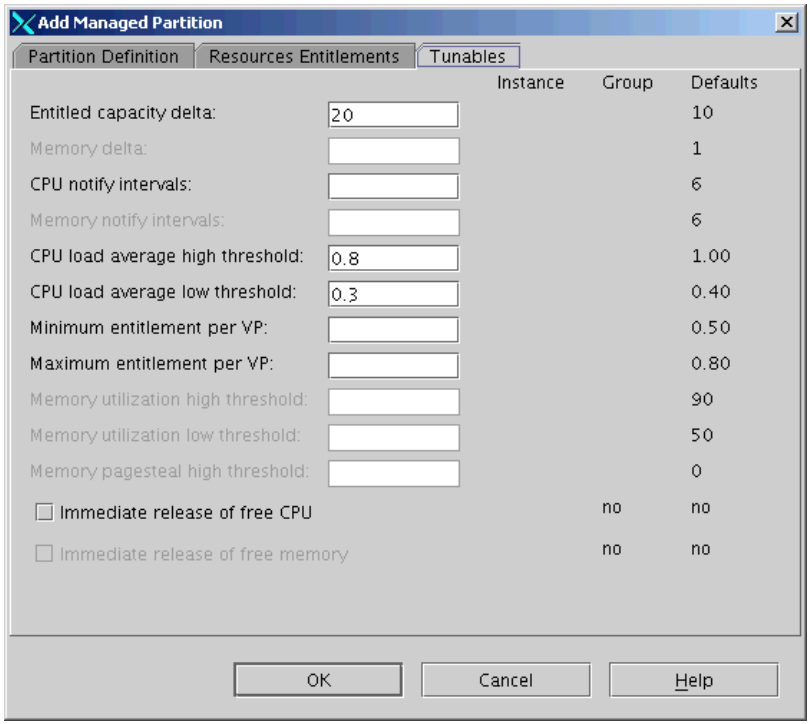

Figure 6-11 Tunabales for partitions

Table 6-1 on page 229 describes the CPU-related tunables we defined.

Table 6-1 Explanation of CPU-related tunables

Tunable	Min	Default	Max	Description
Entitled capacity delta	1	10	100	The amount of CPU entitled capacity to add or remove from a shared processor partition. The value specifies the percent of the partition's current entitled capacity to add or remove.
CPU notify intervals	1	6	100	The number of 10 second sample periods that a CPU-related sample must cross a threshold before the reallocation will take effect.
CPU load average high threshold	0.1	1.0	10.0	The processor load average high threshold value. A partition with a load average above this value is considered to need more processor capacity.
CPU load average low threshold	0.1	0.5	1.0	The CPU load average low threshold value. A partition with a load average below this value is considered to have unneeded CPU capacity.
Minimum entitlement per VP	0.1	0.5	1.0	The minimum amount of entitled capacity per virtual processor. This attribute prevents a partition from having degraded performance by having too many virtual processors relative to its entitled capacity. When entitled capacity is removed from a partition, virtual processors will also be removed if the amount of entitled capacity for each virtual processor falls below this number. Default value is 0.5. Minimum value is 0.1. Maximum value is 1.0.
Maximum entitlement per VP	0.1	0.8	1.0	The maximum amount of entitled capacity per virtual processor. This attribute controls the amount of available capacity that may be used by an uncapped shared processor partition. When entitled capacity is added to a partition, virtual processors will be added if the amount of the entitled capacity for each virtual processor goes above this number. Increasing the number of virtual processors in an uncapped partition allows the partition to use more of the available processor capacity.
Immediate release of free CPU	-	no	-	Indicates when processor capacity not needed by a partition is removed from the partition. A value of no indicates unneeded CPU capacity remains in the partition until another partition has a need for it. A value of yes indicates unneeded CPU capacity is removed from the partition when the partition no longer has a need for it.

Setting up the RMC communication to the managed partitions

To set up the communication, select **Set up Management of Logical Partitions** from the Partition Load Manager menu. Figure 6-12 shows the PLM Setup panel.

If you are using AIX Version 5.2, an .rhosts file may be required. In AIX Version 5.3, this file was not needed.

Important: Before you set up the RMC communication, make sure that the hostnames of the client partitions are resolved correctly on the PLM server. On the clients' partitions the hostname of PLM server has to be resolved correctly.

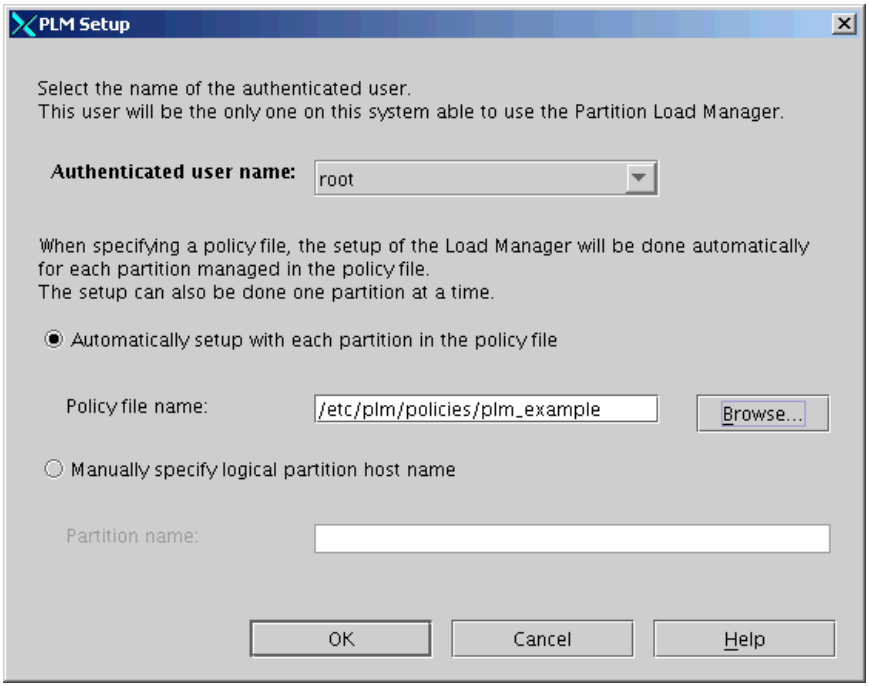

Figure 6-12 PLM Setup panel

This setup procedure requests the user ID under which the PLM is required to run. This is an AIX user which must exist on the PLM server prior to setup. The user ID is used to set up the RMC access control list (ACL) files on the partitions. ACL files are used to authenticate users when they connect to the RMC subsystem. PLM requires access to the IBM.LPAR Resource class on each managed partition. In our example we used the root user ID.

To verify the RMC communication enter the following command on the PLM server for all managed client partitions:

```
# CT_CONTACT=dbsrv lsrsrc IBM.LPAR
Resource Persistent Attributes for IBM.LPAR
resource 1:
        Name              = "DB_Server"
        LPARFlags         = 7
        MaxCPU            = 10
        MinCPU            = 1
        CurrentCPUs       = 2
        MinEntCapacity    = 0.2
        MaxEntCapacity    = 1
        CurEntCapacity    = 0.5
        MinEntPerVP       = 0.1
        SharedPoolCount   = 0
        MaxMemory         = 1024
        MinMemory         = 128
        CurrentMemory     = 512
        CapacityIncrement = 0.01
        LMBSize           = 16
        VarWeight         = 128
        CPUIntvl          = 0
        MemIntvl          = 0
        CPULoadMax        = 0
        CPULoadMin        = 0
        MemLoadMax        = 0
        MemLoadMin        = 0
        MemPgStealMax     = 0
        ActivePeerDomain  = ""
        NodeNameList      = {"dbsrv"}
```

If you see an output similar to that shown here, the RMC setup was successful.

If the setup of the clients was not successful, you can check the setup of the RMC configuration with the **ctsthl** command. On the PLM server system there should be an entry for the partition. On the partitions, there should be an entry for the PLM server machine.

The output on the partition applsrv looks like this:

```
# /usr/sbin/rsct/bin/ctsthl -l
ctsthl: Contents of trusted host list file:
--------------------
Host Identity:                0.0.0.0
Identifier Generation Method: rsa512
Identifier Value:
120200d2ce4c145bd95aaa0ff99633fbe0ad4fd7d90c545a03c4affdca9c39482357cf7dd49f53e
3
--------------------
Host Identity:                127.0.0.1
Identifier Generation Method: rsa512
Identifier Value:
120200babe94a92de3ffbfb1efa34d4e6a9e4a44421a1865c9123f97e43078299d217a25965b2d0
3
--------------------
Host Identity:                9.3.5.144
Identifier Generation Method: rsa512
Identifier Value:
120200d2ce4c145bd95aaa0ff99633fbe0ad4fd7d90c545a03c4affdca9c39482357cf7dd49f53e
3
--------------------
Host Identity:                ::1
Identifier Generation Method: rsa512
Identifier Value:
120200babe94a92de3ffbfb1efa34d4e6a9e4a44421a1865c9123f97e43078299d217a25965b2d0
3
--------------------
Host Identity:                applsrv
Identifier Generation Method: rsa512
Identifier Value:
120200babe94a92de3ffbfb1efa34d4e6a9e4a44421a1865c9123f97e43078299d217a25965b2d0
3
--------------------
Host Identity:                loopback
Identifier Generation Method: rsa512
Identifier Value:
120200babe94a92de3ffbfb1efa34d4e6a9e4a44421a1865c9123f97e43078299d217a25965b2d0
3
--------------------
Host Identity:                server1
Identifier Generation Method: rsa512
Identifier Value:
120200b9a60f06b03ab05c76bcab1afe9dbe47b793027419f3904883da205637b4e7ba60b2af099
3
--------------------
```

You find an entry for the partition applsrv and for the PLM system server1.

If an entry does not match, remove it by running the following command:

```
# /usr/sbin/rsct/bin/ctsthl -d -n applsrv
```

After you removed the Host Identity you can run the setup of your partition again to create the Host Identity again.

Start up the Partition Load Manager using the created policy file. The start panel is shown in Figure 6-13.

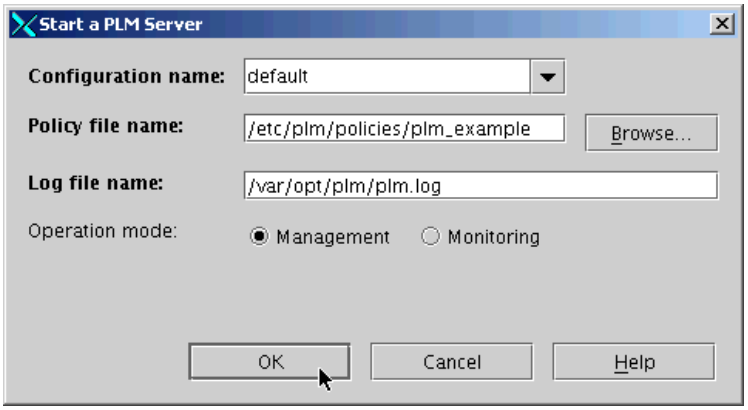

Figure 6-13 Start a PLM Server panel

You can choose to start the PLM in management or monitoring mode. For our example, we started the PLM in management mode.

You can choose the name of your logfile. It is located in the /var/opt/plm directory of the PLM server. When you use the **`tail -f`** command on this file you see the PLM events.

```
# tail -f plm.log
<07/06/04 16:47:21> <PLM_TRC> Event notification of CPUZone low for Apps_Server .
<07/06/04 16:47:32> <PLM_TRC> Event notification of CPUZone low for DB_Server .
<07/06/04 16:48:23> <PLM_TRC> Event notification of CPUZone low for Web_Server .
<07/06/04 16:48:31> <PLM_TRC> Event notification of CPUZone low for Apps_Server .
<07/06/04 16:48:42> <PLM_TRC> Event notification of CPUZone low for DB_Server .
<07/06/04 16:49:33> <PLM_TRC> Event notification of CPUZone low for Web_Server .
<07/06/04 16:49:41> <PLM_TRC> Event notification of CPUZone low for Apps_Server .
<07/06/04 16:49:52> <PLM_TRC> Event notification of CPUZone low for DB_Server .
<07/06/04 16:50:43> <PLM_TRC> Event notification of CPUZone low for Web_Server .
<07/06/04 16:50:51> <PLM_TRC> Event notification of CPUZone low for Apps_Server
```

To demonstrate the process of adding processor resources we produced an artificial workload on the partition websrv. To check the PLM setting for the partition, we chose **Show LPAR Details** in the Web-based System Manager

menu. After choosing the PLM instance, we got an overview of the partition settings.

On Figure 6-14 you see the PLM defined setting for the websrv partition.

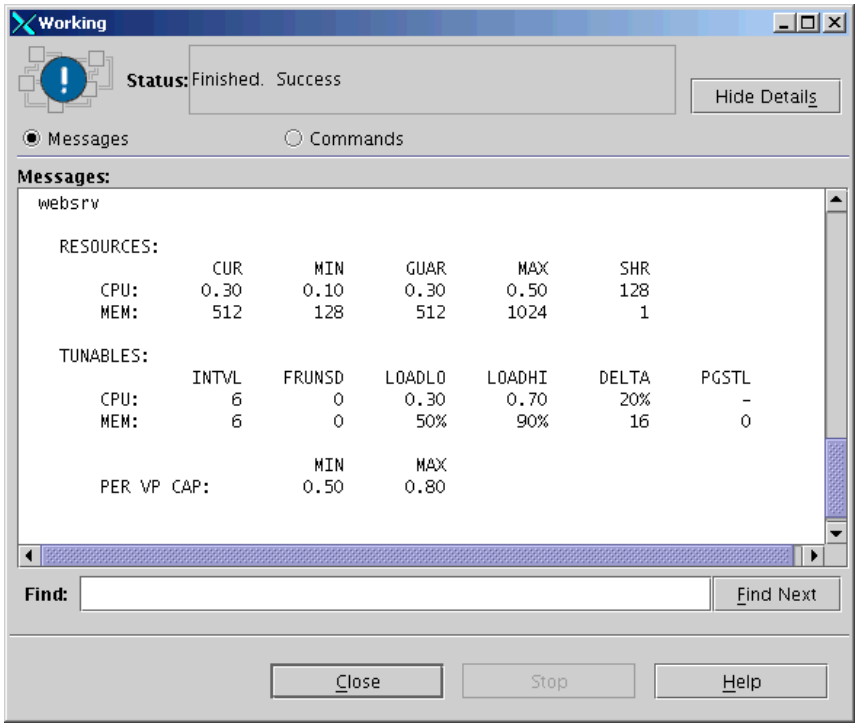

Figure 6-14 Show LPAR Details screen

We produced a processor utilization around 95% on the websrv partition over a period of 10 minutes. On the Show LPAR statistic panel we monitored the threshold value for the load average of the partition. Because it is an average value it starts growing slowly with the time and utilization of the processors.

To see the average processor load the PLM uses to determine the threshold, open the Show LPAR Statistic screen. Select the check box to automatically refresh statistics and choose a refresh rate. The default refresh rate is 30 seconds.

Figure 6-15 shows the LPAR statistics panel. The Load average of the websrv partition is increasing over the defined threshold of 70%.

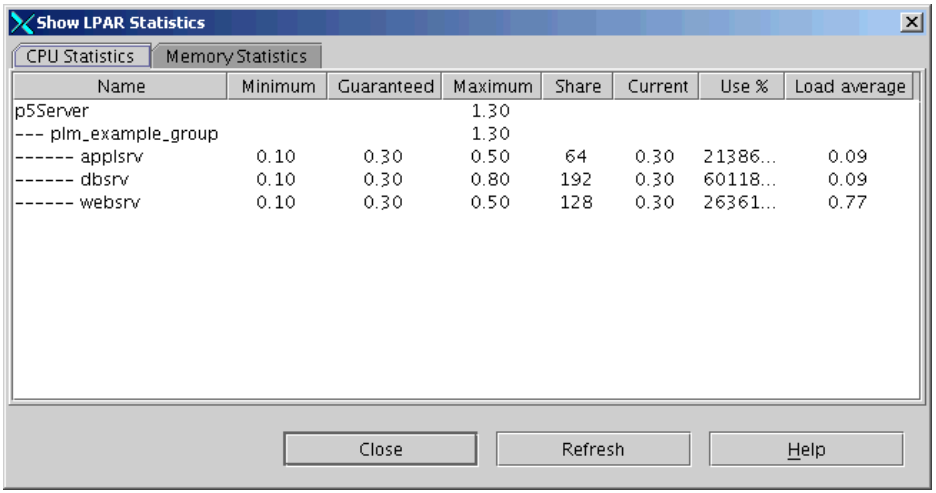

Figure 6-15 Show LPAR Statistics panel

Shown in the following is the plm.log file. Notice that the amount of capacity entitlement is increasing.

```
# tail -f plm.log
<07/14/04 10:35:05> <PLM_TRC> Event notification of CPUZone low for Apps_Server .
<07/14/04 10:35:37> <PLM_TRC> Event notification of CPUZone high for Web_Server .
<07/14/04 10:35:37> <PLM_TRC> CPU resources allocated to Web_Server from free pool.
<07/14/04 10:35:45> <PLM_TRC> Event notification of CPUZone low for DB_Server .
<07/14/04 10:35:45> <PLM_TRC> Added 0 virtual CPUs and 0.06 units of CPU capacity for
    Web_Server.
<07/14/04 10:35:45> <PLM_TRC> Event notification of ConfigChanged for Web_Server .
<07/14/04 10:35:45> <PLM_TRC> Current number of virtual CPUs is 1 for Web_Server .
<07/14/04 10:35:45> <PLM_TRC> Current CPU entitlement is 0.36 for Web_Server .
<07/14/04 10:35:45> <PLM_TRC> Current memory is 512 MB for Web_Server .
<07/14/04 10:36:15> <PLM_TRC> Event notification of CPUZone low for Apps_Server .
<07/14/04 10:36:47> <PLM_TRC> Event notification of CPUZone high for Web_Server .
<07/14/04 10:36:47> <PLM_TRC> CPU resources allocated to Web_Server from free pool.
<07/14/04 10:36:55> <PLM_TRC> Added 0 virtual CPUs and 0.07 units of CPU capacity for
    Web_Server.
<07/14/04 10:36:55> <PLM_TRC> Event notification of ConfigChanged for Web_Server .
<07/14/04 10:36:55> <PLM_TRC> Current number of virtual CPUs is 1 for Web_Server .
<07/14/04 10:36:55> <PLM_TRC> Current CPU entitlement is 0.43 for Web_Server .
<07/14/04 10:36:55> <PLM_TRC> Current memory is 512 MB for Web_Server .
<07/14/04 10:36:55> <PLM_TRC> Event notification of CPUZone low for DB_Server .
```

PLM increases the capacity entitlement in steps of the defined 20% of the current capacity entitlement until it reaches the maximum capacity entitlement.

In our example this is 0.5 processing units as shown in Figure 6-16.

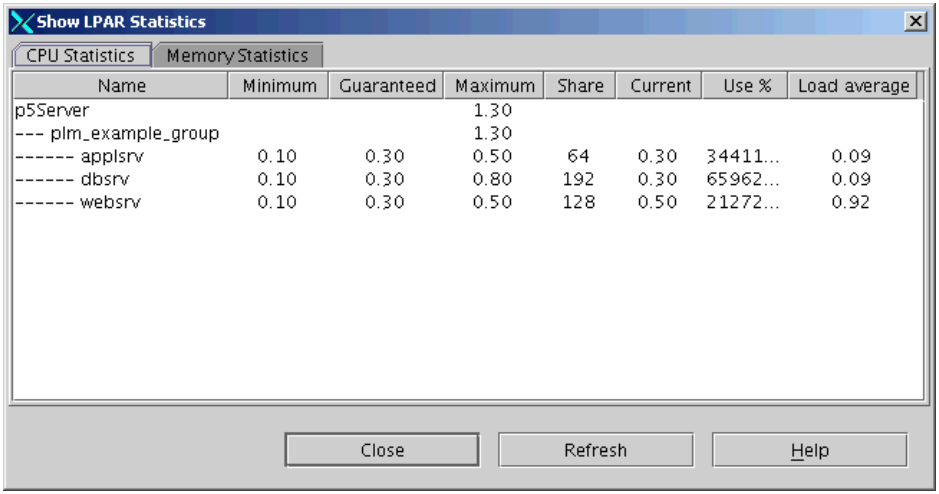

Figure 6-16 Show LPAR Statistics Panel showing websrv partition at maximum

A

Worksheets for partition configuration planning

This appendix contains worksheets that you can use when planning for partition configuration. Print copies of these worksheets, and be sure to complete the worksheets for each partition that you plan to create. A partition environment is complex, and having all the data on a single worksheet can provide a smoother installation and easier system recovery in the event of a problem.

Worksheet instructions

The information in this section explains how to use the planning worksheets.

Partition properties worksheet instructions

During the initial LPAR design phase, high-level choices about LPAR resource usage should be made to enable an adequate machine configuration. The number of desired LPARs should be determined, with usage of each processor, memory, and I/O requirements recorded for each LPAR. Availability requirements should be considered and recorded to ensure adequate hardware redundancy.

- **Partition number**: Arbitrary, used as shorthand to link with I/O properties worksheet.
- **Partition name:** Unique name for this partition (up to 31 chars).
- **Number of processors dedicated/virtual:** Record the minimum/desired/maximum number of dedicated and virtual processors required for this partition.
- **Processing power minimum/desired/maximum:** Record the minimum/desired/maximum processing units.
- **Memory size minimum/ desired/ maximum:** Memory size minimum/desired/maximum memory requirements.
- **Required physical network adapter:** Record number of required physical network adapters.
- **Required virtual network adapters:** Record number of required virtual network adapters.
- **Disk drives:** Note partition disk requirements including rootvg, data, mirroring, for example.
- **Comments:** Record the file systems size, capped or uncapped processor mode, for example.
- **Partition hostname:** Record hostname for partition. This name will need to be resolvable by some method (DNS, /etc/hosts, NIS).
- **Network configuration:** Record IP, netmask, for example, for each adapter.
- **Application software stack levels**: Record levels of all applications to run in partition. You can use this information to ensure application availability, and any OS or application prerequisites.

- **Availability requirements:** Record desired availability requirements for this partition. This information can be used to plan for appropriate hardware redundancy and any needed failure environment. The following definitions may be useful.
  - **Disaster Tolerant (AE-5)**: These business functions must always be available, and any systems failure must be invisible to the user. This means that there can be no interruption of work, no transactions lost and no performance degradation, with continuous operations and remote backup in the case of disasters such as power outage, fire, flood, or earthquake.
  - **Fault Tolerant (AE-4)**: These business functions demand continuous computing, and failures are invisible. This means no interruption of work, no transactions lost, no performance degradation, and continuous 7X24 operation.
  - **Fault Resilient (AE-3)**: These business functions require uninterrupted computing services, either during essential time periods, or during most hours of the day and most days of the week throughout the year. Users stay online, even though the current transaction may need restarting, and performance degradation may occur.
  - **High Availability (AE-2)**: These business functions can survive minimally interrupted computing services. The user will be interrupted but can log on again. Some transactions may have to be rerun, and performance degradation is possible.
  - **Highly Reliable (AE-1)**: These business functions can be interrupted, as long as data integrity is maintained. The user's work stops, and an uncontrolled shutdown occurs.
  - **Conventional (AE-0)**: These business functions can be interrupted, and data integrity is unimportant. The user's work stops, uncontrolled shutdown occurs, and data may be lost or corrupted.

For an I/O drawer, duplicate this worksheet for each installed I/O drawer. Record the drawer location code, adapter type for each slot, and then the partition assignment.

Worksheet examples used for scenarios

Examine the following sample worksheets before filling out worksheets for your scenario. The sample worksheets contain the values we used during the configuration of the scenarios described in Chapter 5, "AIX and Virtual I/O Server configuration scenarios" on page 137.

Table A-1 An example of partition properties worksheet (part 1 of 2)

ID	Partition name	Number of processors dedicated/ virtual /des /min /max	Processing power desired/ minimum/ maximum	Memory size desired/ minimum/ maximum	Required network adapter phys/virt	Disk drives virt / phys	Comments (file system size, processors capped/ uncapped, etc.)
1	I/O_Server_1	V 2/1/8	0.5/0.1/0.8	512/128/1GB	P1/V1	P 4/V4	uncapped, system disk, rootvg for clients, datavg for clients
2	DB_Server	V 2/1/5	0.3/0.1/0.5	512/128/1GB	P0/V1	V 4	uncapped, rootvg, datavg, mirroring
3	Apps_Server	V 3/1/5	0.3/0.1/0.5	512/128/1GB	P0/V1	V 2	uncapped, rootvg, mirroring
4	Web_Server	V 2/1/8	0.3/0.1/0.8	512/128/1GB	P0/V1	V 2	uncapped, rootvg, mirroring
5	I/O_Server_2	V 2/1/8	0.5/0.3/0.8	512/128/1GB	P1/V1	P 4	uncapped

Table A-2 An example of partition properties worksheet (part 1 of 2)

Part. #	Partition hostname	Network configuration / virtual / shared	Application software stack levels	Availability requirements	Comments
1	p52iosrv1	S 9.3.5.150	Virtual I/O Server serving primary disks to the clients		
2	p52iosrv2	S 9.3.5.155	Virtual I/O Server serving disks to the client for HA (mirror)		
3	websrv	V 9.3.5.158	IBM AIX 5L Version 5.3		
4	appssrv	V 9.3.5.157	IBM AIX 5L Version 5.3		
5	dbsrv	V 9.3.5.156	IBM AIX 5L Version 5.3		

Table A-3 An example of Virtual I/O Server configuration planning sheet

IO Server										
PV name	VG name	LV name	LV size	Virtual SCSI device	Server slot	Client part. ID	Client slot	Virtual target device	Client PV name	Comments
hdisk0	rootvg									
hdisk1	rootvg									
hdisk2	rootvg_clients	rootvg_dbsrv	2 GB	vhost0	20	2	3	vdbsvr	hdisk0	rootvg
hdisk2	rootvg_clients	rootvg_appssrv	2 GB	vhost1	30	3	3	vappssrv	hdisk0	rootvg
hdisk2	rootvg_clients	rootvg_websrv	2 GB	vhost2	40	4	3	vwebsrv	hdisk0	rootvg
hdisk3	datavg_clients	datavg_dbsrv	2 GB	vhost0		2		vdbsrvdata	hdisk2	datavg
hdisk3	datavg_clients									

IO Server										
PV name	**VG name**	**LV name**	**LV size**	**Virtual SCSI device**	**Server slot**	**Client part. ID**	**Client slot**	**Virtual target device**	**Client PV name**	**Comments**
hdisk3	datavg_clients	datavg_websrv	2 GB	vhost0						

Table A-4 An example of Ethernet adapter planning

Partition name	Physical / Shared / Virtual adapter	IP address	Server side slot	LAN IDs	Comments
I/O_Server_1	P		Bus 2 Slot T6		for shared Ethernet VIO 1
I/O_Server_1	V/S (trunk flag)	9.3.5.150	2	1	for shared Ethernet
I/O_Server_2	P		Bus 2 Slot C4		for shared Ethernet VIO 2
I/O_Server_2	V/S (trunk flag)	9.3.5.155	2	2	for shared Ethernet
Web_Server	V	9.3.5.158	2	1	
Web_Server	V		4	2	for redundancy
DB_Server	V	9.3.5.156	2	1	
Apps_Server	V	9.3.5.157	2	1	
Apps_Server	V		4	2	for redundancy

Table A-5 An example of I/O drawer resource worksheet

I/O drawer location and serial number	Adapter physical location / slot number	Adapter type	Partition ID assignment	Comments
	Bus 2 Slot T6	PCI 10/100/1000 Mbps Ethernet UTP 2-port adapter	1	1 port for shared Ethernet
	Bus 3 Slot T11	Storage controller	1	4 disks x 36 GB
	Bus 3 Slot T12	Other Mass Storage Controller	1	CD-ROM, Streamer etc.
	Bus 2 Slot C1	Storage controller	2	4 disks x 36 GB
	Bus 2 Slot C2	PCI 10/100 Mbps Ethernet adapter	2	for Link Aggregation
	Bus 2 Slot C4	PCI 10/100/1000 Mbps Ethernet UTP-2 port adapter	2	1 port for shared Ethernet 1 port for Link Aggregation

Empty partition properties worksheets

Complete the following worksheets to plan configurations that you intend to set up on your system.

Table A-6 Partition properties worksheet (part 1 of 2)

ID	Partition name	Number of processors dedicated/ virtual /des /min /max	Processing power desired/ minimum/ maximum	Memory size desired/ minimum/ maximum	Required network adapter phys/virt	Disk drives virt / phys	Comments (file system size, processors capped/ uncapped, etc.)

Table A-7 Partition properties worksheet (part 2 of 2)

Part. #	Partition hostname	Network configuration / virtual / shared	Application software stack levels	Availability requirements	Comments

Table A-8 The Virtual I/O Server configuration planning sheet

IO Server										
PV name	VG name	LV name	LV size	Virtual SCSI device	Server slot	Client part. ID	Client slot	Virtual target device	Client PV name	Comments

Table A-9 Ethernet adapter planning

Partition name	Physical / Shared / Virtual adapter	IP address	Server side slot	LAN IDs	Comments

Table A-10 I/O drawer resource worksheet

I/O drawer location and serial number	Adapter physical location / slot number	Adapter type	Partition assignment	Comments

B

Supported SCSI commands

This appendix provides a list all SCSI commands supported by virtual SCSI.

- TEST_UNIT_READY
- REQUEST_SENSE
- FORMAT_UNIT
- READ6
- READ10
- READ16
- WRITE6
- WRITE10
- WRITE16
- INQUIRY
- MODE_SELECT6
- MODE_SELECT10
- RESERVE6
- RESERVE10
- RELEASE6
- RELEASE10

- MODE_SENSE6
- MODE_SENSE10
- START_STOP_UNIT
- SEND_DIAGNOSTIC
- READ_CAPACITY10
- WRITE_AND_VERIFY
- WRITE_AND_VERIFY16
- SERVICE_ACTION_IN
- READ_CAPACITY16
- REPORT_LUNS
- ABORT_TASK
- ABORT_TASK_SET
- CLEAR_TASK_SET
- LUN_RESET
- CLEAR_ACA

Abbreviations and acronyms

ABI	Application Binary Interface
AC	Alternating Current
ACL	Access Control List
ADSM	ADSTAR Distributed Storage Manager
ADSTAR	Advanced Storage and Retrieval
AFPA	Adaptive Fast Path Architecture
AFS®	Andrew File System
AH	Authentication Header
AIO	Asynchronous I/O
AIX	Advanced Interactive Executive
ANSI	American National Standards Institute
APAR	Authorized Program Analysis Report
API	Application Programming Interface
AppA	Application Audio
AppV	Application Video
ARP	Address Resolution Protocol
ASCI	Accelerated Strategic Computing Initiative
ASCII	American National Standards Code for Information Interchange
ASR	Address Space Register
ATA	Advanced Technology Attachment
ATM	Asynchronous Transfer Mode
AuditRM	Audit Log Resource Manager
AUI	Attached Unit Interface
AWT	Abstract Window Toolkit
BCT	Branch on CounT
BFF	Backup File Format
BI	Business Intelligence
BIND	Berkeley Internet Name Domain
BIST	Built-In Self-Test
BLAS	Basic Linear Algebra Subprograms
BLOB	Binary Large Object
BLV	Boot Logical Volume
BOOTP	Boot Protocol
BOS	Base Operating System
BPF	Berkeley Packet Filter
BRX	Branch Execution Unit
BSC	Binary Synchronous Communications
BSD	Berkeley Software Distribution
CA	Certificate Authority
CAD	Computer-Aided Design
CAE	Computer-Aided Engineering
CAM	Computer-Aided Manufacturing
CATE	Certified Advanced Technical Expert
CATIA	Computer-Graphics Aided Three-Dimensional Interactive Application
CCM	Common Character Mode
CD	Compact Disk
CDE	Common Desktop Environment
CDLI	Common Data Link Interface
CD-R	CD Recordable

CD-ROM	Compact Disk-Read Only Memory
CE	Customer Engineer
CEC	Central Electronics Complex
CFD	Computational Fluid Dynamics
CFM	Configuration File Manager
CGE	Common Graphics Environment
CHRP	Common Hardware Reference Platform
CIM	Common Information Model
CISPR	International Special Committee on Radio Interference
CIU	Core Interface Unit
CLI	Command Line Interface
CLIO/S	Client Input/Output Sockets
CLVM	Concurrent LVM
CMOS	Complimentary Metal-Oxide Semiconductor
CMP	Certificate Management Protocol
COFF	Common Object File Format
COLD	Computer Output to Laser Disk
CPU	Central Processing Unit
CRC	Cyclic Redundancy Check
CRL	Certificate Revocation List
CSID	Character Set ID
CSM	Cluster Systems Management
CSR	Customer Service Representative
CSS	Communication Subsystems Support
CSU	Customer Set-Up
CUoD	Capacity Upgrade on Demand
CWS	Control Workstation
DAD	Duplicate Address Detection
DAS	Dual Attach Station
DASD	Direct Access Storage Device
DAT	Digital Audio Tape
DBCS	Double Byte Character Set
DBE	Double Buffer Extension
DC	Direct Current
DCE	Distributed Computing Environment
DCM	Dual Chip Module
DCUoD	Dynamic Capacity Upgrade on Demand
DDC	Display Data Channel
DDS	Digital Data Storage
DE	Dual-Ended
DES	Data Encryption Standard
DFL	Divide Float
DFP	Dynamic Feedback Protocol
DFS™	Distributed File System
DGD	Dead Gateway Detection
DH	Diffie-Hellman
DHCP	Dynamic Host Configuration Protocol
DIMM	Dual In-Line Memory Module
DIP	Direct Insertion Probe
DIT	Directory Information Tree
DIVA	Digital Inquiry Voice Answer
DLPAR	Dynamic LPAR
DLT	Digital Linear Tape
DMA	Direct Memory Access
DMT	Directory Management Tool
DMTF	Distributed Management Task Force
DN	Distinguished Name
DNLC	Dynamic Name Lookup Cache

DNS	Domain Naming System
DOE	Department of Energy
DOI	Domain of Interpretation
DOM	Document Object Model
DOS	Disk Operating System
DPCL	Dynamic Probe Class Library
DRAM	Dynamic Random Access Memory
DRM	Dynamic Reconfiguration Manager
DS	Differentiated Service
DSA	Dynamic Segment Allocation
DSE	Diagnostic System Exerciser
DSMIT	Distributed SMIT
DSU	Data Service Unit
DTD	Document Type Definition
DTE	Data Terminating Equipment
DVD	Digital Versatile Disk
DW	Data Warehouse
DWA	Direct Window Access
EA	Effective Address
EC	Engineering Change
ECC	Error Checking and Correcting
ECN	Explicit Congestion Notification
EEH	Extended Error Handling
EEPROM	Electrically Erasable Programmable Read Only Memory
EFI	Extensible Firmware Interface
EHD	Extended Hardware Drivers
EIA	Electronic Industries Association
EIM	Enterprise Identity Mapping
EISA	Extended Industry Standard Architecture
ELA	Error Log Analysis
ELF	Executable and Linking Format
EMU	European Monetary Union
EOF	End of File
EPOW	Environmental and Power Warning
ERRM	Event Response resource manager
ESID	Effective Segment ID
ESP	Encapsulating Security Payload
ESS	Enterprise Storage Server®
ESSL	Engineering and Scientific Subroutine Library
ETML	Extract, Transformation, Movement, and Loading
F/C	Feature Code
F/W	Fast and Wide
FC	Fibre Channel
FCAL	Fibre Channel Arbitrated Loop
FCC	Federal Communication Commission
FCP	Fibre Channel Protocol
FDDI	Fiber Distributed Data Interface
FDPR	Feedback Directed Program Restructuring
FDX	Full Duplex
FIFO	First In/First Out
FLASH EPROM	Flash Erasable Programmable Read-Only Memory
FLIH	First Level Interrupt Handler
FLOP	Floating Point Operation
FMA	Floating point Multiply Add operation
FP	Fixed Point
FPR	Floating Point Register
FPU	Floating Point Unit

FRCA	Fast Response Cache Architecture
FRU	Field Replaceable Unit
FSRM	File System Resource Manager
FTP	File Transfer Protocol
FTP	File Transfer Protocol
GAI	Graphic Adapter Interface
GAMESS	General Atomic and Molecular Electronic Structure System
GID	Group ID
GPFS	General Parallel File System
GPR	General-Purpose Register
GUI	Graphical User Interface
GUID	Globally Unique Identifier
HACMP	High Availability Cluster Multi Processing
HACWS	High Availability Control Workstation
HBA	Host Bus Adapters
HCON	IBM AIX Host Connection Program/6000
HDX	Half Duplex
HFT	High Function Terminal
HIPPI	High Performance Parallel Interface
HiPS	High Performance Switch
HiPS LC-8	Low-Cost Eight-Port High Performance Switch
HMC	Hardware Management Console
HMT	Hardware Multithreading
HostRM	Host Resource Manager
HP	Hewlett-Packard
HPF	High Performance FORTRAN
HPSSDL	High Performance Supercomputer Systems Development Laboratory
HP-UX	Hewlett-Packard UNIX
HTML	Hyper-text Markup Language
HTTP	Hypertext Transfer Protocol
Hz	Hertz
I/O	Input/Output
I^2C	Inter Integrated-Circuit Communications
IAR	Instruction Address Register
IBM	International Business Machines
ICCCM	Inter-Client Communications Conventions Manual
ICE	Inter-Client Exchange
ICElib	Inter-Client Exchange library
ICMP	Internet Control Message Protocol
ID	Identification
IDE	Integrated Device Electronics
IDL	Interface Definition Language
IDS	Intelligent Decision Server
IEEE	Institute of Electrical and Electronics Engineers
IETF	Internet Engineering Task Force
IHS	IBM HTTP Server
IHV	Independent Hardware Vendor
IIOP	Internet Inter-ORB Protocol
IJG	Independent JPEG Group
IKE	Internet Key Exchange
ILMI	Integrated Local Management Interface
ILS	International Language Support
IM	Input Method
INRIA	Institut National de Recherche en Informatique et en Automatique

IOCTL	I/O Control
IP	Internetwork Protocol (OSI)
IPL	Initial Program Load
IPSec	IP Security
IrDA	Infrared Data Association (which sets standards for infrared support including protocols for data interchange)
IRQ	Interrupt Request
IS	Integrated Service
ISA	Industry Standard Architecture, Instruction Set Architecture
ISAKMP	Internet Security Association Management Protocol
ISB	Intermediate Switch Board
ISDN	Integrated-Services Digital Network
ISMP	InstallSheild Multi-Platform
ISNO	Interface Specific Network Options
ISO	International Organization for Standardization
ISV	Independent Software Vendor
ITSO	International Technical Support Organization
IXFR	Incremental Zone Transfer
JBOD	Just a Bunch of Disks
JCE	Java™ Cryptography Extension
JDBC	Java Database Connectivity
JFC	Java Foundation Classes
JFS	Journaled File System
JSSE	Java Secure Sockets Extension
JTAG	Joint Test Action Group
JVMPI	Java Machine Profiling Interface
KDC	Key Distribution Center
L1	Level 1
L2	Level 2
L3	Level 3
LAM	Loadable Authentication Module
LAN	Local Area Network
LANE	Local Area Network Emulation
LAPI	Low-Level Application Programming Interface
LDAP	Lightweight Directory Access Protocol
LDIF	LDAP Directory Interchange Format
LED	Light Emitting Diode
LFD	Load Float Double
LFT	Low Function Terminal
LID	Load ID
LLNL	Lawrence Livermore National Laboratory
LMB	Logical Memory Block
LP	Logical Partition
LPAR	Logical Partition
LP64	Long-Pointer 64
LPI	Lines Per Inch
LPP	Licensed Program Product
LPR/LPD	Line Printer/Line Printer Daemon
LRU	Least Recently Used
LTG	Logical Track Group
LUN	Logical Unit Number
LV	Logical Volume
LVCB	Logical Volume Control Block
LVD	Low Voltage Differential
LVM	Logical Volume Manager
MAP	Maintenance Analysis Procedure

MASS	Mathematical Acceleration Subsystem
MAU	Multiple Access Unit
MBCS	Multi-Byte Character Support
Mbps	Megabits Per Second
MBps	Megabytes Per Second
MCA	Micro Channel® Architecture
MCAD	Mechanical Computer-Aided Design
MCM	Multichip Module
MDF	Managed Object Format
MDI	Media Dependent Interface
MES	Miscellaneous Equipment Specification
MFLOPS	Million of FLoating point Operations Per Second
MII	Media Independent Interface
MIB	Management Information Base
MIP	Mixed-Integer Programming
MLR1	Multi-Channel Linear Recording 1
MMF	Multi-Mode Fibre
MODS	Memory Overlay Detection Subsystem
MP	Multiprocessor
MPC-3	Multimedia PC-3
MPI	Message Passing Interface
MPIO	Multipath I/O
MPOA	Multiprotocol over ATM
MPP	Massively Parallel Processing
MPS	Mathematical Programming System
MSS	Maximum Segment Size
MST	Machine State
MTU	Maximum Transmission Unit
MWCC	Mirror Write Consistency Check
MX	Mezzanine Bus
NBC	Network Buffer Cache
NCP	Network Control Point
ND	Neighbor Discovery
NDP	Neighbor Discovery Protocol
NDS	Novell Directory Services
NFB	No Frame Buffer
NFS	Network File System
NHRP	Next Hop Resolution Protocol
NIM	Network Installation Management
NIMOL	NIM on Linux
NIS	Network Information Service
NL	National Language
NLS	National Language Support
NT-1	Network Terminator-1
NTF	No Trouble Found
NTP	Network Time Protocol
NUMA	Non-Uniform Memory Access
NUS	Numerical Aerodynamic Simulation
NVRAM	Non-Volatile Random Access Memory
NWP	Numerical Weather Prediction
OACK	Option Acknowledgment
OCS	Online Customer Support
ODBC	Open DataBase Connectivity
ODM	Object Data Manager
OEM	Original Equipment Manufacturer
OLAP	Online Analytical Processing
OLTP	Online Transaction Processing
ONC+	Open Network Computing
OOUI	Object-Oriented User Interface

OSF	Open Software Foundation, Inc.
OSL	Optimization Subroutine Library
OSLp	Parallel Optimization Subroutine Library
P2SC	POWER2™ Single/Super Chip
PAG	Process Authentication Group
PAM	Pluggable Authentication Mechanism
PAP	Privileged Access Password
PBLAS	Parallel Basic Linear Algebra Subprograms
PCB	Protocol Control Block
PCI	Peripheral Component Interconnect
PDT	Paging Device Table
PDU	Power Distribution Unit
PE	Parallel Environment
PEDB	Parallel Environment Debugging
PEX	PHIGS Extension to X
PFS	Perfect Forward Security
PGID	Process Group ID
PHB	Processor Host Bridges
PHY	Physical Layer
PID	Process ID
PID	Process ID
PIOFS	Parallel Input Output File System
PKCS	Public-Key Cryptography Standards
PKI	Public Key Infrastructure
PKR	Protection Key Registers
PMTU	Path MTU
POE	Parallel Operating Environment
POP	Power-On Password
POSIX	Portable Operating Interface for Computing Environments
POST	Power-On Self-test
POWER	Performance Optimization with Enhanced Risc (Architecture)
PPC	PowerPC
PPM	Piecewise Parabolic Method
PPP	Point-to-Point Protocol
PREP	PowerPC Reference Platform
PRNG	Pseudo-Random Number Generator
PSE	Portable Streams Environment
PSSP	Parallel System Support Program
PTF	Program Temporary Fix
PTPE	Performance Toolbox Parallel Extensions
PTX®	Performance Toolbox
PV	Physical Volume
PVC	Permanent Virtual Circuit
PVID	Physical Volume Identifier
QMF™	Query Management Facility
QoS	Quality of Service
QP	Quadratic Programming
RAID	Redundant Array of Independent Disks
RAM	Random Access Memory
RAN	Remote Asynchronous Node
RAS	Reliability, Availability, and Serviceability
RDB	Relational DataBase
RDBMS	Relational Database Management System
RDF	Resource Description Framework

RDISC	ICMP Router Discovery
RDN™	Relative Distinguished Name
RDP	Router Discovery Protocol
RFC	Request for Comments
RIO	Remote I/O
RIP	Routing Information Protocol
RIPL	Remote Initial Program Load
RISC	Reduced Instruction-Set Computer
RMC	Resource Monitoring and Control
ROLTP	Relative Online Transaction Processing
RPA	RS/6000 Platform Architecture
RPC	Remote Procedure Call
RPL	Remote Program Loader
RPM	Redhat Package Manager
RSC	RISC Single Chip
RSCT	Reliable Scalable Cluster Technology
RSE	Register Stack Engine
RSVP	Resource Reservation Protocol
RTC	Real-Time Clock
RVSD	Recoverable Virtual Shared Disk
SA	Secure Association
SACK	Selective Acknowledgments
SAN	Storage Area Network
SAR	Solutions Assurance Review
SAS	Single Attach Station
SASL	Simple Authentication and Security Layer
SBCS	Single-Byte Character Support
ScaLAPACK	Scalable Linear Algebra Package
SCB	Segment Control Block
SCSI	Small Computer System Interface
SCSI-SE	SCSI-Single Ended
SDK	Software Development Kit
SDLC	Synchronous Data Link Control
SDR	System Data Repository
SDRAM	Synchronous Dynamic Random Access Memory
SE	Single Ended
SEPBU	Scalable Electrical Power Base Unit
SGI	Silicon Graphics Incorporated
SGID	Set Group ID
SHLAP	Shared Library Assistant Process
SID	Segment ID
SIT	Simple Internet Transition
SKIP	Simple Key Management for IP
SLB	Segment Lookaside Buffer
SLIH	Second Level Interrupt Handler
SLIP	Serial Line Internet Protocol
SLR1	Single-Channel Linear Recording 1
SM	Session Management
SMB	Server Message Block
SMIT	System Management Interface Tool
SMP	Symmetric Multiprocessor
SMS	System Management Services
SNG	Secured Network Gateway
SNIA	Storage Networking Industry Association
SNMP	Simple Network Management Protocol

SOI	Silicon-on-Insulator
SP	IBM RS/6000 Scalable POWER parallel Systems
SP	Service Processor
SPCN	System Power Control Network
SPEC	System Performance Evaluation Cooperative
SPI	Security Parameter Index
SPM	System Performance Measurement
SPOT	Shared Product Object Tree
SPS	SP Switch
SPS-8	Eight-Port SP Switch
SRC	System Resource Controller
SRN	Service Request Number
SSA	Serial Storage Architecture
SSC	System Support Controller
SSL	Secure Socket Layer
STFDU	Store Float Double with Update
STP	Shielded Twisted Pair
SUID	Set User ID
SUP	Software Update Protocol
SVC	Switch Virtual Circuit
SVC	Supervisor or System Call
SWVPD	Software Vital Product Data
SYNC	Synchronization
TCB	Trusted Computing Base
TCE	Translate Control Entry
Tcl	Tool Command Language
TCP/IP	Transmission Control Protocol/Internet Protocol
TCQ	Tagged Command Queuing
TGT	Ticket Granting Ticket
TLB	Translation Lookaside Buffer
TLS	Transport Layer Security
TOS	Type Of Service
TPC	Transaction Processing Council
TPP	Toward Peak Performance
TSE	Text Search Engine
TSE	Text Search Engine
TTL	Time To Live
UCS	Universal Coded Character Set
UDB EEE	Universal Database and Enterprise Extended Edition
UDF	Universal Disk Format
UDI	Uniform Device Interface
UIL	User Interface Language
ULS	Universal Language Support
UNI	Universal Network Interface
UP	Uniprocessor
USB	Universal Serial Bus
USLA	User-Space Loader Assistant
UTF	UCS Transformation Format
UTM	Uniform Transfer Model
UTP	Unshielded Twisted Pair
UUCP	UNIX-to-UNIX Communication Protocol
VACM	View-based Access Control Model
VESA	Video Electronics Standards Association
VFB	Virtual Frame Buffer
VG	Volume Group
VGDA	Volume Group Descriptor Area
VGSA	Volume Group Status Area
VHDCI	Very High Density Cable Interconnect
VIPA	Virtual IP Address
VLAN	Virtual Local Area Network
VMM	Virtual Memory Manager

VP	Virtual Processor
VPD	Vital Product Data
VPN	Virtual Private Network
VSD	Virtual Shared Disk
VSM	Visual System Manager
VSS	Versatile Storage Server™
VT	Visualization Tool
WAN	Wide Area Network
WBEM	Web-based Enterprise Management
WLM	Workload Manager
WTE	Web Traffic Express
XCOFF	Extended Common Object File Format
XIE	X Image Extension
XIM	X Input Method
XKB	X Keyboard Extension
XLF	XL Fortran
XML	Extended Markup Language
XOM	X Output Method
XPM	X Pixmap
XSSO	Open Single Sign-on Service
XTF	Extended Distance Feature
XVFB	X Virtual Frame Buffer

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 on ordering these publications, see “How to get IBM Redbooks” on page 259. Note that some of the documents referenced here may be available in softcopy only.

- *Advanced POWER Virtualization on IBM eServer p5 Servers Architecture and Performance Considerations*, SG24-5768 available in December 2004
- *A Practical Guide for Resource Monitoring and Control*, SG24-6615
- *Linux Applications on pSeries*, SG24-6033
- *Managing AIX Server Farms*, SG24-6606
- *The Complete Partitioning Guide for IBM @server pSeries Servers*, SG24-7039
- *Effective System Management Using the IBM Hardware Management Console for pSeries*, SG24-7038

Other publications

These publications are also relevant as further information sources:

- The following types of documentation are located through the Internet at the following URL:

  `http://www.ibm.com/servers/eserver/pseries/library`

  - User guides
  - System management guides
  - Application programmer guides
  - All commands reference volumes
  - Files reference
  - Technical reference volumes used by application programmers

Online resources

These Web sites and URLs are also relevant as further information sources:

- AIX 5L operating system maintenance packages downloads

  http://www.ibm.com/servers/eserver/support/pseries/aixfixes.html
- Autonomic computing on IBM @server pSeries servers

  http://www.ibm.com/autonomic/index.shtml
- Ceramic Column Grid Array (CCGA), see IBM Chip Packaging

  http://www.ibm.com/chips/micronews
- Copper circuitry

  http://www.ibm.com/chips/technology/technologies/copper/
- Frequently asked SSA-related questions

  http://www.storage.ibm.com/hardsoft/products/ssa/faq.html
- Hardware documentation

  http://publib16.boulder.ibm.com/pseries/en_US/infocenter/base/
- IBM @server Information Center

  http://publib.boulder.ibm.com/eserver/
- IBM @server pSeries and RS/6000 microcode update

  http://techsupport.services.ibm.com/server/mdownload2/download.html
- IBM @server pSeries support

  http://www.ibm.com/servers/eserver/support/pseries/index.html
- IBM @server support: Tips for AIX administrators

  http://techsupport.services.ibm.com/server/aix.srchBroker
- IBM Linux news: Subscribe to the Linux Line

  https://www6.software.ibm.com/reg/linux/linuxline-i
- Information about UnitedLinux for pSeries from Turbolinux

  http://www.turbolinux.co.jp
- IBM online sales manual

  http://www.ibmlink.ibm.com
- Linux for IBM @server pSeries

  http://www.ibm.com/servers/eserver/pseries/linux/
- Microcode Discovery Service

  http://techsupport.services.ibm.com/server/aix.invscoutMDS

- POWER4 system micro architecture, comprehensively described in the *IBM Journal of Research and Development*, Vol 46 No.1 January 2002

  http://www.research.ibm.com/journal/rd46-1.html
- SCSI T10 Technical Committee

  http://www.t10.org
- Silicon-on-insulator (SOI) technology

  http://www.ibm.com/chips/technology/technologies/soi/
- SSA boot FAQ

  http://www.storage.ibm.com/hardsoft/products/ssa/faq.html#microcode
- SUSE LINUX Enterprise Server 8 for pSeries information

  http://www.suse.de/us/business/products/server/sles/i_pseries.html
- The LVT is a PC based tool intended assist you in logical partitioning

  http://www-1.ibm.com/servers/eserver/iseries/lpar/systemdesign.htm

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

D

E

F

H

I

L

M

N

O

P

R

S

T

U

V

W

Advanced POWER Virtualization on IBM @server p5 Servers: Introduction and Basic Configuration

Advanced POWER Virtualization on IBM eserver p5 Servers: Introduction and Basic Configuration

Includes basic configuration of clients, Virtual I/O Server, and PLM

Useful worksheets included for documentation and reference

AIX partition, Virtual I/O Server, and HMC command examples

This IBM Redbook provides an introduction to Advanced POWER Virtualization on IBM eServer p5 servers.

The Advanced POWER Virtualization feature is a combination of hardware and software that supports and manages the virtual I/O environment on POWER5 systems. The main technologies are:

- Virtual Ethernet
- Shared Ethernet Adapter
- Virtual SCSI Server
- Micro-Partitioning technology
- Partition Load Manager

The primary benefit of Advanced POWER Virtualization is to increase overall utilization of system resources by allowing only the required amount of processor and I/O resource needed by each partition to be used.

This redbook is also designed to be used as a reference tool for System Administrators who manage IBM eServer p5 servers. It provides detailed instructions for:

- Configuring and creating partitions using the HMC
- Installing and configuring the Virtual I/O Server
- Creating virtual resources for partitions
- Installing partitions with virtual resources

While the discussion in this publication is focused on p5 hardware and the AIX 5L operating system, the basic concepts can be extended to the i5/OS and Linux operating systems as well as the i5 platform.

A basic understanding of logical partitioning is required.

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

SG24-7940-00 ISBN 0738490814