Information Management

IBM

Best Practices for DB2 on AIX 6.1 for POWER Systems

- Explains partitioning and virtualization technologies for Power Systems
- Discusses DB2 performance optimization on System p
- Covers OLTP and data warehouse workloads

Rakesh Dash
Bernard Goelen
Vasfi Gucer
Rajesh K Jeyapaul
Sunil Kamath
Naveen Kumar Bharatha
Michael Kwok
Anupama Padmanabhan
Punit Shah
Basker Shanmugam
Sweta Singh
Amar Thakkar
Adriana Zubiri

Redbooks

ibm.com/redbooks

International Technical Support Organization

Best Practices for DB2 on AIX 6.1 for POWER Systems

April 2010

SG24-7821-00

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

First Edition (April 2010)

This edition applies to DB2 Version 9.5 and Version 9.7, AIX Version 6.1, and VIOS Version 2.1.2.

Contents

Figures

Tables

Examples

Notices

Trademarks

IBM, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corporation in the United States, other countries, or both. These and other IBM trademarked terms are marked on their first occurrence in this information with the appropriate symbol (® or ™), indicating US registered or common law trademarks owned by IBM at the time this information was published. Such trademarks may also be registered or common law trademarks in other countries. A current list of IBM trademarks is available on the Web at http://www.ibm.com/legal/copytrade.shtml

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

- Active Memory™
- AIX 5L™
- AIX®
- BladeCenter®
- Blue Gene®
- DB2®
- DS4000®
- DS6000™
- DS8000®
- eServer™
- GPFS™
- HACMP™
- i5/OS®
- IBM®
- InfoSphere™
- Micro-Partitioning™
- Optim™
- Power Architecture®
- POWER Hypervisor™
- Power Systems™
- POWER4™
- POWER5™
- POWER6®
- PowerHA™
- PowerPC®
- PowerVM™
- POWER®
- pSeries®
- pureXML®
- Redbooks®
- Redpaper™
- Redbooks (logo) ®
- RS/6000®
- solidDB®
- System i5®
- System i®
- System p5®
- System p®
- System Storage™
- Tivoli®
- TotalStorage®
- WebSphere®
- Workload Partitions Manager™

The following terms are trademarks of other companies:

ITIL is a registered trademark, and a registered community trademark of the Office of Government Commerce, and is registered in the U.S. Patent and Trademark Office.

Snapshot, and the NetApp logo are trademarks or registered trademarks of NetApp, Inc. in the U.S. and other countries.

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

Windows, 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, or service names may be trademarks or service marks of others.

Preface

This IBM® Redbooks® publication presents a best practices guide for DB2® and InfoSphere™ Warehouse performance on a AIX® 6L with Power Systems™ virtualization environment. It covers Power hardware features such as PowerVM™, multi-page support, Reliability, Availability, and Serviceability (RAS) and how to best exploit them with DB2 LUW workloads for both transactional and data warehousing systems.

The popularity and reach of DB2 and InfoSphere Warehouse has grown in recent years. Enterprises are relying more on these products for their mission-critical transactional and data warehousing workloads. It is critical that these products be supported by an adequately planned infrastructure. This publication offers a reference architecture to build a DB2 solution for transactional and data warehousing workloads using the rich features offered by Power systems.

IBM Power Systems have been leading players in the server industry for decades. Power Systems provide great performance while delivering reliability and flexibility to the infrastructure.

This book presents a reference architecture to build a DB2 solution for transactional and data warehousing workloads using the rich features offered by Power systems. It aims to demonstrate the benefits DB2 and InfoSphere Warehouse can derive from a Power Systems infrastructure and how Power Systems support these products.

The book is intended as a guide for a Power Systems specialist to understand the DB2 and InfoSphere Warehouse environment and for a DB2 and InfoSphere Warehouse specialist to understand the facilities available for Power Systems supporting these products.

The team who wrote this book

This book was produced by a team of specialists from around the world.

Rakesh Dash is a performance engineer for the ECM Performance Engineering team with IBM. He has been working with IBM for more than five years and has performance consulting and development experience for several IBM products such as DB2, AIX, WebSphere®, IBM Java™, IBM filenet P8. He has authored several white papers on performance best practices and presented the performance best practices methodology in conferences.

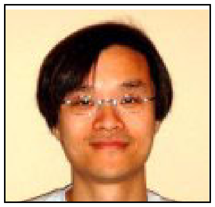
Bernard Goelen is an Advisory IT Specialist in Luxemburg with more than 15 years experience with IBM and business partners. He works on national and international deals, on Power Systems platforms, with extensive experience in AIX, PowerHA™, and Power Virtualization techniques, Live Partition Mobility. He has a key role within IBM Global Technology Services and has an extended role within the IBM System Sales Implementation Services department where he acts as a pre-sales and technical support on large deals. He focuses on server consolidation and IT virtualization and has field experience in virtualization solutions implementations, PowerVM and PowerHA clusters deployments in complex environments, performance audits, and analysis.

Vasfi Gucer is a project leader at the ITSO in Austin, Texas. He has more than 15 years of experience in Systems Development and Enterprise Systems Management. He writes extensively and teaches IBM classes worldwide on Tivoli® software. Vasfi is also an IBM Certified Senior IT Specialist, PMP, and ITIL® Expert.

Rajesh K Jeyapaul is an AIX Development Support Specialist in IBM India. He is specialized in investigating the performance impact of the applications on AIX/System p®. Currently, he is a Technical Lead for AIX Development Support team. He has co-authored an IBM Redbooks publication on DS8000® performance Monitoring. His area of expertise include Power Virtualization, AIX, and High-Availability Cluster Multi-Processing (HACMP™). He holds a Masters Degree in Software Systems from the University of BITS, India and an MBA from University of MKU, India.

Sunil Kamath is a Senior Technical Staff Member and Senior Manager in Information Management at the IBM Toronto Software Labs. He has responsibility for performance, benchmarking, and solutions development for DB2 and solidDB®. Sunil works primarily on database performance and scalability and has led many successful world-record TPC-C and SAP benchmarks. In addition, he has designed and implemented many high performance DB2 database solutions for customers world-wide. Sunil is also the DB2 kernel architect for exploiting hardware, virtualization, operating systems, storage, and compiler technologies in DB2.

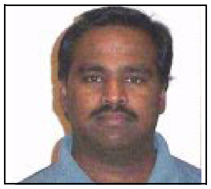
Naveen Kumar Bharatha is an independent database consultant. He has 13 years of IT experience in Database, Systems, and Storage design. His areas of expertise and responsibilities include design of single and multi-partition databases, database administration, virtualization and automation to provide DB2 solutions. He has performed several database and systems migrations in complex environments. He has special interest in creating a Linux® Live OS with persistent DB2 Database. He has worked as Staff Software Engineer with IBM Lab services focusing on customer engagements, POCs involving DB2. He is an IBM certified Database Administrator in DB2 UDB v8.1. He holds a Bachelors degree from JNTU, Hyderabad, India.



Michael Kwok is an Advisory Software Engineer in DB2 for Linux, UNIX®, and Windows®. He works in the DB2 performance QA team as a Data Warehouse Team Lead. His main focus is the overall performance experience in Data Warehouse workloads. He provides technical support on customers' performance problems. Michael Kwok received his Ph.D. degree in Computer Science. His doctoral dissertation is in the area of scalability analysis of distributed systems.

Anupama Padmanabhan works as a Technical Consultant in the ISV Solutions Enablement Group for the enablement of SWG family of products on IBM eServer™ platforms. She has 10 years of experience with IT involving leadership and consulting on financial systems, extensive experience in full software development life cycles, including requirements definition, prototyping, design, coding, testing and maintenance. She holds a Bachelor's degree in Electronics Engineering from University of Bombay, India.

Punit Shah has a background in both DB2 internal and AIX, POWER®, and virtualization technologies. In his current role, Punit is in the DB2 development organization, engaged in DB2 kernel data protection services and backup/restore design and development for DB2 pureScale feature. In his previous role, Punit was architect and leader of System p/IBM Software group enablement team. There, he was responsible for DB2/System p/AIX stack enablement, performance and deep technical exploitation, ensuring that DB2 (and other IBM software products) are using latest AIX and System p technologies, including PowerVM virtualization feature. He architected and implemented several joint DB2 and System p virtualization solutions and features in DB2. In the process, he applied for several patents in the area, and authored around dozen white papers. Other AIX, POWER technologies he enabled in DB2 are decimal floating point (DFP), AIX multi-page support, and JFS2 concurrent I/O (CIO).



Basker Shanmugam works in the IBM DB2 Performance and Solution Development organization focusing on DB2 performance, benchmarking, and developing DB2 solutions with System p virtualization features. He has published several TPC benchmarks and has co-authored several papers on DB2 with System p virtualization. Basker has worked in technical support and performance-related areas that involve Windows, Linux and AIX operating systems, as well as Oracle, DB2 and server virtualization for more than 19 years.

Sweta Singh is a DB2 Performance Analyst. She works in the DB2 Performance QA Team and has worked extensively in analyzing and improving performance of DB2 codebase over the past six years. She is currently the technical lead for OLTP and focuses on improving the overall customer experience in OLTP. She is also actively involved in customer situations related to DB2 performance.

Amar Thakkar is the Team Lead of the DB2 Down Systems Support team in Sydney, Australia. He has over five years experience in diagnosing critical problems and ensuring that clients database downtime is minimized. Amar has extensive knowledge in the areas of migration, memory usage, backup, recovery, and performance-related issues. He enjoys investigating complex issues and resolving them to ensure their impact on business is mitigated.

Adriana Zubiri is the technical manager for the DB2 Performance Quality Assurance team in the IBM Canada lab. She has been part of the DB2 development organization for more than 12 years, where she has led the customer support team, participated in world-record TPCH Benchmarks and designed and implemented new features in the area of join processing. She has worked extensively with customers and specialized in the performance area for data warehouses. She holds a Masters of Science degree in Computing Science from University of Alberta, Canada.

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

Guido Somers
IBM Belgium

Rimas Kalesnykas, Konwen Kuan, Heidi Scott, Malcolm Singh
IBM Canada

Abhilash J Jils, Prabhu S Padashetty
IBM India

Doreen Stein
IBM Germany

Dan Braden, Grover C. Davidson, Brian Hart, James Hoy, Bart Jacob, Pete Jordan, Jessica Scholz, Jaya Srikrishnan, Stephen M Tee, Brian Twichell, Scott Vetter
IBM USA

David Hepple
IBM Ireland

The team would like to express special thanks to Robert Taraba from IBM USA for providing the project resources and his support throughout the project.

Now you can become a published author, too!

Here's an opportunity to spotlight your skills, grow your career, and become a published author - all at the same time! Join an ITSO residency project and help write a book in your area of expertise, while honing your experience using leading-edge technologies. Your efforts will help to increase product acceptance and customer satisfaction, as you expand your network of technical contacts and relationships. Residencies run from two to six weeks in length, and you can participate either in person or as a remote resident working from your home base.

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 books to be as helpful as possible. Send us your comments about this book or other IBM Redbooks publications in one of the following ways:

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

  ibm.com/redbooks

- Send your comments in an e-mail to:

  redbooks@us.ibm.com

- Mail your comments to:

  IBM Corporation, International Technical Support Organization
  Dept. HYTD Mail Station P099
  2455 South Road
  Poughkeepsie, NY 12601-5400

Stay connected to IBM Redbooks

- Find us on Facebook:

  http://www.facebook.com/pages/IBM-Redbooks/178023492563?ref=ts

- Follow us on twitter:

  http://twitter.com/ibmredbooks

- Look for us on LinkedIn:

  http://www.linkedin.com/groups?home=&gid=2130806

- Explore new Redbooks publications, residencies, and workshops with the IBM Redbooks weekly newsletter:

  https://www.redbooks.ibm.com/Redbooks.nsf/subscribe?OpenForm

- Stay current on recent Redbooks publications with RSS Feeds:

  http://www.redbooks.ibm.com/rss.html

1

Introduction

This IBM Redbooks publication describes the best practices to configure a DB2 database server for data warehouse or OLTP environment on a Power System architecture-based system running AIX 6.1.

Running the DB2 product on a Power System using the right blend of virtualization features and meeting your business goals is a real challenge that allows you to reduce various costs (such as power, floor space, and administration) by consolidating your servers. Sharing your system resources, dynamically resource allocation without rebooting, and processor utilization are key features that help you optimize the performances of your DB2 product.

The target audience of this book is database administrators and system administrators who want to optimize the performance of their DB2 database server for data warehouse or OLTP environments on AIX V6.1. It outlines those particular parameters you can set to follow the best practices when configuring a DB2 environment on AIX.

Moreover, this book is designed to help you understand the major differences that exist when running your database on AIX 5L™ compared to running your database on AIX 6.1. This new version of AIX introduces many new features, including workload partitions, advanced security, continuous availability, and managing and monitoring enhancements.

DB2 9.7 also offers a wide range of new features, including autonomics, data compression, pureXML®, automatic storage, performance optimization and security, reliability, and scalability. Most of these features are covered in this book.

This chapter makes an introduction to the concepts that are detailed throughout the book and contains the following sections:

1.1 Introduction to Power Systems

When setting up a Power Systems environment, there are many options that you must consider and plan for properly to achieve the optimal performance goal. Especially for a database server, the database system is the primary application in the system. Database configuration considerations must also be incorporated in the Power Systems planing setup. However, the Power Systems setup tasks are usually performed by the system support team without the database administrator's (DBA) involvement, or the DBA might not have the expertise in this area. The following list details the main points you go through when configuring your logical partitions on a Power System for your database server:

- Setting up and configuring logical partitions
  - Choosing a partition type
  - Configuring physical and virtual processors
  - Configuring the memory
- Considerations for disk storage and choosing from various options
- Considerations for the network and choosing from various options Setting up the Virtual I/O Server (VIOS).

1.1.1 POWER Architecture

POWER Architecture technology is an instruction-set architecture (ISA) that spans applications from consumer electronics (such as gaming consoles and set-top boxes), Internet-enabling technologies (such as routers and switches) to the world's fastest supercomputer, and even Spirit and Opportunity, the Mars rover supercomputers. The first computers from IBM to incorporate the POWER Architecture were called the “RISC System/6000" or RS/6000®.

Note: POWER stands for Performance Optimization With Enhanced RISC.

Power Architecture® refers both to POWER processors used in IBM servers and to PowerPC® processors, which can be found in a variety of embedded systems and desktops.

Note: PowerPC stands for POWER Performance Computing.

Power Architecture encompasses PowerPC, POWER4™, POWER5™ and POWER6® processors.

- Power Architecture drives the world's highest-performance supercomputers, including the #1 Blue Gene®.
- Power processor technology is at the heart of the Power System i® and Power System p, IBM servers.
- The most robust storage solutions, including the IBM TotalStorage® DS6000™ and IBM TotalStorage DS8000, incorporate Power Architecture processors.
- Power Architecture is the leading processing platform for wireless infrastructure, enterprise routing and switching, and wireline telecommunications.
- VMX (vector multimedia extension) implementations of PowerPC that are key to the gaming industry.
- Reliability and performance have made Power Architecture chips a favorite choice for the aerospace industry.
- The POWER processor-based systems offer Virtualization Engine capabilities such as IBM Micro-Partitioning™ technology, which enables partitions as small as 1/10th of a processor and automated, dynamic resource management between partitions.

As of the writing of this book, the latest in the series of POWER processor is POWER6 processor. The POWER6 chip is a dual-core processor that runs at speeds between 4 GHz and 5 GHz depending on the type and model of the system. Figure 1-1 on page 5 shows the POWER6 architecture.

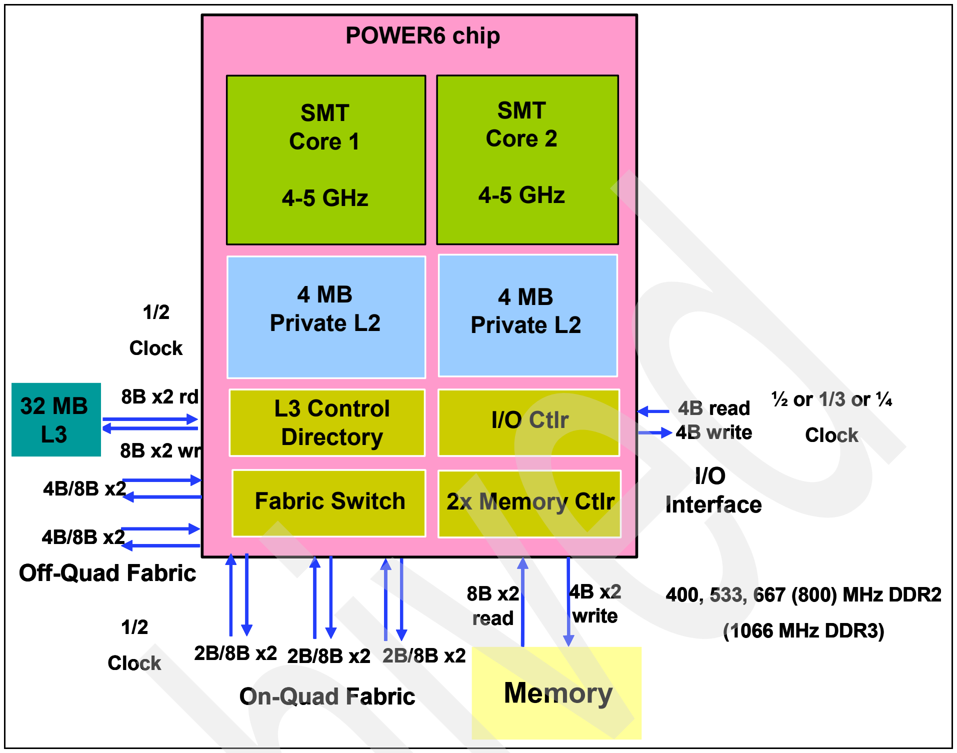


Figure 1-1 POWER6 architecture

The following are the characteristics of the POWER6 architecture:

- The POWER6 processor implements the 64-bit IBM Power Architecture technology.
- The POWER6 core is a ultra-high-frequency design that is optimized for performance for the server market.
- POWER6 processor core includes two accelerators for increasing the performance of specific workloads
  - Vector Multimedia extension (VMX) provides vector acceleration of graphic and scientific workloads.
  - Decimal Floating point unit (DFU) provides acceleration of commercial workloads, more specifically, financial transactions.

Note: The IBM POWER6 processor is the first commercial hardware implementation of the IEEE 754R Floating-Point Arithmetic Standard that defines decimal floating-point formats, on which substantial amount of work is done to improve the performance of commercial workloads.

- The POWER6 processor implements a two-thread SMT for each core.
- The POWER6 processor design reduces the effective latency of main memory storage references with a substantial set of design features, such as three levels of cache, SMT, and Load Lookahead (LLA). It enhances data stream prefetching.
- The POWER6 processor-based systems introduce new features such as Processor Instruction Retry, which allows the system to recover transparently without an impact on a partition using the core.
- Dynamic Processor Deallocation enables automatic deconfiguration of an error-prone processor core before it causes an unrecoverable system error.
- POWER6 processor-based systems have a number of protection schemes, such as hardware scrubbing and ECC ,which are designed to prevent, protect, or limit the effect of errors in main memory.
- POWER6 processor-based systems are designed with cache protection mechanisms.
- PCI Error Recovery been extended to PCIe buses as well in POWER6.

In summary, with its high-frequency core architecture, enhanced SMT capabilities, balanced system throughput, and scalability and extensions, the POWER6 microprocessor provides a higher level of performance. Improvements in functionality, RAS (Reliability, Availability, Serviceability), and power management have resulted in valuable new characteristics of POWER6 processor-based systems.

1.2 Introduction to virtualization

Nowadays, IT Managers are concerned with the total cost of ownership (TCO) to run an IT environment (such as the hardware costs, floor space the hardware takes in a computer room, real estate prices, the power consumption, the heat produced). Moreover, reliability, availability, and serviceability are other aspects that everyone is concerned with.

Virtualization can help you consolidate your multiple systems, running multiple environments and applications on a single system, all into separated environments, and fully secured, as though you were running on an isolated single standalone servers.

Virtualization is not new to IBM. IBM developed virtualization techniques in the 1960s. You can see the complete story on virtualization in Figure 1-2 on page 7.

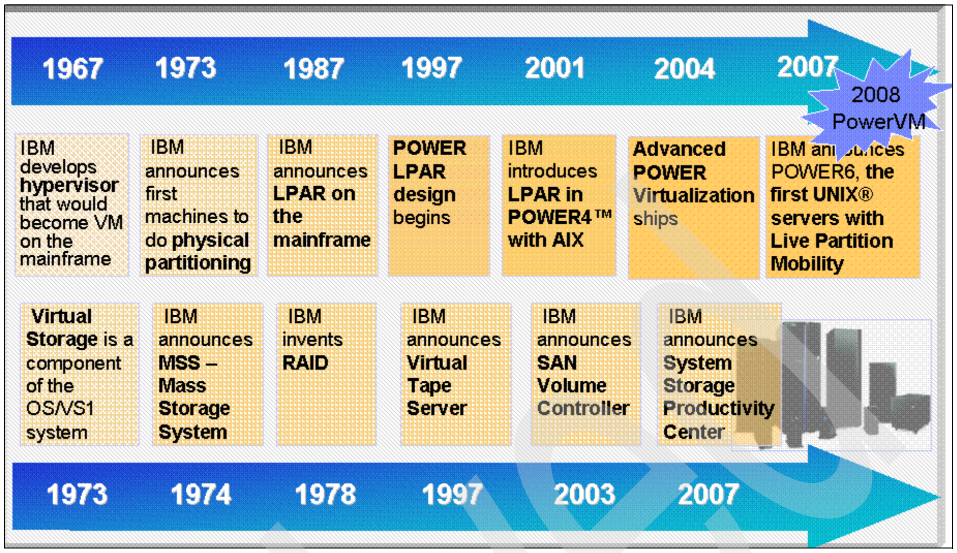

1967
1973
1987
1997
2001
2004
2007
2008 PowerVM
IBM develops hypervisor that would become VM on the mainframe
IBM announces first machines to do physical partitioning
IBM announces LPAR on the mainframe
POWER LPAR design begins
IBM introduces LPAR in POWER4™ with AIX
Advanced POWER Virtualization ships
IBM announces POWER6, the first UNIX® servers with Live Partition Mobility
Virtual Storage is a component of the OS/VS1 system
IBM announces MSS – Mass Storage System
IBM invents RAID
IBM announces Virtual Tape Server
IBM announces SAN Volume Controller
IBM announces System Storage Productivity Center
1973
1974
1978
1997
2003
2007

Figure 1-2 Virtualization history

Virtualization is no longer an option for most data centers. Recent developments and shifts in the marketplace, coupled with high administration cost and energy cost, have elevated virtualization to a must-have deployment strategy. Virtualization offers functions and benefits that allow you to share a physical resource, consolidate multiple separated resources, and emulate hardware resources.

Sharing a resource to create multiple images (such as LPARs, virtual disks, virtual LANs [VLAN]) offers you the benefit of global resource use, which increases flexibility, and eases the management of your infrastructure, all of which takes place in a complete isolated environment.

Aggregation allows you to pool multiple separated distributed resources and make them appear as one single resource to the client. Virtual disk, SVC SAN Volume Controller, and RAID technology are resources that ease management, help to protect your investment, and offer a high level of scalability.

The emulation function allows you to make use of objects (such as virtual tape, iSCSI) that seem to be real (although no physical resource exists). It helps in the compatibility and interoperability mode of your resources and the flexibility of your environment.

Note: DB2 has been engineered to integrate well into virtualized environments.

1.2.1 Why virtualize servers?

Virtualization helps you to create a virtual machine easily, allowing rapid deployment of your new application using logical partitions. The overall cost of your hardware decreases by consolidating your smaller systems onto one bigger one.

When you increase the energy efficiency of your infrastructure by decreasing your power consumption, you meet environmental goals. Moving logical partitions from one server to another enables one of the two servers to power off and so save energy.

Virtualization also offers the flexibility to maximize capacity, allowing you to move your resources with DLPAR operations or LPM (Live Partition Mobility) capabilities.

What virtualization offers you is a consolidated infrastructure on a larger server on which you can maximize your resources, host numerous virtual machines that you can build with just a click, dynamically provision, and host. Moreover, it gives you the illusion that you have more hardware resources than really exist on the physical machine. It offers you totally isolated virtual machines run-time environments, allowing a mix of operating systems with greater manageability and efficiency through balancing your system resources to virtual machines that need resources the most. You can keep your SLAs (service level agreements) and increase the availability and security of your servers, lower the cost of availability, and protect your investment.

Note: Carefully consider planning before moving to a virtualized environment. Communicate with your development team and DBAs so that all work in concert with IT to make the virtualization a success. Do not make virtualization more complex than necessary.

1.2.2 Virtualization benefits

Virtualization helps you reduce your hardware costs. When the physical resources use goes higher, the footprints reduce drastically. Moreover, it improves flexibility and responsiveness because your virtual resources can be dynamically adjusted.

Virtualization is the key enabler of on-demand operating environments. Due to having fewer physical servers to manage, many management tasks become much easier. Also, the management cost is reduced.

Note: Application software licensing costs can decrease when virtualizing your environment on a consolidated server.

For example, the business requests to deploy a new application on a new server. The problem is that you have no server available and need to order one, which takes time to get, because you need approvals prior to place your order.

In a virtualized consolidated environment, it is as simple as a click to create the requested server. This happens without any downtime because the hardware is already there and managed by the system administrators.

Virtualization can be categorized as follows: (see Figure 1-3 on page 10)

- Full hardware and firmware embedded virtualization

  With Power Systems, all virtualization functions are implemented into the hardware and firmware. The firmware is called the Power Hypervisor. See 1.5, “Introduction to PowerVM virtualization” on page 21 for more information about Power Hypervisor.

- Full virtualization

  Full virtualization is a virtualization technique used to provide a certain kind of virtual machine environment, namely, one that is a complete simulation of the underlying hardware. In such an environment, any software capable of execution on the raw hardware can be run in the virtual machine and, in particular, any operating systems

  Full virtualization provides necessary abstractions for the application and therefore does not require any changes to the application or OS.

- Para virtualization

  Para virtualization is a technique that uses an OS-based software interface to virtual machines and requires OS changes, but offers a performance benefit on older hardware.

- OS-based virtualization

  OS-based virtualization is a technique that allows virtual images to be created off a single OS instance.

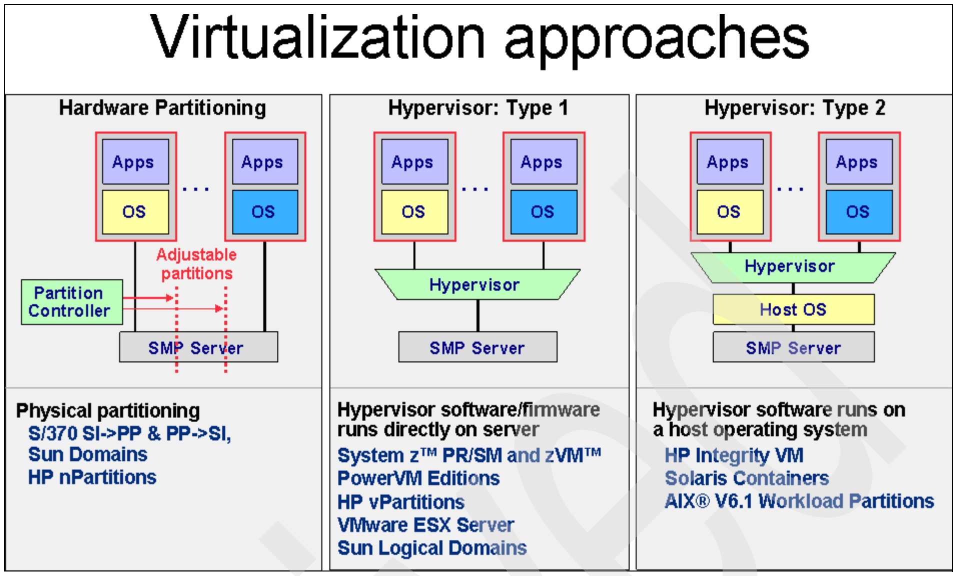


Figure 1-3 Other virtualization approaches

Note: Virtualization concerns the server, the storage, and the network.

1.3 Introduction to AIX 6.1

AIX is an open standards-based UNIX OS that is designed to comply with the Open Group's Single UNIX Specification Version 3. It provides the enterprise-class IT infrastructure for thousands of clients around the world.

On System p servers, mixed environments are supported: AIX 5L V5.2 ML2 and AIX 5L V5.3 partitions with dedicated processors and adapters, and AIX 5L V5.3 partitions using micro-partitioning and virtual devices. AIX 5L V5.3 partitions can use physical and virtual resources at the same time.

AIX 5L is supported on the System p servers in partitions with dedicated processors (LPARs), and shared-processor partitions (micro-partitions).

For System p servers configured with one of the PowerVM features, AIX 5L Version 5.3 or later is required for micro-partitions, virtual I/O, and virtual LAN.

1.3.1 IBM AIX V6.1

AIX V6.1 is the latest version of the AIX operating system, which includes new and improved capabilities for virtualization, security features, continuous availability features, and manageability. AIX V6.1 is the first generally available version of AIX V6.

The following list details on features which AIX V6.1 supports:

- PowerVM AIX 6 Workload Partitions (WPAR), software-based virtualization
- Live Partition Mobility (LPM), with the IBM PowerVM AIX 6 Workload Partitions Manager™ for AIX
- 64-bit kernel for higher scalability and performance
- Dynamic logical partitioning and micro-partitioning support
- Support for Multiple Shared-Processor Pools
- Trusted AIX, multilevel, compartmentalized security
- Integrated Role Based Access Control
- Encrypting JFS2 (Enhanced Journaled File System 2) file system
- Kernel exploitation of POWER6 Storage Keys for greater reliability
- Robust journaled file system and Logical Volume Manager (LVM) software including integrated file system snapshot
- Tools for managing the systems environment
- System Management Interface Tool (SMIT) and the IBM Systems Director Console for AIX

AIX 6.1 runs on systems based on POWER4, PPC970, and POWER5 processor-based plat-forms, but the most capability is delivered on systems built with the new POWER6 processors.

The AIX OS is designed for the IBM Power System p, System i, System p5®, System i5®, eServer p5, eServer pSeries®, and eServer i5 server product lines, as well as IBM BladeCenter® blades based on Power Architecture technology and IBM IntelliStationPOWER workstations.

Table 1-1 provides the AIX V6.1 new features summary information.

Table 1-1 AIX V6.1 new features summary

Features	**Functionality**	**Benefits**
Virtualization		
PowerVM Workload Partitions	WPARs enable the creation of multiple virtual AIX6.1 environments inside of a singleAIX6.1 instance.	Reduced administration and improved system efficiency.
Live Application Mobility	Workload Partitions can be moved from one system to another without restarting the application or causing significant disruption to the application user.	Increased application availability, enhanced workload manageability and energy savings.
Security		
Role Based Access Control (RBAC)	*RBAC* provides improved security and manageability by allowing administrators to grant authorization for management of specific AIX6.1resources to users other than root.	Improved security, decreased administration costs.
Encrypted FileSystem	The IBM Enhanced Journaled Filesystem Extended (JFS2) adds even greater data security with the capability to encrypt the data in a file system.	Improved security.
Trusted AIX	*Trusted AIX* extends the security capabilities of the AIX OS by integrating compartmentalized, multilevel security (MLS) into the base operating system to meet critical government and private industry security requirements.	Highest level of security for critical government and business workloads.

Features	Functionality	Benefits
Secure by default (Installation option)	Enables only the minimal number of system and network services to provide the maximum amount of security.	Improved security on initial installations of AIX6.1.
Trusted Execution	Verifies the integrity programs at execution time.	Improved security.
Support for Long Pass Phrases	AIX 6.1 and AIX 5.3 Technology Level 7 supports greater than eight character passwords for authentication of users.	Improved security.
Continuous Availability		
Concurrent AIX kernel update	Provides a new capability to deliver kernel updates as interim fixes that do not require system reboot to put into effect.	Improved AIX availability.
Storage keys	Storage keys can reduce the number of intermittent outages associated with undetected memory over-lays inside the AIX kernel and kernel extensions.	Improved AIX availability and improved application availability.
Dynamic tracing	New tracing command, ***probevue***, allows a developer or system administrator to dynamically place probes in existing application or kernel code, without requiring special source code or even recompilation.	Easier resolution to application execution and performance problems.

Features	Functionality	Benefits
Manageability		
IBM Systems Director Console for AIX	This new management interface allows administrators to manage AIX6.1 remotely through a browser.	Reduced administrative costs and improved administrative effectiveness by enabling Web-based administration across multiple AIX instances.
Automatic Variable Page Size	Automatically manage the size of pages used when it is running on a system based on POWER6 processors.	Improved performance with reduced administrative effort.
Solution Performance Tuning	The default tuning parameters for AIX6.1 have been changed to provide much better performance for most applications right out of the box.	Improved performance with reduced administrative effort.
Network Installation Manager Support f or NFSv4	Network Installation Manager (NIM) has been enhanced to provide additional security features and flexibility by enabling the use of NFS version4.	Improved performance with reduced administrative effort.

In summary, AIX6.1 and POWER6 provides innovative features for virtualization, security, systems management, and RAS features with no compromise on the performance.

1.4 Introduction to DB2

DB2 Version 9 data server is the fastest-growing, flagship database offering of IBM. It is optimized to deliver industry-leading performance across multiple workloads, while lowering administration, storage, development, and server costs. This section discusses the benefits that DB2 can provide with respect to these points.

1.4.1 Autonomics

Although costs for server management and administration can be hard to measure and might be less apparent than costs for servers, storage, and power, they represent the largest percentage of total IT spending. Refer to Figure 1-4 to see a breakdown of the various costs associated with administration.

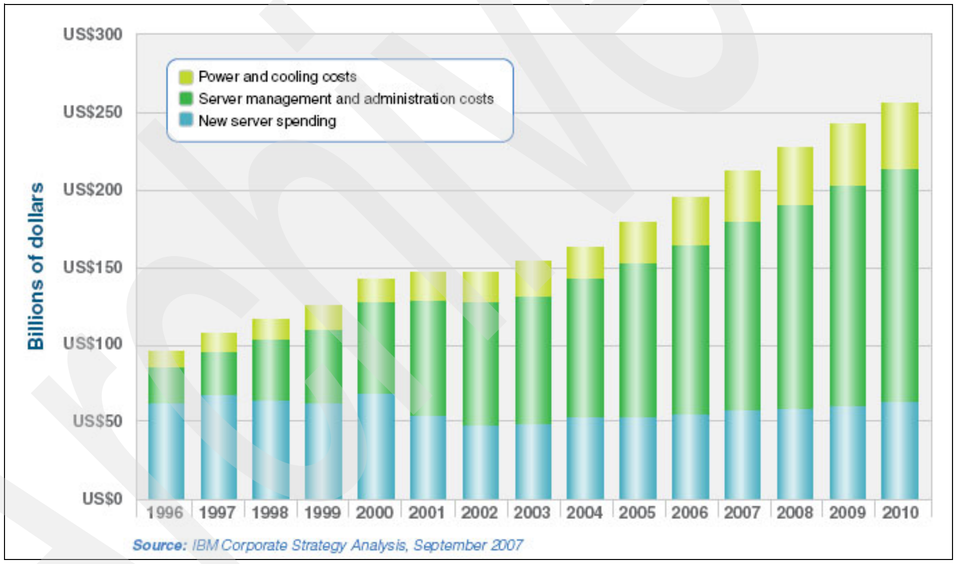


Figure 1-4 Administration costs

DB2 can help cut these administrative costs by taking care of itself as much as possible using the following capabilities:

- Self-configuring:

  DB2 automatically sets up the system and manages configuration settings.

  For example, using the Configuration Advisor (CA), several instance and database parameters, including buffer pools, are adjusted to make the database run well in your environment from the beginning. Using automatic maintenance, DB2 enables the automation of RUNSTATS, REORG and BACKUP utilities.

- Self-healing

  DB2 automatically helps resolve problems as they occur.

  Using the DB2 Health Center it is easy to set up thresholds for warnings, and alarms for database and instance level parameters. It can recommend the course of actions to follow to resolve problems. It is possible to prevent problems and ensure the databases remain always available. Note that the DB2 9 Health Monitor and Health Center are deprecated in DB2 9.7. The tools that replace these are available with the IBM Optim™ solutions.

- Self-optimizing

  DB2 reacts to changes in workloads and adjusts memory and other facets of the software to continuously improve performance

  The self-tuning memory feature in DB2 9 simplifies the task of memory configuration by automatically setting values for several memory configuration parameters at startup. The self-tuning memory manager uses intelligent control and feedback mechanisms to keep track of changes in workload characteristics, memory consumption, and demand for the various shared resources in the database. It dynamically adapts their memory usage as needed.

- Self-protecting

  DB2 addresses external security threats to the system by detecting and preventing unauthorized access

  Extensive security and audit capabilities help protect information from ever-changing threats, and address regulatory requirements such as the Sarbanes-Oxley Act. DB2 supports a robust role-based security model, enabling businesses to divide authority in compliance with data governance standards.

A combination of the following mechanisms can be used by which DB2 can provide such extensive security:

- Authentication
- Authorization
- Trusted Contexts
- Auditing
- Label Based Access Control (LBAC)
- DB2 encryption.Compression

DB2 incorporates Deep Compression technology and the DB2 Storage Optimization Feature, which can reduce storage requirements. For example, using DB2 row compression, for example, can save up to 83% of disk space on your largest tables. The most recent release, DB2 9.7, further improves storage savings by adding data compression for database indexes, temporary database tables, temporary tables, large objects, and XML documents. Accessing data from the disk is the slowest database operation. By storing compressed data on disk, fewer I/O operations need to be performed to retrieve or store the same amount of data. Therefore, for disk I/O-bound workloads, the query processing time can be improved noticeably.

Note: The DB2 Storage Optimization Feature is a licensed feature that enables you to use the DB2 compression features (which already includes deep compression). Deep compression is often referred to row-level compression.

1.4.2 pureXML

DB2 supports both relational and XML data, which can simplify development and deployment of advanced new applications. DB2 pureXML eliminates much of the work typically involved in the management of XML data, and serves XML data at unmatched speeds. Applications can mix relational and XML data as business needs dictate. DB2 9.7 adds end-to-end native XML support for both transactional and data warehouse applications, opening new opportunities to extract business value from XML data.

1.4.3 Automatic storage

DB2 9 extends the automated storage features first introduced in DB2 V8.2.2. Automatic storage automatically increases the size of your database across disks and file systems. With automatic storage management, DB2 allocates storage on demand as the table consumption grows. This feature intends to be a single point of storage management for table spaces. DBAs are no longer required to define the containers for table spaces, but specify a group of storage devices for DB2, such as file systems, in the form of storage paths. DB2 creates the necessary containers automatically across these storage paths. The growth of the existing containers and the additional new ones is managed by DB2. In DB2 9, automatic storage is enabled by default for new databases. In addition, automatic storage support has been added for multi-partition databases.

1.4.4 Performance

The DB2 Performance Optimization Feature gives insight and the ability to optimize workload execution, which can be accomplished using a combination of DB2 Workload Manager, DB2 Performance Expert, and DB2 Query Patroller. This can help reduce hardware acquisition costs by optimizing the performance of servers and postponing costly hardware upgrades. Figure 1-5 shows that between January 1, 2003 and March 23, 2009, DB2 has held certain industry benchmarks for more days than all other vendors combined.

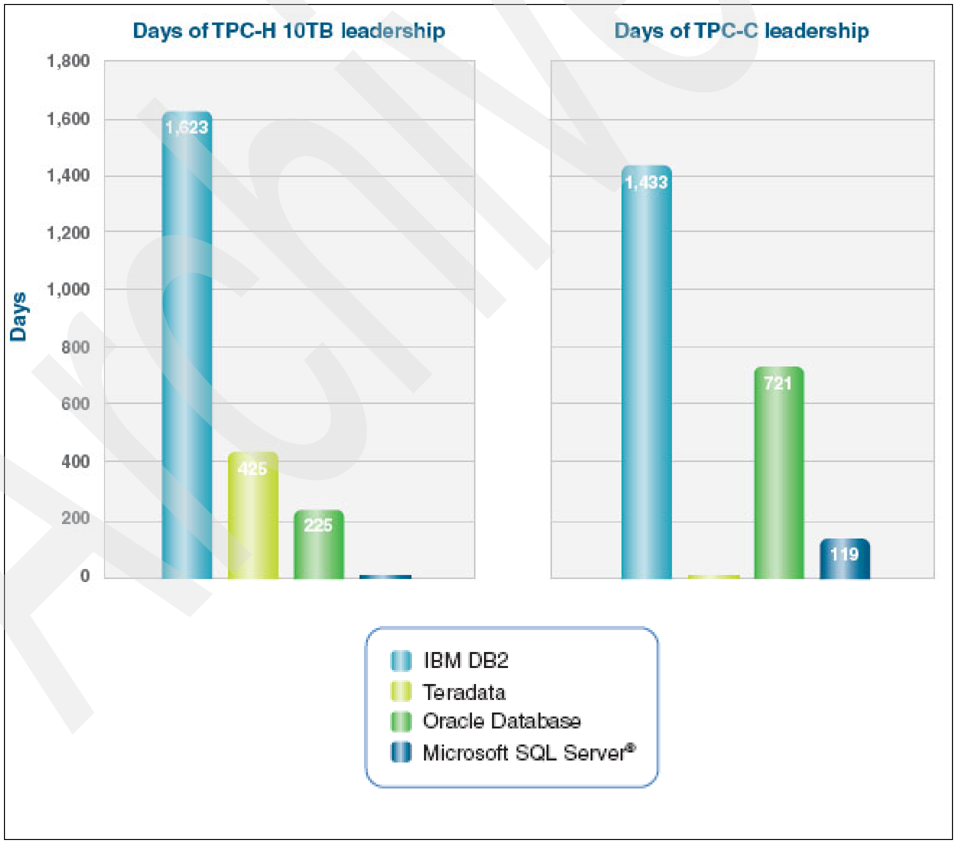


Figure 1-5 DB2 industry benchmarks

1.4.5 Reliability and scalability

DB2 helps businesses increase availability by reducing planned and unplanned outages. There are a few methodologies that can be used to ensure uninterrupted access to the database, namely DB2 Replication (also known as DataPropagator), High Availability Disaster Recovery (HADR) and DB2 pureScale. DB2 HADR allows failover to a standby system in the event of a software or hardware failure on the primary system. Online reorganization actively reconstructs the database table for better performance, while permitting uninterrupted access to the table data. DB2 pureScale, introduced in DB2 9.7 Enterprise Server Edition, offers clustering technology that helps deliver high availability and exceptional scalability transparent to applications.

Unlike other distributed shared-disk database cluster technologies, DB2 pureScale does not require administrators to perform complex tuning or update application code when scaling. Flexible application workload balancing and grouping capabilities allow multiple servers to appear as a single database. New members (DB2 server instances) can be used immediately. It is designed to simplify database and clustering administration so that members can be added to and removed from the cluster easily, enabling IT staff to scale the cluster up or down quickly to meet changing business requirements. DB2 pureScale also delivers enhanced performance by using IBM Power Systems and incorporating PowerHA pureScale technology. Figure 1-6 shows the DB2 pureScale.

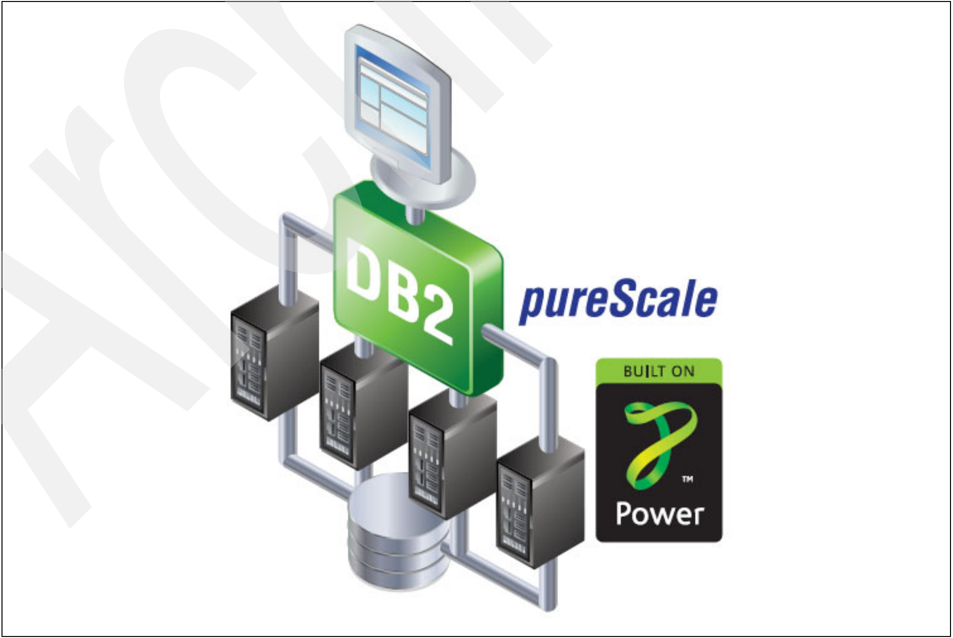


Figure 1-6 DB2 pureScale

1.4.6 Multi-vendor SQL and API support and migration to DB2

There are number of enhancements that simplify the movement of applications to DB2 and use existing skills. With DB2 9.7, differences are now the exception, not the rule, when it comes to many database and development features, including concurrency models, SQL dialects, data types, procedural languages, packages and scripting languages. Applications moved to DB2 can run with full native execution, delivering high performance.

For more information see the *SQL compatibility enhancements* document at the following Web page:

http://publib.boulder.ibm.com/infocenter/db2luw/v9r7/topic/com.ibm.db2.luw.wn.doc/doc/c0054107.html

1.4.7 DB2 in virtualization environment

Coupled with the IBM PowerVM features, DB2 9.7 is uniquely positioned to exploit virtualization. DB2 9 is enabled to use advanced virtualization features that make it an excellent choice for an information-management platform on the Power System server. DB2 works seamlessly in the Power system virtualization environment. There is no additional package or driver to install or special configuration required for either DB2 or AIX. Additionally, DB2 is aware of, and reacts to, any dynamic logical partitioning (DLPAR) event (that is, any runtime change to host-partition operating-system resources, such as a change in computing resources and physical memory). The self-tuning memory-management (STMM) feature of DB2 9 automatically adjusts and redistributes DB2 heap memory in response to dynamically changing partition memory and workload conditions. The STMM feature, coupled with LPM, allows you to move an active database that runs on a system with limited memory resources to a system that has additional memory. Then, with the help of DLPAR operation and STMM, the database can make use of this additional memory, providing increased throughput to the workload.

Given this, IBM has introduced sub-capacity (virtualization) licensing. This enables customers to use server virtualization to consolidate their infrastructure more effectively and reduce their overall total cost of ownership (TCO). It also allows flexible software licensing using advanced virtualization capabilities (such as shared processor pools, micro-partitioning, virtual machines and dynamic reallocation of resources). This new licensing gives expanding customers the flexibility to choose how to add workload environments without making trade-offs between hardware design and software licensing.

1.5 Introduction to PowerVM virtualization

PowerVM is part of the IBM Advanced Power Virtualization offering and is available on Power Systems-based servers. PowerVM helps confine your investment to your hardware. It is a combination of elements (such as hardware, firmware and software) that provide virtualization for your CPU, your network, and your disks.

Partitions can have dedicated or shared processor resources. With shared resources, PowerVM can automatically adjust pooled processor resources across multiple operating systems, borrowing processing power from idle partitions to handle high transaction volumes in other partitions.

Key components are as follows:

- The hardware

  POWER6 hardware systems provide the hardware component of PowerVM or POWER6 hardware systems are the hardware component of PowerVM.

- Power Hypervisor

  Power Hypervisor is the foundation for virtualization on Power Systems, enabling the hardware to be divided in multiple logical partitions, and ensuring complete isolation between them. Power Hypervisor is designed to provide business-critical availability. It is always active and cannot be de-activated. It is responsible to dispatch the workload across the physical shared processors, and provide inter-partition communication enabling the VIOS's virtual SCSI and virtual Ethernet function.

- VIOS

  VIOSs are enhanced AIX servers with the capability to virtualize I/O adapters, such as Ethernet cards and Fibre Channel cards. The virtualization of those elements enables you to benefit fully from your hardware investment.

Moreover, PowerVM extends the base system functions of your server to include the following capabilities:

- Micro-partitioning

  Micro-partitioning enables a processor to be divided to a tenth of a whole CPU and is available from POWER5 systems, running AIX 5.3 or later. It adds flexibility to manage your processors efficiently to create virtually partitions up to ten times the number of available processors.

- Live partition mobility

  This capability enables you to move a running logical partition (known as Active partition mobility) physically from one physical server to another, without interrupting application service (interruption of some milli-seconds only) to your clients. Inactive partition mobility is also possible for those partitions that are in power off mode. Live partition mobility improves the availability of your server by eliminating planned outages, and balancing workloads during peaks.

- (Dynamic) logical partitioning (DLPAR)

  A logical partition is a set of system resources that contain whole or a portion of a CPU, memory, physical, or virtual I/O resources, all logically grouped into one same partition. The dynamic capability allows adding resources dynamically and on demand.

- Virtual Ethernet

  Virtual Ethernet is managed by the Power Hypervisor and shares the internal Power Hypervisor bandwidth.

- Shared Ethernet Adapter (SEA)

  SEAs are logical adapters that bridge a physical Ethernet card with the virtual Ethernet adapter, so that a logical partition can communicate with the outside world.

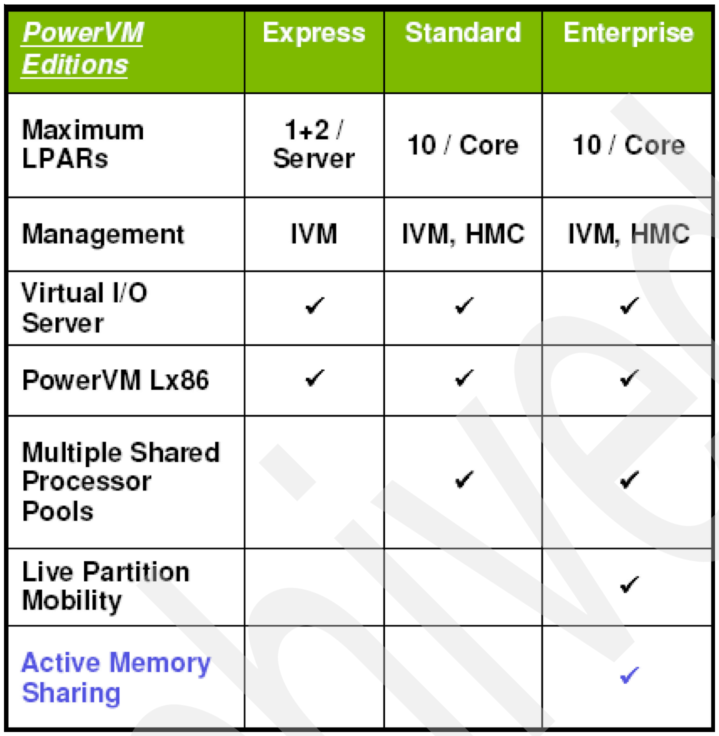
- Shared-processor pools

  Shared processor logical partitions are allocated processor units out of the processor pool. The amount of processor capacity that is allocated to that partition (its entitled capacity) can be as small as a tenth of a physical CPU and as big as the entire processor. The granularity to add processor units is 1/100 of a CPU. The shared processor pool can have up to one whole CPU contained in the physical server.

- N-Port Identifier Virtualization (N-PIV)

  N-PIV adapters are particular 8 GB Fibre Channel adapters that allow a client to see the SAN device, transparently provided by the VIOS. For example the client logical partition sees the hdisk as MPIO DS5100/5300 Disk, instead of as a Virtual SCSI Disk Drive.

PowerVM provides advanced virtualization capabilities to your Power System or Power Blade server. It exists in editions as shown in Figure 1-7.



PowerVM Editions	Express	Standard	Enterprise
Maximum LPARs	1+2 / Server	10 / Core	10 / Core
Management	IVM	IVM, HMC	IVM, HMC
Virtual I/O Server	✓	✓	✓
PowerVM Lx86	✓	✓	✓
Multiple Shared Processor Pools		✓	✓
Live Partition Mobility			✓
Active Memory Sharing			✓

Figure 1-7 PowerVM Editions

- Express Edition

  The Express ASE edition allows you to use only 1 VIOS and a single LPAR, managed by the Integrated Virtualization Manager (IVM).

- Standard Edition

  This edition allows you to create as much as 10 LPARs per core, and supports multiple shared processors pools. See the following Web page for limits:

  http://www-03.ibm.com/support/techdocs/atsmastr.nsf/WebIndex/TD103130

- Enterprise Edition

  This edition covers all capabilities of the express and standard edition, plus the Live Partition Mobility and the Active Memory™ Sharing capabilities.

All Power Systems servers can use standard virtualization functions or logical partitioning (LPAR) technology by using either the Hardware Management Console (HMC) or the Integrated Virtualization Manager (IVM). Figure 1-8 shows the PowerVM virtualization architecture.

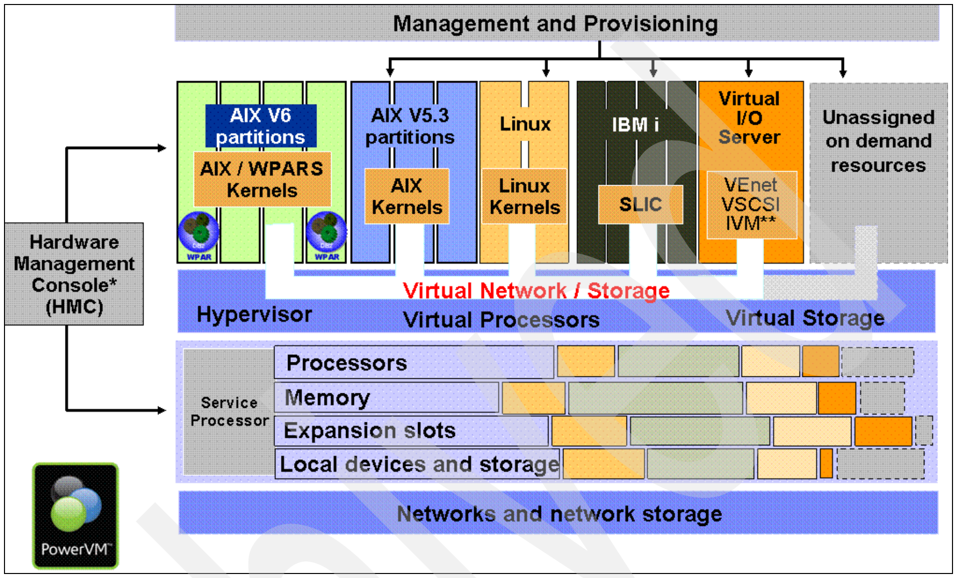


Figure 1-8 PowerVM virtualization architecture

The HMC is a separate server that controls the build of logical partition (LPAR), enables starting and stopping of logical partitions at distance by a remote terminal console, and enables the dynamic functionalities of your Power System to add, move or remove hardware to your logical partitions (DLPAR) dynamically. An HMC can manage multiple Power Systems. The HMC user interface is Web-based and is the focal point for your server environment and the hardware.

Note: The HMC is only used with Power Systems servers, while the Integrated Virtual Manager (IVM) is only used for Power Blades servers.

Similar to the HMC, the IVM can be used to create and maintain logical partitions. It is also integrated with the VIOSs. Nevertheless, the IVM can only control one Power System.

Note: IVM is disabled if an HMC is connected to your Power System.

LPAR: logical partition are built out of a profile. It can be seen as a virtual machine. The entire Power System can be divided into one single or multiple LPARs, each running its own operating system and completely isolated from each other. The profile contains all of the information needed to boot the logical partition, number of processors, memory, and I/O adapters (physical or virtual).

Note: A logical partition's profile can be configured to contain all of a system's resources. This is similar to a standalone system.

Tip: Rather than configuring your system as a full system partition, you may consider configuring a partition containing all the system resources. This gives you more flexibility if you want to add an extra partition to your system.

1.5.1 Logical partition type

Logical partitions are built upon a profile that contains all the physical and virtual devices that you need to start your partition. In Figure 1-9 on page 26, you see the flow to create a profile for a logical partition. Frequently, questions arise regarding whether a dedicated or shared processor is the correct logical partition type for a particular workload and system environment. There are a number of factors to consider, such as performance, overall system use, and consolidation of multiple workloads onto a single server to improve power usage and cooling efficiencies. The chart in Figure 1-9 on page 26 shows a macro view of how you can configure a logical partition's profile.

For more information refer to Chapter 5, "LPAR considerations" on page 225.

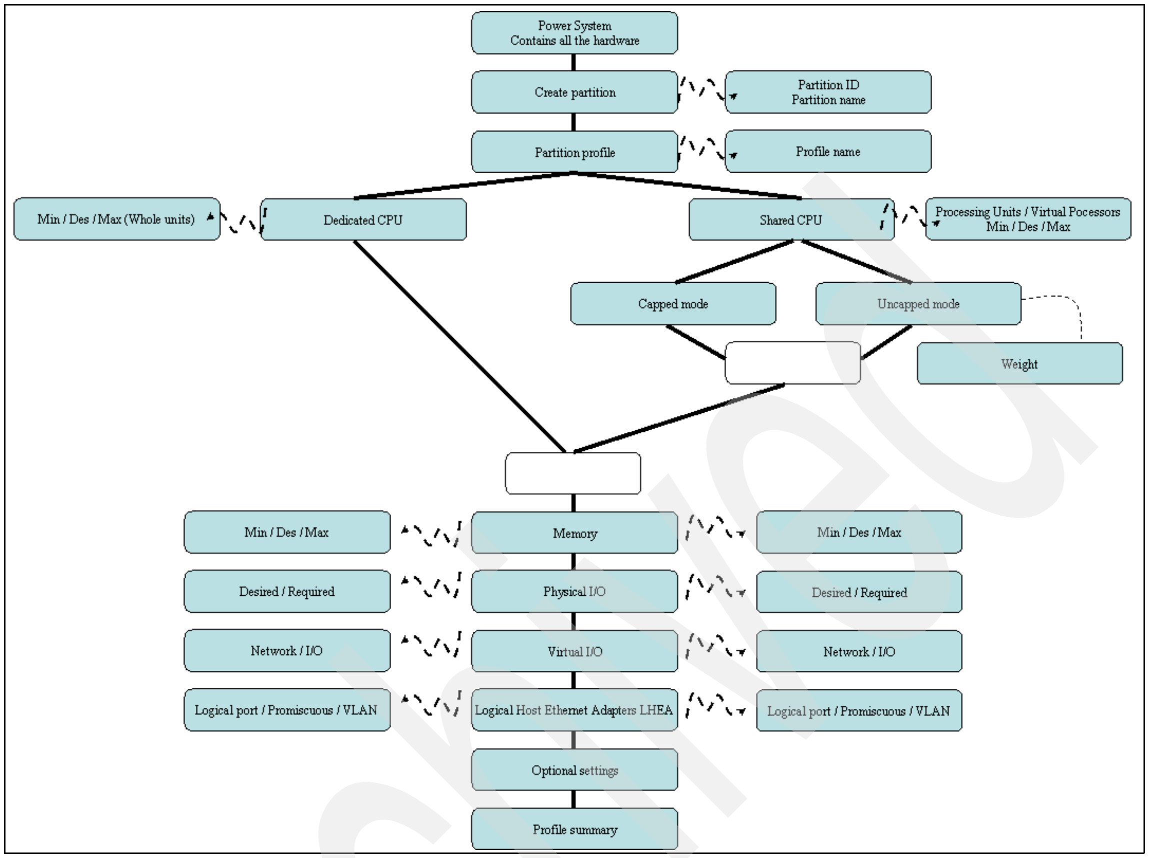


Figure 1-9 Profile chart

1.5.2 Processor virtualization

When dividing a Power System, you can choose between two partitioning methods depending on how you want to share the CPU capacity among the LPARs:

- Shared
- Dedicated

Figure 1-10 on page 27 shows the terminologies used when talking about the processor distribution for your LPARs.

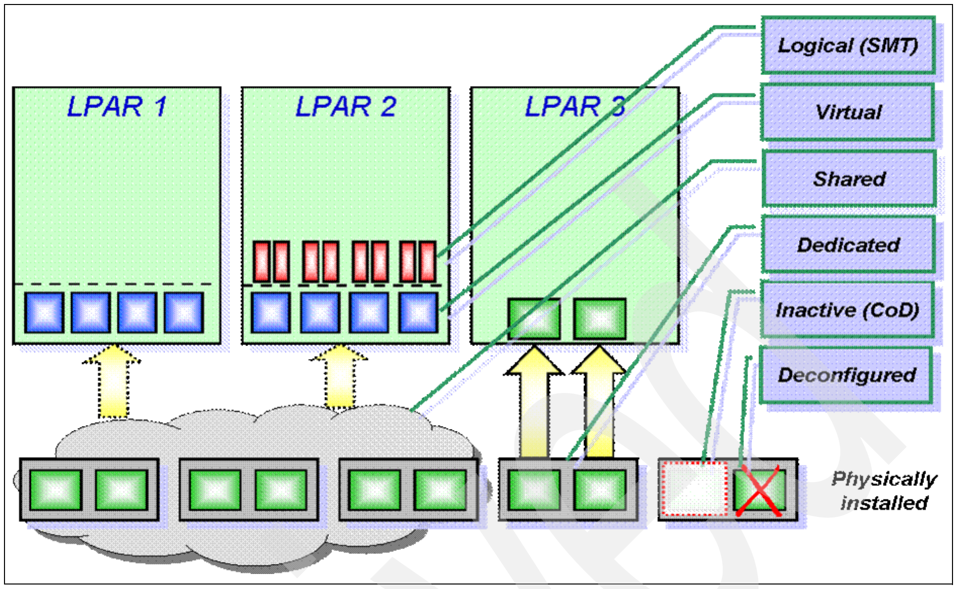


Figure 1-10 Processors terminologies

Let us go over a few of these virtualization concepts:

- Dedicated CPU

  A dedicated CPU is an entire processor that you allocate to your profile. Only that particular LPAR uses that processor.

Attention: Upon activation, if a logical partition does not use 100% of its dedicated processor resources, the unused processor resources are ceded to the shared processor pool. This is called *Donating Dedicated CPU*. When the logical partition needs that donated CPU resource back, the Power Hypervisor immediately liberates it and give it back to the donating logical partition.

- Shared CPU

  With shared CPUs, physical CPUs are placed in a pool of processors and shared between multiple logical partitions. The logical partition is assigned a portion of that CPU pool. It can be as small as a 10th of a real CPU, with a granularity of a 100th. That portion is called a *processing unit*. The ability to divide a CPU into smaller units is known as the *Micro-Partitioning technology*. The granularity of that processor unit is a hundredth of a CPU. An example of a shared processor unit allocation is 1.34 CPU. In this mode, processing units that are not used return to the shared processor pool for other LPAR if they need additional CPU.

Tip: In shared CPU mode, the logical partition is guaranteed to have its processing entitlement whenever it needs it. The Power Hypervisor is managing the processing entitlements and allocates it to the demanding partition.

With the shared CPU, we have two other configurations:

– Capped mode

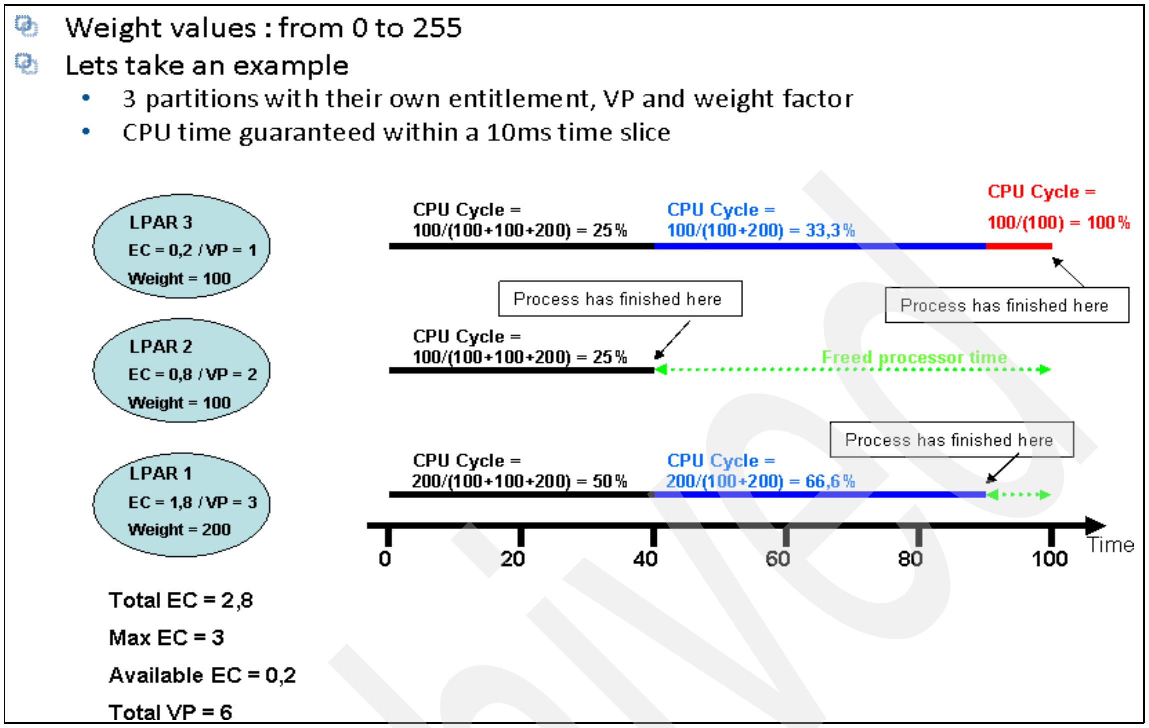
  In capped mode, the processor units allocated to the logical partition cannot exceed the processor unit it was allocated.

– Uncapped mode

  In uncapped mode, the logical partition can get more CPU units than it is allocated, up to the shared CPU pool available units. These additional CPU units are given and treated by the Power Hypervisor. In this mode, a weight factor prioritizes the portion of CPU that is given to a partition, as shown in Figure 1-11 on page 29.

  In Figure 1-11 on page 29, the following abbreviations are used:

  - EC stands for Entitled Capacity and represents the actual processor core capacity available to a partition.
  - VP stands for Virtual Processor and defines the maximum number of physical processor cores that the system can access simultaneously to provide the processing capacity of a shared processors partition.



- Weight values : from 0 to 255
- Lets take an example
  - 3 partitions with their own entitlement, VP and weight factor
  - CPU time guaranteed within a 10ms time slice

LPAR 3
EC = 0,2 / VP = 1
Weight = 100
LPAR 2
EC = 0,8 / VP = 2
Weight = 100
LPAR 1
EC = 1,8 / VP = 3
Weight = 200
Total EC = 2,8
Max EC = 3
Available EC = 0,2
Total VP = 6
CPU Cycle = 100/(100+100+200) = 25%
CPU Cycle = 100/(100+200) = 33,3%
CPU Cycle = 100/(100) = 100%
Process has finished here
Process has finished here
CPU Cycle = 100/(100+100+200) = 25%
Freed processor time
Process has finished here
CPU Cycle = 200/(100+100+200) = 50%
CPU Cycle = 200/(100+200) = 66,6%
0
20
40
60
80
100
Time

Figure 1-11 Weight factor calculation

- Active processors

  Active processors are the usable physical processors in the system.

- Inactive processors

  Inactive processors are those processors that can be activated on demand. They are known as Capacity On Demand (COD) processors and need an activation code to be converted into active processors

- Deactivated processors

  Deactivated processors are processors that have a hardware problem. In case of a hardware problem and if COD processors are available, the broken processor is swapped with a COD processor. If COD processors are not available, ask an IBM Customer Engineer engineer to change the defective processor, which can require down-time.

- Virtual processors

  Virtual processors are abstractions of physical processors that are assigned to logical partitions. The operating system uses the number of virtual processors assigned to the logical partition to calculate the number of operations that the operating system can perform concurrently. Virtual processors are relevant to micro-partition. Micro-partitioning it maps virtual processors to physical processors. The virtual processors are assigned to the partitions instead of the physical processors.

- SMT Simultaneous Multi-Threading

  SMT Simultaneous Multi-Threading is a concept where multiple threads of execution can execute on the same processor at the same time. With SMT enabled, each hardware thread is seen as a logical processor. SMT is not a virtualization concept.

Tip: When SMT is not enabled, a virtual processor appears as a single logical processor.

1.5.3 Memory virtualization

PowerVM has been enhanced with active memory sharing (AMS), an advanced virtualization technology that intelligently flows memory from one logical partition to another to increase use and flexibility of memory. It allows for dynamic sharing of memory. It is designed for logical partitions with variable memory needs. Using AMS improves the overall memory use of your Power System, as shown in Figure 1-12 on page 31. The graph shows the percentage of memory assigned to a partition at a particular point of time for dedicated memory for AMS.

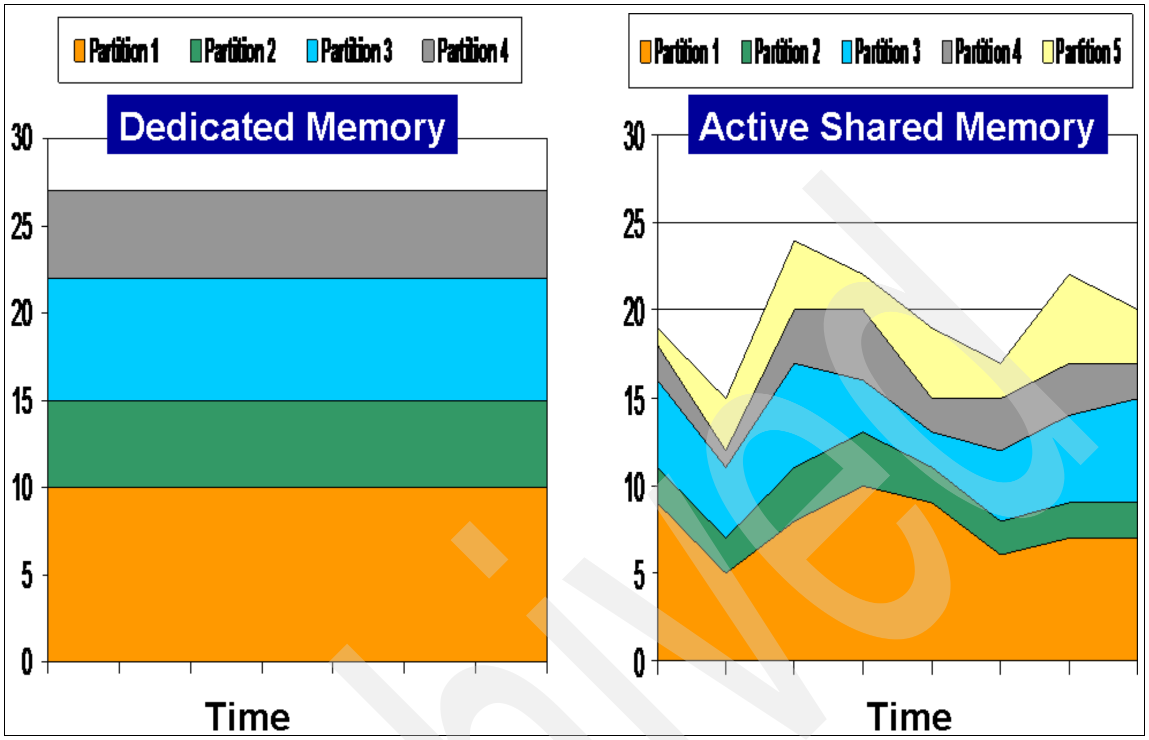


Figure 1-12 Dedicated versus shared memory environment

Note: AMS needs PowerVM Enterprise Edition key installed on your system.

- Dedicated memory

  Dedicated memory is physical memory dedicated to an LPAR. It is reserved for that partition and cannot be shared. A logical partition that uses dedicated memory, known as a *dedicated memory partition*, only uses a fixed amount of memory that it was assigned.

- Shared memory

  Shared memory is that memory assigned to the shared memory pool and shared among multiple logical partitions. A logical partition that uses shared memory, known as a *shared memory partition*, allows an appropriate logical memory size to be defined without requiring a corresponding amount of physical memory to be allocated.

You must specify micro-partitions if you want to specify shared memory for this logical partition.

1.5.4 I/O virtualization

Virtual I/O is often referred to as a set of network and storage virtualization features, such as:

- Virtual Ethernet

  The Virtual Ethernet adapter allows logical partitions to communicate with each other within the same Power System, using the Power Hypervisor's internal switch. They do not need any network hardware adapter or cables to communicate.

- Shared Ethernet Adapter (SEA)

  This logical adapter enables an LPAR to communicate with the outside world. SEA adapters can be configured in a SEA fail-over mechanism to protect your client logical partitions from VIOS network failure. Moreover, SEA is mandatory for Live Partition Mobility use.

- Integrated Virtual Ethernet (IVE)/Host Ethernet Adapter (HEA)

  IVE, or HEA, is a dual or quad port Fibre Channel card, available in 1 Gbps (dual or quad port) or 10 Gbps (quad port only). It enables an easy way to manage the sharing of the integrated high-speed Ethernet adapter ports. The integrated virtual Ethernet is directly connected to the GX+ bus instead of being connected to a PCIe or PCI-X bus. This provides the IVE card with high throughput and low latency. It is available on p520 and p550 servers only. The IVE card is installed by manufacturing and does not support hot-swappable or hot-plugable capabilities.

Figure 1-13 illustrates the difference between the use of SEA and IVE.

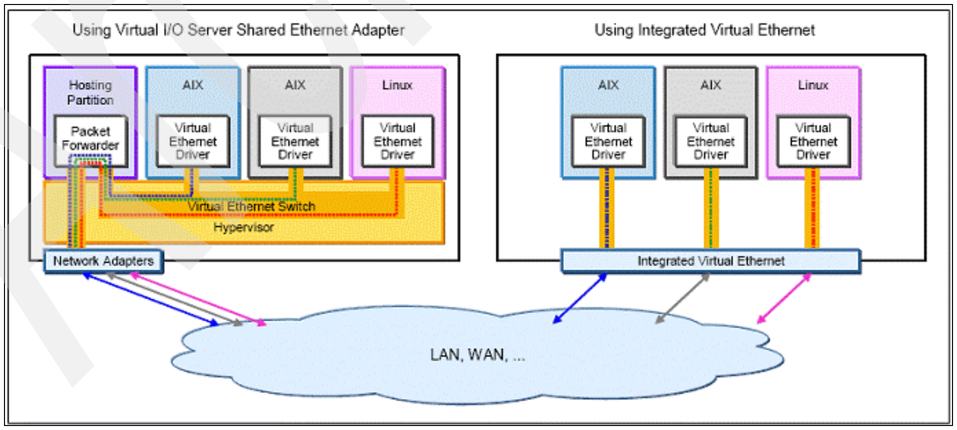


Figure 1-13 SEA versus IVE use

- Virtual storage

  The virtual SCSI adapter allows logical partitions to connect to the SAN storage through the VIOSs. It is recommended to configure your virtual server SCSI adapters to connect to the virtual client SCSI adapter. In this way you ensure that only that particular logical partition can connect to the disks mapped on the VIOS. LUNs from VIOSs can be mapped in entire disks or masked as LUN (a logical volume is first created on the VIOS and then masked to the client).

- N_Port ID Virtualization

  N_PIV adapters simplify the management of Fibre Channel SAN environments. They enable access to SAN devices. NPIV allows multiple N_Port IDs to share one single physical N_Port. This allows multiple Fibre Channel initiators to occupy a single physical port. N-PIV allows the SAN based LUN to be exported to the LPAR rather than the VIOS. This puts the SAN administrators back in charge of which LPAR owns the LUNs, so they can decide which LPAR gets low cost disks or high performance disks. Currently, they give the LUN to the VIOS and the VIOS administrator decides which LPAR gets the disk space. N-PIV adapters are physical 8 GB Fiber Cards and require N-PIV support on your switches.

All virtual network and storage devices are identified with their slot number. Slot number can aid you in your management while creating your naming convention, for situations where lots of logical partitions co-exists and where administration of those virtual resources become more and more complex.

Tip: In a dual VIOS setup, try to get a mapping in slot numbers to ease the management of your servers. For example, you can use a particular slot number range dedicated to your virtual networks and make it big enough to allow for growth (for example 11–30) where virtual SCSI slot number range ranges from 31 to 50.

1.5.5 Useful links

The following list the links to the Information Center documentation for the main topics discussed in this chapter. You can refer to these links for more information about these topics.

AIX 6.1

- http://publib.boulder.ibm.com/infocenter/aix/v6r1/index.jsp
- http://publib16.boulder.ibm.com/pseries/index.htm
- http://www-03.ibm.com/support/techdocs/atsmastr.nsf/WebIndex/TD103130

PowerVM

- http://www-03.ibm.com/systems/power/software/virtualization/index.html
- http://www14.software.ibm.com/webapp/set2/sas/f/vios/documentation/installreadme.html

Power Systems

- http://publib.boulder.ibm.com/infocenter/powersys/v3r1m5/index.jsp
- http://www-03.ibm.com/systems/power/hardware/
- http://www14.software.ibm.com/webapp/set2/sas/f/power5cm/power6.html

DB2

- http://publib.boulder.ibm.com/infocenter/db2luw/v9r7/index
- http://publib.boulder.ibm.com/infocenter/db2luw/v9r7/topic/com.ibm.db2.luw.wn.doc/doc/c0054107.html

2

AIX configuration

This chapter discusses the recommended AIX tunable for the best performance of DB2. By default, AIX V6.1 provides the optimized tunable values for the best performance of the system and application. DB2 recommends certain configurable parameter changes as the best practise taking the performance and security into considerations. The DB2 registry variables on AIX is also discussed. Finally, a table containing the configurational parameters differences between AIX 5.3 and AIX 6.1 release is provided for quick reference.

Topics discussed in this chapter are as follows:

2.1 AIX configuration and tuning for DB2

In this section we discuss AIX tunable parameters germane to DB2.

2.1.1 Virtual memory considerations

AIX Virtual Memory Manager (VMM) classifies memory into computational and non-computational memory.

- Computational memory includes application memory (for example, DB2 process text, data, and stack segments) and kernel memory.
- Non-computational memory is classified as file system cache, which includes file system data from JFS (journaled file system), JFS2, NFS (Network File System), GPFS™ (General Parallel File System) or any file system type. This is also considered as permanent memory.
  - All permanent memory except JFS is called client memory.
  - All file system cache is tracked by the parameter, numperm.
  - Size of the client memory is tracked by the VMM parameter, numclient.

Figure 2-1 shows the computational and non computational memory occupancy in real memory.

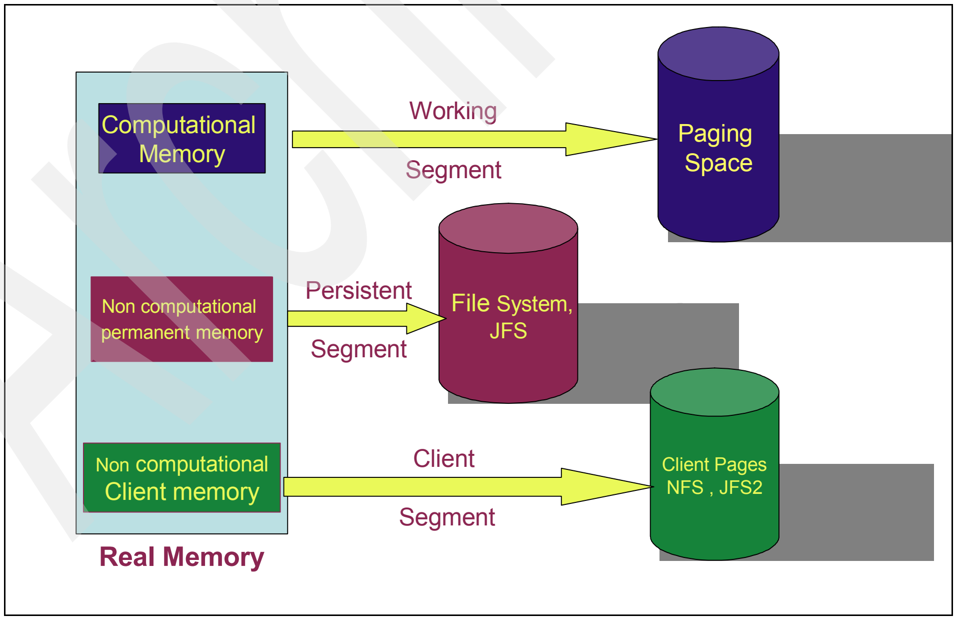


Figure 2-1 Computational and non-computational memory occupancy

Computational memory is represented as working storage (segment) in the svmon output. Non computational memory is considered to be either client or persistent segment.This been represented as “clnt” in the svmon output as shown in Example 2-7 on page 45. This is also called file pages.

Figure 2-2 shows the file system cache consumption in real memory.

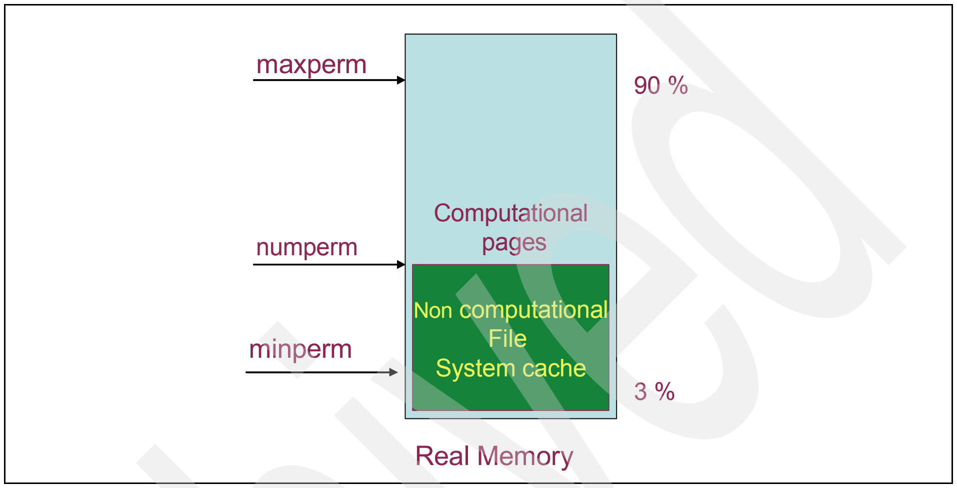


Figure 2-2 The file system cache consumption as pointed out by numperm

minperm and maxperm values show the range that the file system cache can grow. If computational pages need more memory, then the file cache pages are stolen (page stealing) based on the repaging rate.

When there is no record of the page having been referenced recently, it is considered as a new page fault. But when a page that is known to have been referenced recently is referenced again and not found in memory, it is considered as repage fault. Keeping in control of the repage faults reduces I/O demand and potentially improve the system performance.

lru_file_repage specifies whether the repaging rate is considered in determining whether to steal a file or computational pages, when the file pages consumption goes beyond minperm.

VMM maintains a logical free list of memory pages to satisfy a page fault. It uses a page replacement algorithm to discover which memory pages (computational or non computational) currently in memory are to be moved to free list.

- minfree specifies the number of memory frames in the free list at which VMM starts to steal pages in memory to replenish the free list.
- maxfree specifies the number of memory pages in the free list at which VMM stops page stealing.
- Page replacement is done through the kernel process "lrud".

Figure 2-3 shows the page stealing based on minfree and maxfree values.

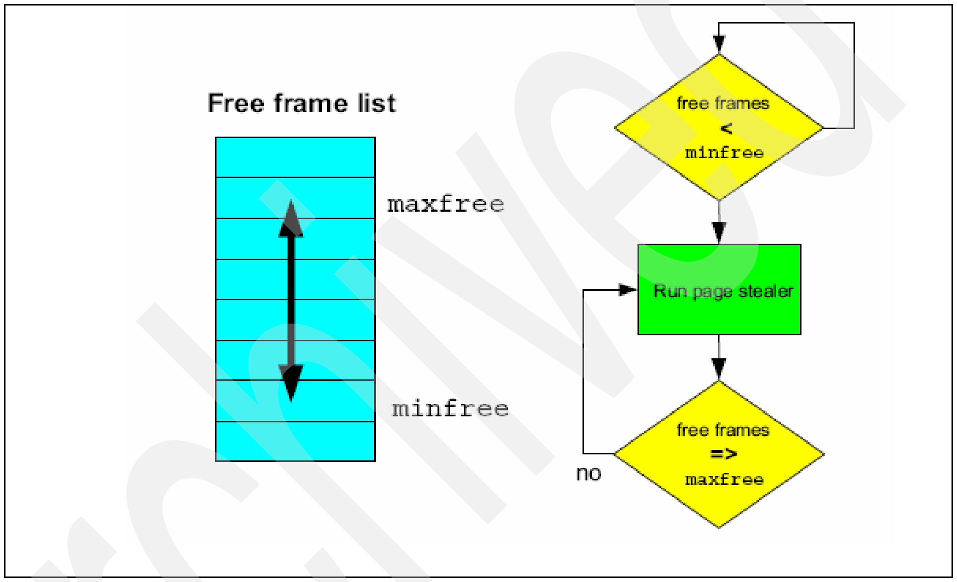


Figure 2-3 Page stealing based on minfree and maxfree values

2.1.2 AIX vmo command

This command sets or displays the current or next boot values for all VMM tuning parameters. This command can also make changes transient (lost at the next boot) or permanent (applied at the next boot).

For example, the **vmo -L** command displays the current, default, and the boot vmo parameter settings, as shown in Figure 2-4 on page 39.

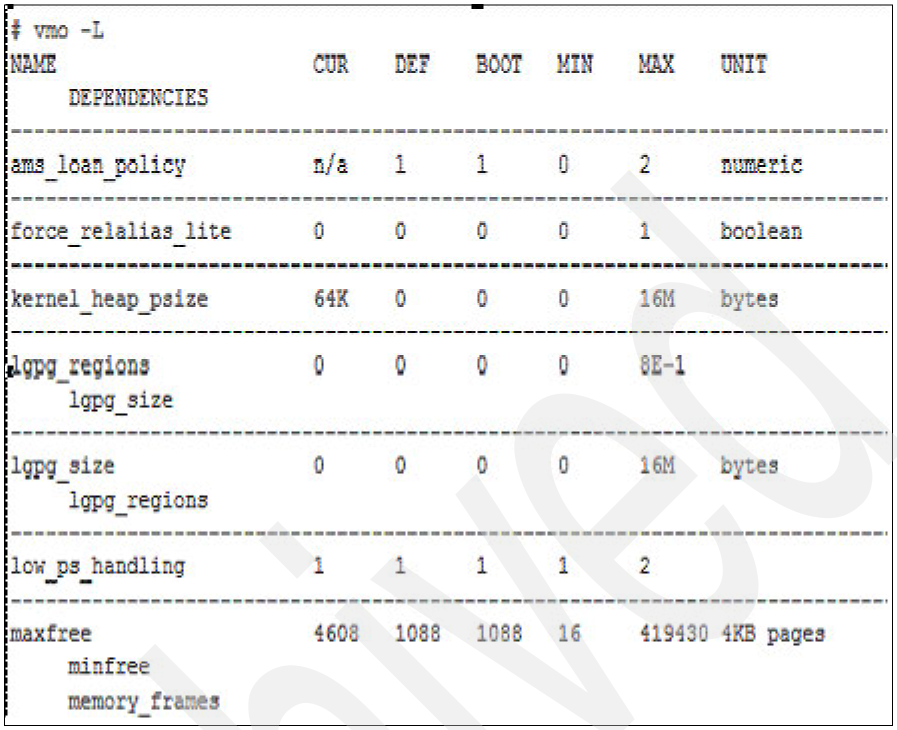

```
# vmo -L
NAME                      CUR    DEF    BOOT   MIN    MAX    UNIT
     DEPENDENCIES
--------------------------------------------------------------------------------
ams_loan_policy           n/a    1      1      0      2      numeric
--------------------------------------------------------------------------------
force_relalias_lite       0      0      0      0      1      boolean
--------------------------------------------------------------------------------
kernel_heap_psize         64K    0      0      0      16M    bytes
--------------------------------------------------------------------------------
lgpg_regions              0      0      0      0      8E-1
     lgpg_size
--------------------------------------------------------------------------------
lgpg_size                 0      0      0      0      16M    bytes
     lgpg_regions
--------------------------------------------------------------------------------
low_ps_handling           1      1      1      1      2
--------------------------------------------------------------------------------
maxfree                   4608   1088   1088   16     419430 4KB pages
     minfree
     memory_frames
```

Figure 2-4 Output of vmo -L

Example 2-1 shows the current vmo settings in the system.

Example 2-1 Output of "vmo -a"

```
# vmo -a
        cpu_scale_memp = 8
 data_stagger_interval = 161
                 defps = 1
   force_relalias_lite = 0
             framesets = 2
             htabscale = n/a
     kernel_heap_psize = 4096
          kernel_psize = 4096
  large_page_heap_size = 0
          lgpg_regions = 0
             lgpg_size = 0
       low_ps_handling = 1
       lru_file_repage = 0
     lru_poll_interval = 10
             lrubucket = 131072
            maxclient% = 90
               maxfree = 5418
               maxperm = 219580
              maxperm% = 90
                maxpin = 212018
               maxpin% = 80
  ...........
```

Example 2-2 shows how to modify and make the changes persistent across the reboot. The **`vmo -o`** command is used to make the changes, and the option **`-p`** is used to make the changes persistent dynamically and across reboot.

Example 2-2 `vmo` to make the changes permanent across reboot

```
# vmo -p -o minfree=4906 -o maxfree=5418
Setting maxfree to 5418 in nextboot file
Setting minfree to 4906 in nextboot file
Setting maxfree to 5418
Setting minfree to 4906
```

Note: Beginning with AIX Version 6.1, tunables are classified as restricted use tunables. They exist and must be modified primarily for specialized intervention by the IBM development support or development teams.

As these parameters are not recommended for user modification, they are no longer displayed by default but only with the -F option (force) on the command. For example, the command **`vmo -F -a`** lists all the tunables including restricted.

2.1.3 VMM considerations for DB2

Protecting computational memory is important for DB2, as it maintains its own data cache. Following AIX tunables discusses about the default values and the DB2 recommended values.

minfree

When the size of the free list falls below this number, the VMM begins stealing pages. It continues stealing pages until the size of the free list reaches maxfree.

If the value free frame waits from vmstat -s value increases over the period of time, then it is advisable to increase the minfree value.

- Default value in AIX6.1: 960
- DB2 recommended value:
  - 4096 for memory less than8 GB
  - 8192 for memory greater than 8 GB
- How to set: vmo -p -o minfree=4096
- Type: Dynamic (No reboot required)

maxfree

The difference between minfree and maxfree must not go beyond 1024, as it increases the time in replenishing the free list. The difference must be equal to or greater than j2_maxPageReadAhead common maxpgahead IO tunable value. See 2.1.7, “Input and output tunable considerations” on page 63 for details about this tunable. See Example 2-3.

- Default Value in AIX6.1: 1088
- DB2 Recommended value:
  - minfree + 512
  - if minfree is 4096, then maxfree=4608
- How to set: vmo -p -o maxfree=4608
- Type: Dynamic

Example 2-3 Setting the maxfree value

```
# vmo -p -o maxfree=4608 -o minfree=4096
Setting maxfree to 4608 in nextboot file
Setting minfree to 4096 in nextboot file
Setting maxfree to 4608
Setting minfree to 4096
```

maxclient%

This specifies the maximum physical memory usage for client pages caching. If the percentage of real memory occupied by file pages is higher than this level, the page-replacement algorithm steals only client pages. It is recommended to set strict_maxclient to 1 along with this setting for DB2. This considers maxclient as the hard limit and strictly steals client pages as soon as the limit is reached.

- Default value in AIX6.1: 90%
- DB2 recommended value: 90%

Note: This is a restricted tunable in AIX6.1. Hence no further change is recommended.

maxperm%

This specifies the maximum physical memory usage for file pages caching.

If the percentage of real memory occupied by file pages rises above this level, the page-replacement algorithm steals only file pages.

- Default value in AIX6.1: 90%
- DB2 recommended value: 90%

Note: This is a restricted tunable in AIX6.1. Hence no further change is recommended.

minperm%

If the percentage of real memory occupied by file pages falls below this level, the page replacement unit steals only computational values. Hence a low value is recommended for DB2. AIX by default sets this value to 3% compared to AIX5L where by default it is 20%.

- Default value in AIX6.1: 3%
- DB2 recommended value: 3%

Note: This is not a restricted tunable in AIX6.1.

strict_maxclient

If the client page caching increases beyond the maxclient% value, setting this parameter acts as a hard limit and triggers file pages to be stolen.

- Default value in AIX6.1: 1
- DB2 recommended value: default value

Note: This is a restricted tunable in AIX6.1. Hence no further change is recommended.

lru_file_repage

When VMM needs memory if the free memory pages goes below minfree, or when maxclient% is reached with strict_maxclient set, the lru makes a decision whether to steal the computational pages or the file pages.

This determination is based on the number of parameters, but the key parameter is lru_file_repage.

Setting this tunable value to 0 indicates that only the file pages are stolen if the file pages are greater than the minperm value. This prevents computational pages from being paged out. This is a restricted tunable in AIX6.1. Hence no further change is recommended.

- Default value in AIX6.1: 0
- DB2 recommended value: default value

Notes:

- AIX Version 6 allows the system to use up to 90% of its real memory for file caching, but it favors computational pages as resident pages over file pages.By setting lru_file_repage to 0, it forces the page replacement algorithm to steal computational pages only when the percentage of cached file pages is less than the minperm% value. Hence it might not be required to reduce the maxclient, maxperm value to avoid paging as practiced with AIX5.3.
- In the situation where reducing the maxperm and maxclient values to lower helps to improve the performance, it is recommended to inform or contact IBM Technical Support for a complete solution. IBM does not support any changes to restricted tunables where the change is the solution.
- Tunables such as lru_poll_interval and strict_maxperm are restricted tunables and DB2 recommends the AIX6.1 default value. No change is suggested.
- The tunable page_steal_method is restricted tunable. No change is suggested. Its default value is 1, which creates a linked list of dirty pages to steal the pages rather than scanning memory.

Steps to monitor the memory

The following describes the steps to monitor the memory:

1. The **svmon** command can be used to monitor the computational memory (such as the stack heap of a process) and the persistent memory (such as the file system cache). Computational memory is shown as work, which it considers as working storage, the persistent memory is shown as, pers, and the paging space as pgsp. Example 2-4 shows the **svmon -G** output.

Example 2-4 svmon -G

```
# svmon -G
               size       inuse        free         pin     virtual
memory       524288      422096      171824      206465      330676
pg space     131072        3014
               work        pers        clnt       other
pin          118951           0           0       87514
in use       330676           0       91420
PageSize   PoolSize       inuse        pgsp         pin     virtual
s    4 KB         -      276528        3014      103585      185108
m   64 KB         -        9098           0        6430        9098
```

In Example 2-4, most of the memory is consumed by computational memory, 330676 4 K pages. Persistent memory is nil and client memory has consumed 91420 4 K pages. Also, only a few 4 KB pages have consumed the paging space.

As the maxperm and minperm values are restricted and not recommended to change in AIX6.1, it is suggested to monitor the memory usage as described in the steps that follow:

1. Use **lsps -a** to understand the paging space consumption over the period of time. See Example 2-5.

Example 2-5 lsps -a

```
# lsps -a
Page Space      Physical Volume   Volume Group Size %Used Active  Auto  Type Chksum
          hdisk0           rootvg         512MB     3   yes   yes    lv     0
```

2. Monitor the paging activity using **vmstat**, as shown in Example 2-6.

Example 2-6 vmstat-Monitor pi/po for paging activity

```
# vmstat 1

System configuration: lcpu=6 mem=2048MB ent=0.50

kthr    memory              page              faults              cpu
```

```
----- ----------- ------------------------ ------------ -----------------------
 r  b   avm   fre  re  pi  po  fr   sr  cy  in   sy  cs us sy id wa    pc    ec
 1  0 330629 168041   0   0   0   0    0   0   8  232 190  0  1 99  0  0.01   1.8
 1  0 330629 168041   0   0   0   0    0   0   8   93 176  0  1 99  0  0.01   1.3
 1  0 330793 167877   0   0   0   0    0   0  10 6769 177  7  7 86  0  0.07  14.5
```

Note: Use **svmon -Pgt 5** to understand the top 5 paging process

3. Use the **vmstat** command to understand the file cache usage by the system, as shown in Example 2-7.

Example 2-7 vmstat -v | grep num

```
vmstat -v | grep num
17.6 numperm percentage
17.6 numclient percentage
```

4. Run **svmon -Put 5** to understand which process is consuming more memory, which lists the top five processes. See Example 2-8.

Example 2-8 svmon -Put 1

```
Pid Command          Inuse      Pin     Pgsp  Virtual 64-bit Mthrd  16MB
266418 java             42548     7663        0    37138      N     Y     N

     PageSize                Inuse        Pin       Pgsp    Virtual
     s    4 KB               21748         79          0      16338
     m   64 KB                1300        474          0       1300

    Vsid      Esid Type Description              PSize  Inuse   Pin Pgsp Virtual
    3449         3 work working storage            sm  15677     0    0   15677
   3205d         d work fork tree                   m    792     0    0     792
                   children=4d6cdc, 0
    4002         0 work fork tree                   m    508   474    0     508
                   children=802760, 0
   23199         - clnt /dev/hd9var:12359           s    694     0    -       -
    ff6f         - clnt /dev/hd9var:16629           s    676     0    -       -
   173a3         - work                             s    381    75    0     381
   3f437         f work working storage            sm    247     0    0     247
   2b39d         - clnt /dev/hd9var:12300           s    243     0    -       -
   33591         - clnt /dev/hd9var:20583           s    230     0    -       -
   1d406         - clnt /dev/hd9var:16392           s    212     0    -       -
    d0ee         - clnt /dev/hd9var:20700           s    208     0    -       -
   192c4         - clnt /dev/hd9var:12363           s    178     0    -       -
   37573         - clnt /dev/hd9var:28845           s    164     0    -       -
   17863         - clnt /dev/hd9var:28853           s    134     0    -       -
   191a4         - clnt /dev/hd9var:12373           s    104     0    -       -
   2b11d         - clnt /dev/hd9var:12379           s     94     0    -       -
   195e4         - clnt /dev/hd9var:20991           s     83     0    -       -
   3d7f6         - clnt /dev/hd9var:28797           s     61     0    -       -
   237d9         - clnt /dev/hd9var:28860          s     61     0    -       -
   1d666         - clnt /dev/hd9var:32821          s     60     0    -       -
   21398         - clnt /dev/hd9var:12370          s     57     0    -       -
   3b5d5         - clnt /dev/hd9var:16585          s     53     0    -       -
   376d3         - clnt /dev/hd9var:28838          s     51     0    -       -
   39574         - clnt /dev/hd9var:24714          s     50     0    -       -
    75ab         - clnt /dev/hd9var:16580          s     50     0    -       -
   3f737         - clnt /dev/hd9var:28738          s     50     0    -       -
   255da         - clnt /dev/hd9var:20987          s     49     0    -       -
   1f667         - clnt /dev/hd9var:32822          s     44     0    -       -
   3b0f5         - clnt /dev/hd9var:12382          s     40     0    -       -
```

Note:

- If DB2 is doing more file cache, determine if there are any table spaces set up with file system cache, and decide whether file system caching can be disabled.
- If there is any non-DB2 file systems or process causing file system cache to increase and causing paging leading to low performance, rectify it.
- To determine which table spaces have file system cache enabled, use the **`db2pd -db <database name> -tablespaces`** command and see the FSC column. The possible values are ON or OFF.

Monitoring helsp to understand the memory activity in the system. Because AIX does not recommend changing the restricted values such as minperm, maxperm, contact IBM technical support for further assistance, if required. In summary, the best practise for DB2 regarding AIX VMM setting is as follows:

- minfree=4096 for physical memory <8 GB and 8192 for memory > 8 GB
- maxfree=minfree+512
- maxperm%=90 (default)
- maxclient%=90(default)
- strict_maxclient=1(default)
- minperm%=3(default)
- lru_file_repage=0(default)

Except for minfree and maxfree, no other modification is required. It does not require reboot of the system after modification.

2.1.4 Large page considerations

Using larger page sizes increases memory access performance because fewer address translations in the hardware have to be done, and caching mechanisms can be used more efficiently. However, memory regions might be wasted if a larger page size is allocated and populated with data less than the page size.

Starting with AIX6.1 on a POWER6-based processors the VMM can dynamically promote pages to a larger page size. This page promotion (4 k to 64 k) is completely transparent to the application and is done without the need of user intervention. To see how many large pages are in use on your system, use the **`vmstat -l`** command.

> **Note:** VMM only supports dynamically varying the page size of working storage memory. Working storage memory comprises process stack, data and shared library text segment in AIX address space.

To make use of the 16 MB page size, you have to specify the amount of physical memory that you want to allocate to back large pages. The default is not to have any memory allocated to the large page physical memory pool. It can be specified using the **`vmo`** command. The following example allocates 1 GB to the large page physical memory pool (a reboot is required after running the command to make the changes effective):

```
vmo -r -o lgpg_regions=64 -o lgpg_size=16777216
```

Pros and cons of enabling a large page

The following list details the pros and cons of enabling a large page:

- AIX treats large pages as pinned memory. Therefore, no paging support is available.
- The data of an application that is backed by large pages remains in physical memory until the application completes.
- Enabling 16 MB pages blocks the memory for its usage. This deprives the application of using that pool of memory, which affects the system if it does not have sufficient physical memory for other application requirement.
- Memory allocated to large page is not available for 4 k. Therefore, allocating too much physical memory to a large page degrades the system.
- The size of the large page physical memory pool is fixed at boot time and remains the same. A reboot is required to change the size of the large page memory pool.
- Applications which do a large number of fork()s (such as shell scripts) are especially prone to performance degradation when large pages are used.

> **Note:** Huge memory pages cannot be used if Live Partition Mobility is considered (unless you are planning to use inactive mobility).

Large page enablement for DB2

This is an optional setting for DB2. If your system has sufficient physical memory such that allocating huge memory for large page does not affect 4 k page allocation, then enabling 16 MB page can be considered.

To enable the 16 MB page for DB2, follow these steps:

1. Issue the **vmo** command with the following flags:

   ```
   vmo -r -o lgpg_size=LargePageSize -o lgpg_regions=LargePages
   ```

   In the example, `LargePageSize` specifies the size in bytes and `LargePages` specifies the number of large pages to reserve.

   For example, if you need to allocate 25 GB for large page support, run the command as follows:

   ```
   vmo -r -o lgpg_size=16777216 -o lgpg_regions=1600
   ```

2. Run the **bosboot** command so that the **vmo** command takes effect following the next system boot.

   ```
   bosboot  -ad  /dev/ipldevice
   shutdown -Fr now
   ```

3. After the server comes up, enable it for pinned memory. Issue the **vmo** command with the following flags:

   ```
   vmo -o v_pinshm=1
   ```

4. As root user, add the following capabilities for the DB2 instance user to prevent unauthorized users from using large pages for their applications.

   ```
   chuser  capabilities=CAP_BYPASS_RAC_VMM,CAP_PROPAGATE  $USER
   ```

5. Use the **db2set** command to set the DB2_LARGE_PAGE_MEM registry variable to DB, then start the DB2 database manager. For example:

   ```
   db2set DB2_LARGE_PAGE_MEM=DB
   db2start
   ```

6. Validate large page support with the following command:

   ```
   vmstat -l 1 or vmstat -P ALL
   ```

Note: The **`tprof -a -y <process> -O all`** command can also be used for large page analysis.

2.1.5 Paging space considerations for DB2

We recommend that DB2 be tuned in a way that does not use paging space regularly. Periodic paging space usage monitoring is recommended. The following recommendations are for the paging space considerations:

- The general recommendation is to have the paging space be twice the size of the physical memory.

  Example: For a RAM size of 128 GB it is recommended to have 256 GB paging space.
- Use a minimum paging space at default rootvg under the size of the RAM not less than 512 MB and then have the additional (multiple) paging space in alternate disks not exceeding 64GB.
- Use multiple paging spaces, each allocated from a separate physical volume. More than one paging space on a disk is not recommended.
- Create the secondary paging space on physical volumes that are more lightly loaded than the physical volume in rootvg. If minimum inter-disk policy is chosen, moving the paging space to another disk in the same volume group is fine.
- The secondary paging spaces must all be of the same size to ensure that the algorithm performed in turn can work effectively.
- It is better to have the paging space on fast local disks. If it is configured from SAN, make sure to avoid the SAN failure points. Some of the considerations from SAN are detailed in the following list:
  - It is recommended to use a separate LUN for paging space.
  - If LUN is configured through Virtual I/O Server (VIOS), consider providing a dual VIOS with multipath (MPIO) implementation.
  - It is recommended to mirror the disks at the AIX level using the **`mirrorvg`** command.
  - It is recommended to use stripped and mirrored disks.
  - It is recommended to use either RAID5 or RAID10 configurations at SAN.
  - Ensure FC adapter redundancy to avoid any failure.

Paging activity probable causes

The following issues are probable causes for paging activity:

- Free list reaches 0

  Monitor the status of free list that AIX VMM maintains through the **vmstat** command (Example 2-7 on page 45). If the free list represented as fre in the **vmstat** command reaches 0 due to an outburst of memory requirement, the system might page regardless of memory usage.

  In such situations it is recommended to increase the minfree value. Always ensure that the difference between minfree and maxfree does not exceed 1024.

- If active virtual memory exceeds the real memory, then paging happens. Monitor the growth of active virtual memory from the **vmstat** output. See Example 2-9.

Example 2-9 vmstat output showing the memory statistics

```
memory                         page
--------------------- -----------------------------------------
       avm        fre    fi    fo    pi    po       fr       sr
  10560014     133369     1   114     0     0        0        0
  10552156     142713     7  2106     0     0       30       38
  10557399     137113     0    89     0     0        0        0
  10542898     152814     7  2118     0     0       27       30
  10538954     167571     0    97     0     0      188      202
```

Multiply the avm value by 4 K to convert from pages to bytes. From Example 2-9, the avm value of 10538954 corresponds to 42155 MB (10538954*4 K).

In such situation it is recommended to have more physical memory added to the system.

Paging space related AIX commands

The following list details paging space-related AIX commands:

- Monitor the growth of paging space using **lsps -a**, as shown in Example 2-4.

*Example 2-10 **lsps -a** command*

```
# lsps -a

Page Space
  Physical Volume   Volume Group   Size        %Used   Active  Auto   Type   Chksum
  hdisk0      rootvg            1152MB 21      yes  yes  lv   0
```

- To create additional paging space, use the **mkps** command. For example, to create a paging space in volume group myvg that has four logical partitions and is activated immediately and at all subsequent system restarts, use:

```
mkps  -a  -n  -s4 myvg
```

- To change the attribute of the paging space use **chps** command. For example to change the size of the myvg paging space use:

```
chps  -s 4 myvg
```

This adds four logical partitions to the myvg paging space.

Important: It is not recommended to move the default paging space under /. This might affect the boot sequence of the system.

2.1.6 Network tunable considerations

AIX by default provides the best recommended configurable parameters for a network. Similar to VMM tunables, network parameters are also classified as restricted and non-restricted tunables. These non-restricted parameters can be modified as per the requirement but it is not recommended to modify restricted tunables. For modifying any restricted tunables, consult IBM technical support.

From the DB2 perspective, tunables are modified for effective bandwidth use and are modified considering the security aspect.

These tunables can be modified using the command **no**.

AIX no command

To display the user modifiable network tunables, use the **no -a** command, as shown in Example 2-11.

Example 2-11 Output of **no -a** *command*

```
# no -a
              arpqsize = 12
            arpt_killc = 20
           arptab_bsiz = 7
             arptab_nb = 149
             bcastping = 0
```

The **no -F -a** command can be used to display both the non-restricted and the restricted tunables.

To modify the tunable and to make the modification permanent use the **`no -p -o <tunable>=<value>`** command, as shown in Example 2-12.

Example 2-12 Modify and make changes permanent across reboot

```
# no -p -o tcp_recvspace=65536
Setting tcp_recvspace to 65536
Setting tcp_recvspace to 65536 in nextboot file
Change to tunable tcp_recvspace, will only be effective for future connections
```

To modify the command to be effective from the next reboot, perform the following command, as shown in Example 2-13.

Example 2-13 Modification takes effect at the next reboot

```
# no -r -o ipqmaxlen=250
Setting ipqmaxlen to 250 in nextboot file
Warning: changes will take effect only at next reboot
```

To achieve a high level of security and performance improvement, the following network tunable modifications are suggested for DB2 environment. The effect of the modification varies with tunables. A few are dynamic, a few after reboot and a few when the first connection occurs.

- Dynamic

  Changes takes place immediately
- Reboot

  Changes effected in the next system root
- Connect

  Changes in effect when the first connection occurs.

clean_partial_connection

This parameter is recommended to be enabled to avoid SYNC attack (unacknowledged SYN/ACK packets from server by client). As SYNC attack leads to increases in the listen queue backlog (denial of service), removing partial connections makes room for new non-attack connections.

- Default value: 0
- DB2 recommended value: 1
- How to set: no -p -o clean_partial_conns= 1
- Type: Dynamic

This prevents the socket listen queue to be filled up with incomplete 3-way TCP/IP handshake partial connections

ip6srcrouteforward

This option specifies whether or not the system forwards source-routed IPv6 packets. Source routing is a technique where the sender of a packet can specify the route that a packet must take through the network.

- Default value: 1
- DB2 recommended value: 0
- How to set: no -p -o ip6srcrouteforward=0
- Type: Dynamic

A value of 0 causes all source-routed packets that are not at their destinations to be discarded, preventing attacks through source route.

ipignoreredirects

This parameter is used to control ICMP Redirects. The ICMP Redirect message is used to notify a remote host to send data packets on an alternative route. Setting it to 1 ensures that malicious ICMP request cannot be used to create manipulated routes.

- Default value: 0
- DB2 recommended value: 1
- How to set: no -p -o ipignoreredirects=1
- Type: Dynamic

Setting this parameter to 1 ensures that malicious ICMP request cannot be used to create manipulated routes.

ipqmaxlen

This specifies the number of received packets that can be queued on the IP protocol input queue. Examine ipintrq overflows using **`netstat -s`** (See Example 2-14), and consider increasing this value as recommended.

Example 2-14 Verification of overflows

```
# netstat -s | grep ipintrq
        0 ipintrq overflows
```

- Default value: 100
- DB2 recommended value: 250
- How to set: no -r -o ipqmaxlen=250
- Type: Reboot

Preventing overflows helps in processing the packets at the IP level.

An increase in value might result in more time spent in the off-level interrupt handler, because IP has more packets to process on its input queue. This can adversely affect processes which requires CPU time.It is best to increase ipqmaxlen by moderate increments. See Example 2-15 on how to increase ipqmaxlen value.

Example 2-15 ipqmaxlen

```
# no -r -o ipqmaxlen=250
Setting ipqmaxlen to 250 in nextboot file
Warning: changes will take effect only at next reboot
```

ipsendredirects

This is to specify whether the kernel must send redirect signals. Disable this parameter to prevent illegal access through source routing.

- Default value: 1
- DB2 recommended value: 0
- How to set: no -p -o ipsendredirects=0
- Type: Dynamic

Disabling this prevents redirected packets from reaching a remote network.

ipsrcrouterecv

This parameter specifies whether the system accepts source routed packets. Source routing is a technique where the sender of a packet can specify the route that a packet must take through the network. The default value of 0 causes all source-routed packets destined for this system to be discarded. A value of 1 allows source routed packets to be received.

- Default value: 0
- DB2 recommended value: 1
- How to set: no -p -o ipsrcrouterecv=1
- Type: Dynamic

This parameter ensures that the pickets destined to the destination is received.

rfc1323

The enhancement suggested through RFC 1323 is used. RFC1323 suggests maintaining high performance and reliability over high speed paths (bps) through improved TCP window scale option. The TCP window scale option as defined in RFC1323 increased the window scale size from 64 KB to 1 GB. DB2 recommends enabling this parameter for more efficient use of high bandwidth networks.

- Default value: 0
- DB2 recommended value: 1
- How to set:
  - **`ifconfig en0 rfc1323 1`** (Interface specific, dynamic, no reboot required)
  - **`no -p -o rfc1323=1`** (system specific)
- Type: Connect

The recommended setting enables increased performance for high speed networks. It is advisable to enable this to have higher send and receive buffer values (greater than 64 KB)

> **Note:** The rfc1323 network option can also be set on a per interface basis through the **`ifconfig`** command. It can be verified using **`ifconfig -a`**.

tcp_nagle_limit

This tunable helps avoid sending large number of small packets (packet consolidation). By default, it is set to 65535, the maximum size of IP packet in AIX. When using DB2, we recommend that you disable it. This ensures that AIX does not try to consolidate packets.

- Default value: 65535
- DB2 recommended value:1
- How to set: no -p -o tcp_nagle_limit=1
- Type: Dynamic

Consolidating the packets affects the real time data transfer.

TCP header size of 40bytes is sent even to send 1 byte of data across the wire, which is a huge overhead.

tcp_nodelayack

Enabling this parameter causes TCP to send immediate acknowledgement (Ack) packets to the sender.We recommend to set it to ON.

- Default value: 0
- DB2 recommended value: 1
- How to set: no -p -o tcp_nodelayack=1
- Type: Dynamic

Delay in sending the ACK affects the real time data transfer.

Enabling this option causes slightly more system overhead, but can result in much higher performance for network transfers if the sender is waiting on the receiver's acknowledgement.

tcp_recvspace

The tcp_recvspace tunable specifies how many bytes of data the receiving system can buffer in the kernel on the receiving socket queue. The attribute must specify a socket buffer size less than or equal to the setting of the sb_max network attribute.

- Default value: 16384
- DB2 recommended value: 262144
- How to set:
  - **`ifconfig en0 tcp_recvspace 262144`** (interface specific, dynamic, no reboot required)
  - **`no -p -o tcp_recvspace=262144`**
- Type: Connect

With RFC1323 enabled, the recommended value ensures high data transfer rate. Higher value might increase the workload on the adapter. Also, it decreases the memory space for data in RAM. Monitor using **`vmstat`** for memory usage and paging.

tcp_sendspace

The tcp_sendspace tunable specifies how much data the sending application can buffer in the kernel before the application is blocked on a send call.

- Default value: 16384
- DB2 recommended value: 262144
- How to set:
  - **`ifconfig en0 tcp_sendspace 262144`** (interface specific, dynamic, no reboot required)
  - **`no -p -o tcp_sendspace= 262144`**
- Type: Connect

With RFC enabled, the recommended value ensures high data transfer rate. Higher values might increase the workload on the adapter. Also, it decreases the memory space for data in RAM. Monitor using **`vmstat`** for memory usage and paging.

Note:

- AIX6.1 enables use_isno by default and is a restricted variable. Because this overrides the global network option values, it is recommended to set the interface specific option using **`ifconfig`** for rfc1323,tcp_sendspace and tcp_recvspace.
- The application can override all of these with the setsockopt() subroutine.

tcp_tcpsecure

This option protects TCP connections from the following three vulnerabilities:

- The first vulnerability involves the sending of a fake SYN to an established connection to abort the connection. A tcp_tcpsecure value of 1 provides protection from this vulnerability.
- The second vulnerability involves the sending of a fake RST to an established connection to abort the connection. A tcp_tcpsecure value of 2 provides protection from this vulnerability.
- The third vulnerability involves injecting fake data in an established TCP connection. A tcp_tcpsecure value of 4 provides protection from this vulnerability.

Values of 3, 5, 6, or 7 protects the connection from combinations of these three vulnerabilities.

- Default value: 0
- DB2 recommended value: 5
- How to set: no -p -o tcp_tcpsecure=5
- Type: Dynamic

These settings protects TCP connections against vulnerabilities.

jumbo frames

With the advent of Gigabit Ethernet, the TCP/IP protocol now provides an ability to send large frames (called *jumbo frames*). An Ethernet frame contains 1500 bytes of user data, plus its headers and trailer. By contrast, a jumbo frame contains 9000 bytes of user data, so the percentage of overhead for the headers and trailer is much less and data-transfer rates can be much higher.

This feature is particularly useful in environments such as replication, where large data sets are transferred over the network. DB2 can make use of this features by enabling it.

Maximum transmission unit (MTU)

This is the maximum size of a frame that can be sent on the wire. The MTU size of the network can have a large impact on performance. The following list details the recommendations:

- All devices on the same physical network, or logical network if using VLAN tagging, must have the same MTU size.

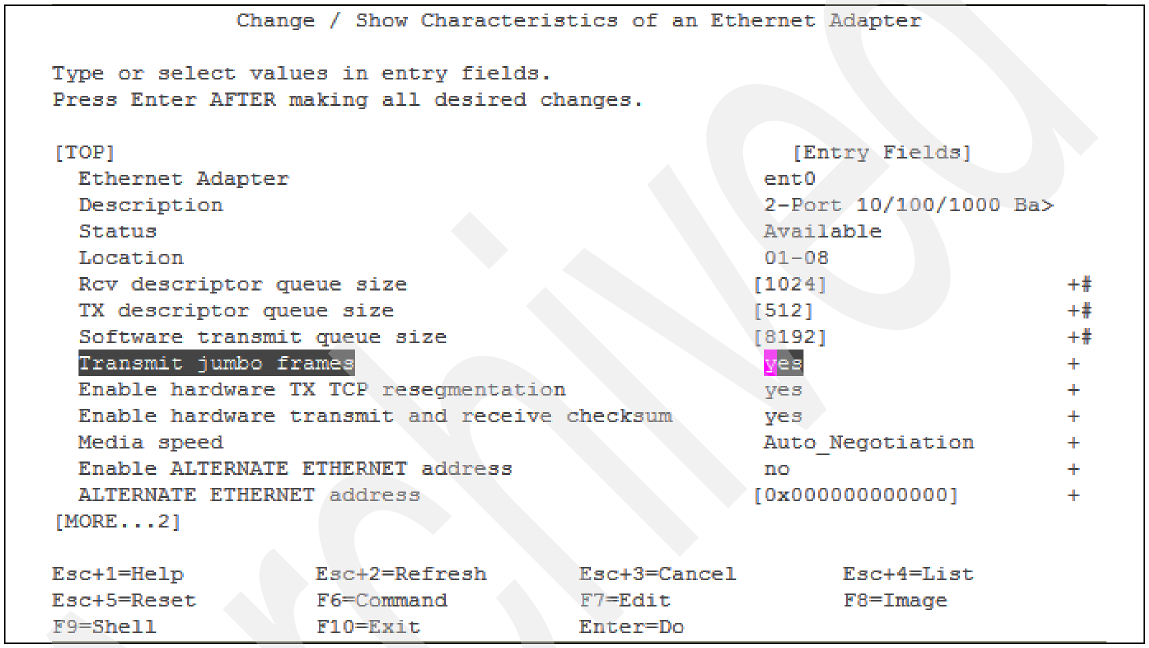
  For example, it is not recommended to have a Gigabit Ethernet adapter using jumbo frame mode with a MTU size of 9000 bytes, while other adapters in the network use the default MTU size of 1500 bytes.

- Configure Ethernet switches to use jumbo frames if jumbo frames are supported on your Ethernet switch.
- With Gigabit and 10 Gigabit Ethernet, if all of the machines in the network have Gigabit Ethernet adapters and no 10/100 adapters in the network, then it is best to use jumbo frame mode. For example, a server-to-server connection within the computer lab can typically be done using jumbo frames.
- It is important to select the MTU size of the adapter early in the network setup so that all the devices and switches can be properly configured.

We explain the steps to enable jumbo frames on Gigbit Ethernet. Trying to change the MTU size with the **ifconfig** command does not work. Use SMIT to display the adapter settings with the following steps:

1. Select **Devices**.
2. Select **Communications**.
3. Select **Adapter Type**.
4. Select **Change/Show Characteristics of an Ethernet Adapter**.
5. Change the **Transmit Jumbo Frames** option from **No** to **Yes**.

The SMIT panel looks like Figure 2-5.



```
                 Change / Show Characteristics of an Ethernet Adapter

Type or select values in entry fields.
Press Enter AFTER making all desired changes.

[TOP]                                                    [Entry Fields]
  Ethernet Adapter                                     ent0
  Description                                          2-Port 10/100/1000 Ba>
  Status                                               Available
  Location                                             01-08
  Rcv descriptor queue size                           [1024]                +#
  TX descriptor queue size                            [512]                 +#
  Software transmit queue size                        [8192]                +#
  Transmit jumbo frames                                yes                  +
  Enable hardware TX TCP resegmentation                yes                  +
  Enable hardware transmit and receive checksum        yes                  +
  Media speed                                          Auto_Negotiation     +
  Enable ALTERNATE ETHERNET address                    no                   +
  ALTERNATE ETHERNET address                          [0x000000000000]      +
[MORE...2]

Esc+1=Help          Esc+2=Refresh       Esc+3=Cancel        Esc+4=List
Esc+5=Reset         F6=Command          F7=Edit             F8=Image
F9=Shell            F10=Exit            Enter=Do
```

Figure 2-5 Enabling "jumbo frames" using smitty

Note: Jumbo frames can be used with Etherchannel. If set to **Yes** it forces jumbo frames on all underlaying adapters. Virtual Ethernet adapters do not have jumbo frames attributes, but can send jumbo sized packets.

Monitoring the network performance

You can use the following features to monitor the network performance:

- Kernel memory use by network packets:

  Kernel memory buffers called mbufs are used to store data in the kernel for incoming and outbound network traffic. Incorrect configuration affects both network and system performance.

  mbuf is controlled by:

  – "thewall" network variable. No tuning is required as the system automatically tunes the value ranging from 0-64GB. Default value is 1 GB.

  – "maxmbuf" system variable defines the maximum real memory allowed for MBUFS. If this value is "0", it indicates that the "thewall" value is used. If "maxmbuf >0", then it overrides the "thewall" value. Following command gives the "maxmbuf" value.

  ```
  # lsattr -E -l sys0 | grep maxmbuf
  maxmbuf       0 Maximum Kbytes of real memory allowed for MBUFS
  True
  ```

  – "maxmbuf" value can be changed using **chdev.**

  ```
  chdev -l sys0 -a maxmbuf=1048576
  ```

  Monitor the network memory usage using the following command.

  ```
  echo "kmbucket -s" | kdb | egrep "thewall|allocated" | tr "." " "
  | awk
  '/thewall/ {thewall=$2} /allocated/ {allocated=$2} END{ print
  "ibase=16;scale=5;",allocated,"/400/",thewall}'| bc
  ```

  If the output shows .02063, it means 2.063% of mbuf has been allocated. The threshold is indicated by sockthresh, which is 85% by default. After the allocated memory reaches 85% of the thewall or maxmbuf value, a new socket connection fails with ENOBUFS, until the buffer usage drops below 85%

  Monitor for any mbuf requirement failures using the **netstat -m** command. See Example 2-16.

Example 2-16 netstat -m , showing the memory statistics

```
# netstat -m

Kernel malloc statistics:

******* CPU 0 *******
By size           inuse     calls failed   delayed    free   hiwat   freed
64                  539   1347842      0         6      37    1744       0
128                2098    644202      0        62     334     872       0
```

```
256                870    187644     0       52     202    1744       0
512               2440  35547446     0      267     224    2180       0
1024               534    694480     0      129      26     872       0
2048               594   3269944     0      270     190    1308       0
4096               135      1201     0       26      20     436       0
8192                 4        66     0        2       6     218       0
16384              128       134     0       18      10     109       0
32768               24        24     0        6       0      54       0
65536               59        59     0       30       0      54       0
131072               3         3     0        0      26      52       0
<rest of the cpu output is truncated for clarity>

Streams mblk statistic failures:
0 high priority mblk failures
0 medium priority mblk failures
0 low priority mblk failures
```

In this example:

– By size: Shows the size of the buffer.
– inuse: Shows the number of buffers of that size in use.
– failed: Shows how many allocation requests failed because no buffers were available.

Note:

- You do not see a large number of failed calls. There might be a few, which trigger the system to allocate more buffers as the buffer pool size increases. If the requests for mbuf (failed) is high over a period of time, consider increasing the thewall value.

  `no -o thewall=<newvalue>`

- After the thewall value is increased, use **vmstat** to monitor total memory use, paging activity to understand whether the increase has any negative impact on overall memory performance.
- Kernel memory buffer (mbuf) to handle data packets are pinned kernel memory in RAM. These memory are never paged out. Increase in mbuf decreases the system memory for data segment.

- Monitoring the network statistics using **netstat**:

  The **netstat** command can be used to determine the amount of traffic in the network.

  Use **netstat -an -f inet** and check columns Recv-Q and Send-Q to see the use of the send/recvspace buffers. This also gives you the number of connections. See Example 2-17.

Example 2-17 netstat -an -f inet

```
# netstat -an -f inet | more
Active Internet connections (including servers)
Proto Recv-Q Send-Q  Local Address          Foreign Address        (state)
tcp        0      0  *.*                    *.*                    CLOSED
tcp4       0      0  *.13                   *.*                    LISTEN
tcp        0      0  *.21                   *.*                    LISTEN
tcp4       0      0  *.22                   *.*                    LISTEN
tcp        0      0  *.23                   *.*                    LISTEN
tcp4       0      0  *.25                   *.*                    LISTEN
tcp4       0      0  *.37                   *.*                    LISTEN
tcp4       0      0  *.111                  *.*                    LISTEN
tcp        0      0  *.199                  *.*                    LISTEN
tcp        0      0  *.512                  *.*                    LISTEN
tcp        0      0  *.513                  *.*                    LISTEN
tcp        0      0  *.514                  *.*                    LISTEN
```

  Use **netstat -i** to understand the packets received (IPkts), the number of input errors (Ierrs), packets transmitted (OPkts), and number of output errors (Oerrs). See Example 2-18.

Example 2-18 netstat -i

```
# netstat -i
Name  Mtu   Network     Address             Ipkts Ierrs    Opkts Oerrs  Coll
en0   1500  link#2      a.c.bb.1c.eb.b    27493958     0  3947570     0     0
en0   1500  9.184.65    indus65           27493958     0  3947570     0     0
```

  If the number of errors during input packets is greater than 1% of the total number of input packets, that is, `Ierrs > 0.01 x Ipkts`, run the **netstat -m** command to check for a lack of memory.

Note: The **nmon** command can be used to understand the network statistics. Run **nmon** and type `n` to view the details.

2.1.7 Input and output tunable considerations

Input/output tunable parameters are managed through the **ioo** command.

AIX ioo command

This command sets or displays the current or next boot values for all input and output tunables. To display the current value for all the tunable, **ioo -a** is used, as shown in Example 2-19.

Example 2-19 Output of the command, ioo -a, to display the current value

```
# ioo -a
                    aio_active = 0
                   aio_maxreqs = 65536
                aio_maxservers = 30
                aio_minservers = 3
         aio_server_inactivity = 300
         j2_atimeUpdateSymlink = 0
```

Similar to VMM and network tunables, IO tunable also has a set of restricted tunables, which can be viewed with **ioo -F -a** command options.

Note: Restricted tunables are not recommended to be modifed. They exist and must be modified primarily for specialized intervention by the development support or development teams. Contact IBM Support if any modification is required.

To modify the tunable and to make the modification permanent use the **ioo -p -o <tunable>=<value>** command. See Example 2-20.

Example 2-20 To modify and make the changes permanent

```
# ioo -p -o lvm_bufcnt=10
Setting lvm_bufcnt to 10 in nextboot file
Setting lvm_bufcnt to 10
```

To modify the tunable and to make the modification in the next boot, use the **ioo -r -o <tunable>=<value>** command. See Example 2-21.

Example 2-21 To modify and make the changes after reboot

```
# ioo -r -o lvm_bufcnt=10
Setting lvm_bufcnt to 10 in nextboot file
Warning: changes will take effect only at next reboot
```

From the DB2 perspective, the default AIX6 **ioo** tunables are recommended. Some of the tunables that needs attention are covered in the following sections . Though no changes are recommended, constant monitoring using **vmstat -v** and **iostat -D** helps to understand the impact of these parameters.

j2_maxPageReadAhead

This parameter defines the number of pages that needs to be read ahead during the sequential file read operation on JFS2. This parameter is taken into consideration while setting the minfree/maxfree value, as discussed in 2.1.3, "VMM considerations for DB2" on page 40.

- Default value: 128
- DB2 recommended value: 128

j2_maxRandomWrite

This parameter specifies a threshold for random writes to accumulate in RAM before subsequent pages are flushed to disk by the JFS2's write-behind algorithm.The random write-behind threshold is on a per-file basis.

For DB2, we recommend that you use the default value. The default value of "0" disables random write-behind and indicates that random writes stay in RAM until a sync operation. If **vmstat n** shows page out and I/O wait peaks on regular intervals (usually when the sync daemon is writing pages to disk), useful to set this value to 1 or higher if too much I/O occurs when **syncd** runs.

- Default value: 0
- DB2 recommended value: 0
- How to set: ioo -p -o j2_maxRandomWrite=<value>
- Type: Dynamic

j2_minPageReadAhead

This parameter specifies the minimum number of pages to be read ahead when processing a sequentially accessed file on Enhanced JFS. This is useful for sequential access. A value of 0 might be useful if the I/O pattern is purely random. For DB2, we recommend that you use the default value, which is 2.

- Default value: 2
- DB2 recommended value: 2

maxpgahead

This parameter specifies the maximum number of pages to be read ahead when processing a sequentially accessed file. This is a restricted tunable in AIX 6.1. For DB2, we recommend that you use the default value, which is 8.

- Default value: 8
- DB2 recommended value: 8

minpgahead

This parameter specifies the number of pages with which sequential read-ahead starts. This is a restricted tunable in AIX6.1. For DB2, we recommend that you use the default value, which is 2.

- Default value: 2
- DB2 recommended value: 2

Tip: Set both the maxnpgahead and minpgaheadparameters to a power of two. For example, 2, 4, 8,...512, 1042... and so on.

maxrandwrt

This parameter specifies a threshold (in 4 KB pages) for random writes to accumulate in RAM before subsequent pages are flushed to disk by the write-behind algorithm. A value of 0 disables random write-behind and indicates that random writes stay in RAM until a sync operation. Setting maxrandwrt ensures these writes get flushed to disk before the sync operation has to occur.

- Default value: 0
- DB2 recommended value: 0
- How to set: ioo -p -o maxrandwrt=<threshold value>
- Type: Dynamic

If `vmstat n` shows page out and I/O wait peaks on regular intervals (usually when the sync daemon is writing pages to disk), set it to a threshold level to reduce the high I/O. However, this can degrade performance because the file is being flushed each time after the threshold value is reached. Tune this option to favor interactive response time over throughput. After the threshold is reached, all subsequent pages are immediately flushed to disk.

j2_nBufferPerPagerDevice

Specifies the number of file system bufstructs for JFS2.This is a restricted tunable. For DB2, we recommend that you use the default value, which is 512.

- Default value: 512
- DB2 recommended value: 512

numfsbufs

This parameter specifies the number of file system bufstructs. This is a restricted tunable. For DB2, we recommend that you use the default value, which is 196.

- Default value: 196
- DB2 recommended value: 196

Note: If you make any changes to the file system bufstructs tunables (j2_nBufferPerPagerDevice and numfsbufs), the new values take effect only when the file system is remounted.

j2_nPagesPerWriteBehindCluster

JFS file systems and JFS2 file systems are partitioned into 16 KB partitions or 4 pages. Each of these partitions is called a *cluster*. In JFS2, this parameter specifies the number of pages per cluster processed by JFS2's write behind algorithm.

- Default value: 32
- DB2 recommended value: 32

This is useful to increase if there is a need to keep more pages in RAM before scheduling them for I/O when the I/O pattern is sequential. It might be appropriate to increase if striped logical volumes or disk arrays are being used.

pv_min_pbuf

This specifies the minimum number of pbufs per PV that the LVM uses.This is a global value that applies to all VGs on the system. The `lvmo` command can be used to set a value for a particular VG. In this case, the higher of the two values is used for this particular VG.

- AIX default: 512
- DB2 recommended value: 512

Note: This is a restricted tunable in AIX. The best practise is to monitor "pending disk I/Os blocked with no pbuf" in `vmstat -v` and discuss with IBM technical support to consider any further modification.

sync_release_ilock

This parameter specifies whether the i-node lock is to be held or not while flushing I/O to file when the sync daemon is running.

If set, flush all I/O to a file without holding the i-node lock, and use the i-node lock to do the commit.

A value of 0 indicates off and that the i-node lock is held while all dirty pages of a file are flushed. I/O to a file is blocked when the syncd daemon is running.

This is a restricted tunable in AIX. For DB2, we recommend that you use the default value, which is 0.

- Default value: 0
- DB2 recommended value: 0

Note: The j2_ parameters are not applicable for DB2 table spaces configured with the FILE SYSTEM CACHING DB2 parameter, where the file system caching is not involved.

Asynchronous IO (AIO) consideration for DB2

Asynchronous Input Output (AIO) is a software subsystem within AIX that allows a process to issue an I/O operation and continue processing without waiting for the I/O to finish.

Asynchronous I/O operations run in the background and do not block user applications. This improves performance because I/O operations and applications processing run simultaneously. Applications such as databases and file servers take advantage of the ability to overlap processing and I/O.

There are two AIO subsystems:

- Legacy AIO
- Posix AIO

The difference between them is parameter passing. Both the subsystem can run concurrently in the system.

From AIX6.1 onwards, no AIO servers are started by default. They are started when applications initiate AIO requests and stay active as long as they are servicing requests.

- When AIO is enabled, minimum number of AIO servers are created based on the minserver value.
- If additional servers are required, more AIO servers are added up to the maximum, based on the maxserver value.
- The maximum number of outstanding requests is defined by maxreqs.
- These are all **`ioo`** tunables. Changes do not require system reboot.

Note: In AIX6 all AIO parameters become the **`ioo`** command tunables. The **`aioo`** command used in the previous version of AIX5.3 is removed.

From the DB2 perspective, because the AIO tuning is dynamic, no tuning is required.

- Default AIO settings of minserver, maxserver, maxreqs are recommended. No change is required.
- AIO fast path (aio_fastpath) is also enabled by default, so no changes are required. This is a restricted tunable, as well.
- AIO CIO fastpath (aio_fsfastpath) is also enabled by default and so, no changes are required. This is a restricted tunable, as well.

Note: AIO can be used with file systems mounted with DIO/CIO. DB2 uses DIO/CIO when the table space is used with NO FILE SYSTEM CACHING.

AIO monitoring and tuning suggestions

The following list describes AIO monitoring and tuning suggestions:

- To understand the number of active AIO servers, use the **`pstat -a | grep -c aioscan`** command.
- To get the general AIO report, use the **`iostat`** command:

  ```
  iostat -AQ 1 5
  ```

  This command displays the output for every second. Monitoring has to be done over a period of time to understand the status of aio server.
- To understand the state of the each active AIO servers, use the following command:

  ```
  ps -vg | egrep “aio | SIZE”
  ```

  From the output in Example 2-22, we can identify the %CPU,%MEM and size consumed by the each aio kernel process.

Example 2-22 ps vg | egrep "aio|SIZE"

```
# ps vg | egrep "aio|SIZE"
    PID    TTY STAT  TIME PGIN  SIZE   RSS   LIM  TSIZ   TRS %CPU %MEM COMMAND
 127048      - A     0:00    9   448   448    xx     0     0  0.0  0.0 aioserver
 385024      - A     0:00    8   448   448    xx     0     0  0.0  0.0 aioserver
 442584      - A     0:00    9   448   448    xx     0     0  0.0  0.0 aioserver
 446682      - A     0:00   11   448   448    xx     0     0  0.0  0.0 aioserver
 450780      - A     0:00    7   448   448    xx     0     0  0.0  0.0 aioserver
 454878      - A     0:00    8   448   448    xx     0     0  0.0  0.0 aioserver
 458976      - A     0:00    8   448   448    xx     0     0  0.0  0.0 aioserver
 463074      - A     0:00    7   448   448    xx     0     0  0.0  0.0 aioserver
 467172      - A     0:00    7   448   448    xx     0     0  0.0  0.0 aioserver
 471270      - A     0:00    5   448   448    xx     0     0  0.0  0.0 aioserver
 475368      - A     0:00    6   448   448    xx     0     0  0.0  0.0 aioserver
 479466      - A     0:00    6   448   448    xx     0     0  0.0  0.0 aioserver
 483564      - A     0:00    2   448   448    xx     0     0  0.0  0.0 aioserver
 487662      - A     0:00    0   448   448    xx     0     0  0.0  0.0 aioserver
```

- Tuning consideration for the aio_maxservers is as follows. Monitor the status over the period of time using these commands.
  – If all the active aio servers are consuming higher CPU%, then the maxservers value can be increased by 10%. For example, increase the default value of 30 to 33.
  – If a few of the aio servers are consuming less CPU, the system has the required server as it needs.
  – To modify the “aio_maxservers” value, following option has to be used **`ioo -o aio_maxservers=<value>`**. It does not require a reboot. See Example 2-23 on page 69 for an example on how to modify the aio-maxservers value.

Example 2-23 changing aio_maxservers value

```
# ioo -o aio_maxservers=33
Setting aio_maxservers to 33
# ioo -a | grep aio
                    aio_active = 0
                   aio_maxreqs = 65536
                aio_maxservers = 33
                aio_minservers = 3
         aio_server_inactivity = 300
```

Note: AIX6 introduces a new parameter, aio_server_inactivity, that controls how long in seconds the AIO server sleeps waiting for work. The main benefit at the AIX layer is to free pinned memory and decrease the number of processes after a peak workload activity with the AIO subsystem, which helps lighten the load process scheduling and reduce system resource usage.

The default value is 300 seconds.

In summary, for DB2 we do not recommend any changes to the default **ioo** tunable values in AIX 6.1. Any further modification is to be considered based on the monitoring over a period of time as described.

2.1.8 Scheduler tunable considerations

Processor scheduler tunable parameters are managed with the command **schedo**. Similar to other tunables such as VMM, IO and network, scheduler tunables are also classified into both restricted and non-restricted tunables.

In general, we do not recommend that you make any changes with respected to the scheduler tunables for DB2 environments. From the AIX tunable perspective, apart from the recommended modifications, if a performance degradation is still observed, relevant performance data is submitted to IBM for analysis.

Note: There might be specific cases where changing the restricted values might improve the performance. In this situation, contact technical support for further analysis. It is never recommended to change the restricted tunables from their defaults.

2.1.9 DB2 Groups, users, and password configuration on AIX

You can create required groups, users and set initial password using the commands in Example 2-24.

Example 2-24 Create required groups, users and set initial password

```
mkgroup id=999 db2iadm1
mkgroup id=998 db2fadm1
mkgroup id=997 dasadm1
mkuser id=1004 pgrp=db2iadm1 groups=db2iadm1 home=/db2home/db2inst1 db2inst1
mkuser id=1003 pgrp=db2fadm1 groups=db2fadm1 home=/db2home/db2fenc1
db2fenc1
mkuser id=1002 pgrp=dasadm1 groups=dasadm1 home=/db2home/dasusr1 dasusr1
passwd db2inst1
passwd db2fenc1
passwd dasusr1
```

Note: To enable long password support on the AIX 6.1, install APAR IZ35001.

2.1.10 DB2 user ID resource limits (ulimits)

For DB2 instance owner, fenced user, and administration user IDs set user process resource limits (ulimits) to unlimited using the **chuser** command, as shown in Example 2-25.

Example 2-25 chuser command

```
chuser fsize=-1 fsize_hard=-1 data=-1 data_hard=-1 stack=-1 stack_hard=-1 rss=-1
       rss_hard=-1 nofiles=-1 nofiles_hard=-1 db2inst1
```

2.1.11 File system mount point permissions

DB2 installation in a separate file system:

- For root installations when DB2 product is installed on separate file system or on a non-default directory, ensure others are given read and execute permission for underlying mount point.

  For example, when DB2 is installed in `/db2bin/opt/IBM/db2/V9.7/` and `/db2bin` is a separate file system, ensure others have read and execute file permissions on empty `/db2bin` mount point directory

  This allows non-root users to successfully execute the **db2ls** command. Using **db2ls**, you can list installed DB2 products and features.

DB2 instance directory in a separate file system:

In scenarios where a separate file system is used for DB2 instance home directory, ensure that DB2 instance ID owns the mount directory (home dir) for successful DB2 instance creation.

For example db2inst1 must own the `/db2home/db2inst1` directory when a separate file system is mounted on it.

2.1.12 File system mount options

The following list details the file system mount options:

- Mount option rbrw is recommended for specific file systems where FILE SYSTEM CACHING is used to get better performance (such as temporary table spaces, table spaces with LONG and LOB data, BACKUP file systems).

  The option rbrw specifies that when sequential read or sequential write of a file in this file system is detected, the real memory pages used by the file are released after the pages are written or read to disk. This helps in improving the performance of syncd, which flushes the real memory pages to disk.
- Do not use explicit dio/cio mount options for file systems, which are used for DMS table spaces defined with NO FILE SYSTEM CACHING as DB2 chooses the best I/O method required.

2.1.13 Maximum number of AIX processes allowed per user:

DB2 users (such as the fenced user ID) can create all the required processes. The maxuproc kernel parameter helps in limiting the number of process that can be created by a user. DB2 recommends the value of 4096.

Verify the existing “maxuproc” value as follows:

```
lsattr -E -l sys0 -a maxuproc
```

Monitor the maximum number of processes under DB2 instance ID using the **`ps –fu  db2inst1`** or **`db2_local_ps`** command and adjust the maxuproc value accordingly. The following example sets the maxuproc to 4096

```
chdev –l sys0 –a maxuproc=4096
```

Reboot is required to make the changes effective.

2.2 DB2 registry variables

In the following sections we discuss DB2 registry variables. The registry variables can be examined or modified using the **db2set** command.

DB2_LOGGER_NON_BUFFERED_IO

- Default value: AUTO
- Recommended: Default

See '3.2.2, "Tablespace design" on page 92' to read more about buffered and non-buffered IO.

Starting with Version 9.7, the default value for this variable is AUTOMATIC. When set to AUTOMATIC, active log files are opened with DIO. This eliminates the operating system overhead of caching database recovery logs. The database manager determines which log files benefit from using non-buffered I/O.

When set to ON, all active log files are always opened with DIO. When set to OFF all active log files are buffered I/O. In Version 9.5 Fix Pack 1 or later, the default was OFF.

Do not use any explicit mount options for the file systems used for DB2 log files.

DB2_USE_IOCP

- Default value: ON
- Recommended: Default

IOCP has to be configured before enabling this variable. This feature enables the use of AIX I/O completion ports (IOCP) to submit and collect asynchronous I/O (AIO) requests and enhance performance in a non-uniform memory access (NUMA) environment by avoiding remote memory access. This is also available on DB2 v9.5 starting from fixpack 3.

It is recommended to leave this parameter at its default. You might monitor the system IO statistics or **nmon** to fine-tune this parameter.

Note: DB2_USE_IOCP is ON by default since DB2 v9.7.

DB2_USE_FAST_PREALLOCATION

- Default value: ON
- Recommended: Default

This feature is to reserve table space and speed up the process of creating or altering large table spaces. This is available from DB2 v9.7 fixpack 1 and DB2 v9.5 fixpack 5.

DB2_PARALLEL_IO

- Default value: NULL
- Recommended: Refer to Storage chapter to set an optimum value for this registry variable.

2.2.1 DB2_Resource_policy

In the following sections we discuss more advanced DB2 registry variables.

DB2_RESOURCE_POLICY

- Default value:
- Recommended:

This variable defines a resource policy that can be used to limit what operating system resources are used by the DB2 database. It can contains rules for assigning specific operating system resources to specific DB2 database objects.

This registry variable can be used to limit the set of processors that the DB2 database system uses. The extent of resource control varies depending on the operating system.

On AIX NUMA and Linux NUMA-enabled machines, a policy can be defined that specifies what resource sets the DB2 database system uses. When resource set binding is used, each individual DB2 process is bound to a particular resource set. This can be beneficial in performance tuning scenarios.

The following steps illustrate AIX resource sets configuration to enable processor affinity for DB2 partitions. A single node with 16 processors and eight logical database partitions is considered.

1. Define AIX resource sets in `/etc/rsets` file.

   Define eight new resource sets in the `/etc/rsets` files to use CPU Number 8 through 15. These 8eight resource sets are named as DB2/MLN[1-8], as shown in Example 2-26.

Example 2-26 Resource sets defined in /etc/rsets file

```
DB2/MLN1:
owner = db2tpch
group = db2admin
perm = rwr-r-
resources = sys/cpu.00008
DB2/MLN2:
owner = db2tpch
group = db2admin
perm = rwr-r-
resources = sys/cpu.00009
DB2/MLN3:
owner = db2tpch
group = db2admin
perm = rwr-r-
resources = sys/cpu.00010
DB2/MLN4:
owner = db2tpch
group = db2admin
perm = rwr-r-
resources = sys/cpu.00011
DB2/MLN5:
owner = db2tpch
group = db2admin
perm = rwr-r-
resources = sys/cpu.00012
DB2/MLN6:
owner = db2tpch
group = db2admin
perm = rwr-r-
resources = sys/cpu.00013
DB2/MLN7:
owner = db2tpch
group = db2admin
perm = rwr-r-
resources = sys/cpu.00014
```

```
DB2/MLN8:
owner = db2tpch
group = db2admin
perm = rwr-r-
resources = sys/cpu.00015
```

2. Update the AIX kernel.

   The newly defined resource sets are added to the kernel data structures using the following SMIT fast path:

   ```
   $ smit reloadrsetcntl
   ```

   This menu gives the option to reload the database now, at next boot, or at both times. Select *both* such that the rset loads now and after each reboot.

3. Update `db2nodes.cfg`.

   The `db2nodes.cfg` file resides in `INSTANCE_OWNER/sqllib` directory. The file format is:

   ```
   nodenum hostname logical_port netname resourcesetname
   ```

   After updating the eight resource names in the resourcesetname field, `db2nodes.cfg` looks like Example 2-27.

Example 2-27 Resource names in db2nodes.cfg file

```
1 CLYDE 0 CLYDE DB2/MLN1
2 CLYDE 1 CLYDE DB2/MLN2
3 CLYDE 2 CLYDE DB2/MLN3
4 CLYDE 3 CLYDE DB2/MLN4
5 CLYDE 4 CLYDE DB2/MLN5
6 CLYDE 5 CLYDE DB2/MLN6
7 CLYDE 6 CLYDE DB2/MLN7
8 CLYDE 7 CLYDE DB2/MLN8
```

2.2.2 DB2 memory registry variables

Following memory related registry parameters only applicable when STMM is enabled.

Use the **`nmon`**, **`vmstat`**, **`lsps`** commands to monitor and fine-tune the following memory-related registry parameters. Refer to 2.1.3, “VMM considerations for DB2” on page 40 for more information about monitoring and tuning VMM parameters.

DB2_LARGE_PAGE_MEM

- Default value: NULL
- Recommended: Refer to 2.1.4, “Large page considerations” on page 46

Use this registry variable to improve performance for memory access-intensive applications that use large amounts of virtual memory.

To enable the DB2 database system to use them, the operating system must be configured to use large or huge pages. Refer to 2.1.4, “Large page considerations” on page 46 for all pre-requisites and steps to set this parameter.

DB2MEMDISCLAIM

- Default value: YES
- Recommended: Default

Memory used by DB2 database system processes might have associated paging space. This paging space might remain reserved even when the associated memory has been freed. Whether or not this is much depends on the operating system's (tunable) virtual memory management allocation policy.
The DB2MEMDISCLAIM registry variable controls whether DB2 agents explicitly requests that the operating system disassociate the reserved paging space from the freed memory.

A setting of **Yes** results in smaller paging space requirements and less IO might provide performance benefit. However, when there is plenty of real memory and paging space, a setting of **No** might yield performance benefit.

You need to monitor the paging space use and VMM parameters (see 2.1.3, “VMM considerations for DB2” on page 40) and set this registry value accordingly.

DB2_MEM_TUNING_RANGE

- Default value: NULL
- Recommended: Default

Using this registry parameter, DB2 instance sets minimum and maximum free physical memory thresholds available to the server (for other applications). It is recommended to leave at default. The setting of this variable has no effect unless the self-tuning memory manager (STMM) is enabled and database_memory is set to AUTOMATIC.

You need to monitor the paging space use and other VMM parameters (see 2.1.3, “VMM considerations for DB2” on page 40) to adjust the thresholds

2.2.3 DB2 Communications registry variables

Use **nmon - Network Interface View (‘n’)** to monitor network performance and fine-tune this parameter. You might also use **netstat** and **entstat** to monitor and fine-tune any of the following TCP/IP registry parameters. See 2.1.6, “Network tunable considerations” on page 51.

Note: Following registry parameters override OS level settings.

DB2_FORCE_NLS_CACHE

- Default value: FALSE
- Recommended: Default

This variable is used to eliminate the chance of lock contention in multi-threaded applications. When this registry variable is **TRUE**, the code page and territory code information is saved the first time a thread accesses it. From that point, the cached information is used for any other thread that requests this information.

This eliminates lock contention and results in a performance benefit in certain situations. This setting must not be used if the application changes locale settings between connections. It is probably not needed in such a situation because multi-threaded applications typically do not change their locale settings because it is not thread safe to do so.

DB2TCPCONNMGRS

- Default value: 1
- Recommended:Default

This variable determines the number of TCP/IP connection managers on a given server. With multiple TCP/IP connection managers there is a performance boost on remote connections in systems with a lot of users, frequent connects, and disconnects, or both.

Note: Each additional TCP/IP connection manager has additional overhead to memory and storage.

Use **nmon - Network Interface View (‘n’)** to monitor network performance and fine-tune this parameter.

DB2SORCVBUF and DB2SOSNDBUF

- Default value: 65536
- Recommended: Default

The DB2SORCVBUF registry variable determines the value of TCP/IP receive buffers. DB2SOSNDBUF determines the value of TCP/IP send buffers.

Use **`nmon - Network Interface View (‘n’)`** to monitor network performance and fine-tune this parameter. You might also use /**`usr/bin/entstat –d <interface name>`** to display all the statistics.

DB2_HADR_SORCVBUF and DB2_HADR_SOSNDBUF

- Default value: Set by OS
- Recommended: Default

To maximize network and HADR performance, the TCP socket buffer sizes might require tuning. If you change the TCP socket buffer size at the system level, the settings are applied to all TCP connections on the machine. Setting a large system level socket buffer size consumes a large amount of memory.

These two registry variables allow tuning of the TCP socket send and receive buffer size for HADR connections only. They have the value range of 1024 to 4294967295 and default to the socket buffer size of the operating system, which varies depending on the operating system. Some operating systems automatically round or silently cap the user-specified value.

Note: HADR log shipping workload, network bandwidth, and transmission delay are important factors to consider when tuning the TCP socket buffer sizes. These factors can be monitored using **`nmon - Network Interface View (‘n’)`**.

DB2CHECKCLIENTINTERVAL

- Default value: 50
- Recommended: Default

This variable specifies the frequency of TCP/IP client connection verifications. For heavy DB2 workloads, setting this parameter to a high value (less frequent verifications) might provide performance benefit.

DB2TCP_CLIENT_CONTIMEOUT

- Default value: 0
- Recommended: Default

The DB2TCP_CLIENT_CONTIMEOUT registry variable specifies the number of seconds a client waits for the completion on a TCP/IP connect operation. If a connection is not established in the seconds specified, the DB2 database manager returns the error `-30081 selectForConnectTimeout`.

There is no timeout if the registry variable is not set or is set to 0.

Note: Operating systems also have a connection timeout value that might take effect prior to the timeout you set using DB2TCP_CLIENT_CONTIMEOUT. For example, AIX has a default tcp_keepinit=150 (in half seconds) that terminates the connection after 75 seconds.

DB2TCP_CLIENT_KEEPALIVE_TIMEOUT

- Default value: 0
- Recommended: Default

The DB2TCP_CLIENT_KEEPALIVE_TIMEOUT registry variable specifies the maximum time in seconds before an unresponsive connection is detected as no longer alive. When this variable is not set, the system default TCP/IP keep alive setting is used (typically two hours). Setting this parameter to a lower value than the system default allows the database manager to detect connection failures sooner, and avoids the need to reconfigure the system default, which impacts all TCP/IP traffic and not connections established by DB2.

DB2TCP_CLIENT_RCVTIMEOUT

- Default value: 0
- Recommended: Default

The DB2TCP_CLIENT_RCVTIMEOUT registry variable specifies the number of seconds a client waits for data on a TCP/IP receive operation. If data from the server is not received in the seconds specified, then the DB2 database manager returns the error `-30081 selectForRecvTimeout`.

There is no timeout if the registry variable is not set or is set to 0.

Note: The value of the DB2TCP_CLIENT_RCVTIMEOUT can be overridden by the CLI, using the db2cli.ini keyword ReceiveTimeout or the connection attribute SQL_ATTR_RECEIVE_TIMEOUT

2.2.4 DB2 Database Manager Configuration (DBM) parameters

There are many DB2 Database Manager Configuration (DBM) parameters that have to be set specific to each workload. On AIX DLPARs set CPUSPEED, COMM_BANDWIDTH to -1 such that they are computed appropriately by DB2.

2.2.5 DB2 Database Configuration (DB) parameters

Starting with DB2 v9.1, Self Tuning Memory Manager (STMM) feature has been added, which simplifies memory management by setting critical memory related parameters to AUTOMATIC.

Note: In addition to setting STMM to AUTOMATIC, the database configurations for the memory consumers are set to AUTOMATIC. These consumers include:

- bufferpool
- lock
- package cache
- sort heap

2.3 Configurational differences: AIX 5.3 versus AIX 6.1

Table 2-1 summarizes the configurational differences between AIX 5.3 and AIX 6.1.

Table 2-1 Tunables: AIX 5.3 versus AIX 6.1

Parameter group	Parameter	AIX 5.3 default setting	AIX 6.1 default setting	DB2 best practice	Comments
ioo	j2_maxPageRead Ahead	128	128	128	The vmo tunable minfree is at least set to this value.
ioo	j2_maxRandomWrite	128	0	0	Random write behind is disabled hence the file pages flushed to disk only during sync.
ioo	j2_minPageRead Ahead	2	2	2	No change in value from AIX 5.3.
ioo	j2_nBufferPerPager Device	512	512	512	Restricted tunable in AIX 6.1.

Parameter group	Parameter	AIX 5.3 default setting	AIX 6.1 default setting	DB2 best practice	Comments
ioo	maxpgahead	128	8	8	Restricted tunable in AIX 6.1
ioo	maxrandwrt	128	0	0	Restricted tunable in AIX 6.1.Similar to JFS Random write.
ioo	minpgahead	2	2	2	No change in value from AIX 5.3.
ioo	numfsbufs	4096	196	196	Restricted tunable in AIX 6.1
ioo	sync_release_ilock	1	0	0	Restricted tunable in AIX 6.1.
no	bcastping	0	0	0	No change in value from AIX 5.3.
no	clean_partial_conns	1	0	1	Enabled to overcome SYNC attack.
no	directed_broadcast	0	0	0	No change in value from AIX 5.3.
no	extendednetstats		0	0	Restricted tunable in AIX 6.1.
no	icmpaddresmask	0	0	0	Disable Internet Control Message Protocol (ICMP) query message. No change in value from AIX 5.3.
no	ip6srcrouteforward	0	1	0	IPv6 source forwarding is disabled.
no	ipforwarding	0	0	0	No change from AIX 5.3 recommendation.Kernel does not involve in IP forwarding.
no	ipignoreredirects	1	0	1	Used to control ICMP redirects and setting it to 1 ensures that malicious ICMP request cannot be used to create manipulated routes.

Parameter group	Parameter	AIX 5.3 default setting	AIX 6.1 default setting	DB2 best practice	Comments
no	ipqmaxlen1	250	100	250	Default will not handle large workloads. Hence increased to 250.
no	ipsendredirects	0	1	0	Disabled to prevent illegal access through source routing.
no	ipsrcrouteforward	1	1	1	Source route forwarding is encouraged.
no	ipsrcrouterecv	0	0	1	We recommend 1 as required by topsvcs.
no	ipsrcroutesend	1	1	1	No change in value from AIX 5.3.
no	nonlocsrcroute	0	1	1	Source route encouraged
no	rfc1323	1	0	1	The rfc1323 tunable enables the TCP window scaling option.
no	sb_max	1310720	104857 6	1048576	Number of socket buffer queued with each sockets. More socket buffer space provided compared with AIX 5.3.
no	tcp_mssdflt		1460	1460	Segment size used in communicating with remote networks.
no	tcp_nagle_limit	1	65535	1	DB2 already disable this per connection. This tunable will ensure AIX does not try to consolidate packets.
no	tcp_nodelayack	1	0	1	Does not want to delay the ACK.
no	tcp_pmtu_discover	0	1	0	We recommend 0 to prevent illegal access through source routing.

Parameter group	Parameter	AIX 5.3 default setting	AIX 6.1 default setting	DB2 best practice	Comments
no	tcp_recvspace	262144	16384	262144	The tcp_recvspace tunable specifies how many bytes of data the receiving system can buffer in the kernel on the receiving sockets queue. This can also be set through DB2 overrides.
no	tcp_sendspace	262144	16384	262144	The tcp_sendspace tunable specifies how much data the sending application can buffer in the kernel before the application is blocked on a send call. This can also be set through DB2 overrides.
no	tcp_tcpsecure3		0	5	
no	use_isno4	0	1	1	Restricted tunable in AIX 6.1.
vmo	cpu_scale_memp	8	8	8	Restricted tunable in AIX 6.1.
vmo	page_steal_method	0	1	1	Restricted tunable in AIX6.1
vmo	lru_file_repage	1p	0	0	Restricted tunable in AIX 6.1.
vmo	maxclient%	80	90	90	Restricted tunable in AIX 6.1.
vmo	maxfree	minfree+ 512	1088	minfree+ 512	Difference between minfree and maxfree must not be more than 1000.
vmo	maxperm%	80	90	90	Restricted tunable in AIX 6.1.
vmo	memory_affinity	1	1	1	Restricted tunable in AIX 6.1.

Parameter group	Parameter	AIX 5.3 default setting	AIX 6.1 default setting	DB2 best practice	Comments
vmo	minfree	4096 [memory less than 8 GB] 8192 [memory greater than 8 GB	960	4096	It is at least the size of minimum page read ahead value.
vmo	minperm%	20	3	3	Set to low value to avoid computational pages to be paged.
vmo	strict_maxclient	0	1	1	Restricted tunable in AIX 6.1.

3

Storage layout

This chapter describes the storage principles guiding the storage layout, the mapping of physical storage to logical storage, the layout of file systems, and how DB2 uses them.

This chapter has the following sections:

3.1 Introduction

The main objective of a database is to store and retrieve data quickly and efficiently. One of the most important considerations when designing a database is its storage layout. Where the data is placed can significantly affect the performance of the operations against the database, and when designed and implemented it can be hard to change. Therefore, it is important for database administrators (DBAs) to understand the advantages and disadvantages of choosing a particular strategy. Furthermore, understanding the concepts related to the physical layout is essential when DBAs have to discuss with the system administrator how the storage needs to be configured

In this chapter, we refer to two scenarios: one for OLTP environments and one for data warehouse environments, as each has its own requirements. Figure 3-1 and Figure 3-2 on page 87 show a high level graphical view of a typical storage layout in both environments.

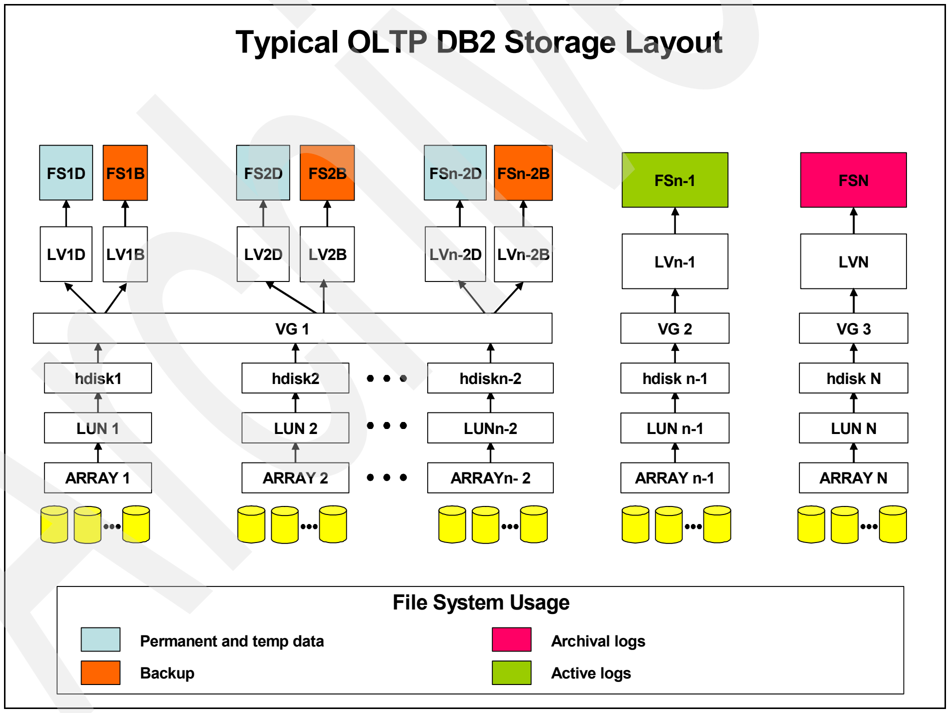


Figure 3-1 Typical OLTP storage layout

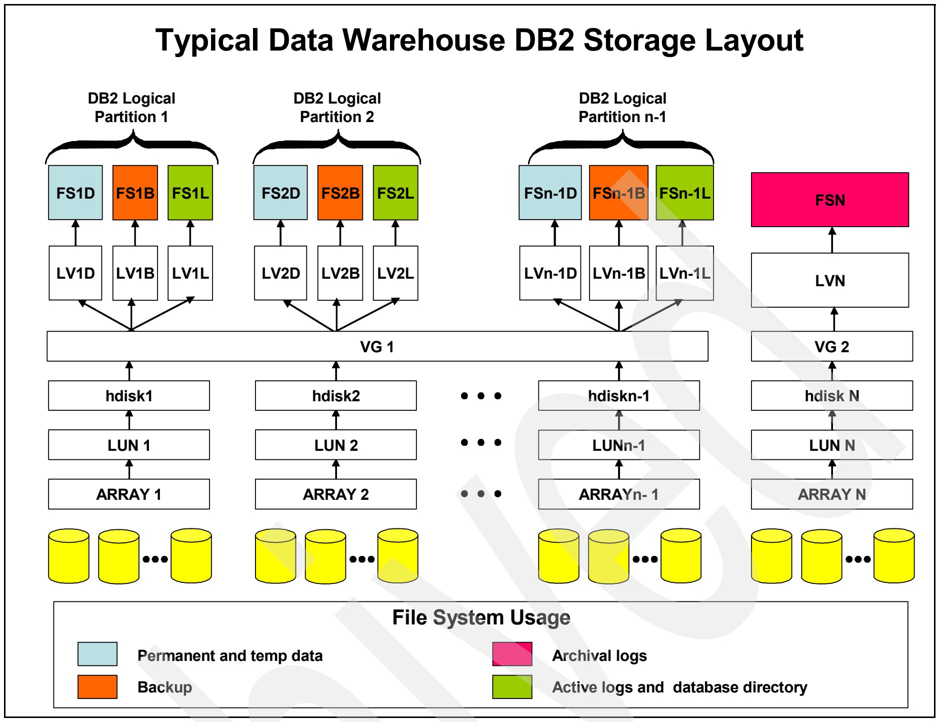


Figure 3-2 Typical Data Warehouse storage layout

In the rest of this chapter we discuss the details on how to build and configure such environments for best performance.

3.2 DB2 storage design

The first step in designing a database storage layout is to understand what storage space is needed. In a DB2 database, storage is required for the following data categories:

- Instance directory
- Catalog table and database directory
- Permanent data: Tables and indexes
- Temporary data
- Transaction logs
- Backup data and archived logs

One simple but effective design is to store each of these data categories in a dedicated file system. These file systems are resided on separate physical disks. Further discussion on the mapping between file systems and disk spindles is presented in subsequent sections.

This simple design ensures that categories of data are being accessed without interfering with others. As a result, the database system can achieve better performance and availability. In an OLTP (online transaction processing) environment, in particular, separating transaction logs from the permanent/temporary data is important because of its high transaction rate. For a DW (data warehouse) environment, however, permanent data and transaction logs might be collocated in the same physical disks. This is because DW workloads are often read-only and do not have as high transaction rate as OLTP workloads. Thus, the demand on transaction logs is lower. Allowing the sharing of disks gives DBAs the flexibility to manage storage space and better use any unused space.

On the other hand, this simple design can help with problem determination. For example, if we observe that a set of disks corresponding to a file system is much busier than the others, we can identify which part of the system or data is hot. Better parallelism or more resources (for example, more number of spindles) might be needed to alleviate the issue.

3.2.1 Automatic storage

To make storage management easier, starting from V8.2 FP9, DB2 introduces a feature called *automatic storage*. This feature allows storage to be managed at the database level. The responsibility of creating, extending, and adding containers is taken over by the database manager. We no longer need to explicitly define containers for the table space.

In DB2 v9.7, all databases are created with automatic storage[1] by default. In particular, when creating a database, we establish one or more initial storage paths in which table spaces have their containers. For example:

```
CREATE DATABSE TESTDB on /path1, /path2
```

This command creates the database with two storage paths: `/path1` and `/path2`. The recommendation is for each storage path to reside in a separate file system. With the **`CREATE DATABASE`** command, the default table spaces created by DB2 (for example, SYSCATSPACE, USERSPACE1, and TEMPSPACE) now have two containers, one on each of the two storage paths.

[1] If you do not want to use automatic storage for a database, you must explicitly specify the **`AUTOMATIC STORAGE NO`** clause on the **`CREATE DATABASE`** command.

Any new table spaces created without explicitly providing containers definitions are also created in these two storage paths. For example:

```
CREATE TABLESPACE TBSP1
```

This creates a table space with two containers, one on each of the two storage paths (`/path1` and `/path2`). As the database grows, the database manager automatically extends the size of containers across all the storage paths.

Furthermore, automatic storage allows us to add a new storage space if we run out of space in the existing storage paths. For example, suppose we want to add extra storage space to the database from the previous example, and this storage space is in a new storage path, say `/path3`. We use the **ALTER DATABASE** statement as follows:

```
ALTER DATABASE TESTDB ADD STORAGE ON /path3
```

The new storage path is not used until there is no more room to grow within the containers on the existing storage paths (`/path1` and `/path2`). To use the new storage path immediately, issue an **ALTER TABLESPACE** statement to rebalance all the database table spaces resided in the existing storage paths. For instance:

```
ALTER TABLESPACE TBSP1 REBALANCE
```

The rebalance process runs asychronously in the background and does not affect the availability of data. However, rebalance is an expensive operation and has significant IO overhead. Therefore, it is important to gauge the performance impact on the database system when rebalancing is being performed.

One suggestion is to start with rebalancing a relatively small table space and use the result as a reference to estimate how long it takes to rebalance the other table spaces, and what the performance impact is. Then, decide the time of day that is suitable for performing the rebalance of larger table spaces.

Alternatively, we can use the throttling utility in DB2 to limit the performance impact of rebalancing on the system. The database manager configuration parameter util_impact_lim sets the limit on the impact of which all throttled utilities can have on the overall workload of the system. By default, this parameter is set to 10, which means all throttled utilities combined can have no more than a 10% average impact upon the workload as judged by the throttling algorithm. A value of 100 indicates no throttling. The **SET UTIL_IMPACT_PRIORITY** command is used to set the priority that a particular utility has over the resources available to throttled utilities as defined by the util_impact_lim.

For our rebalancing, we use the **`SET UTIL_IMPACT_PRIORITY`** command to set the priority of the rebalancing process so that the impact of rebalancing is controlled. Here are the steps:

1. Run **`db2 list utilities show detail`** to obtain the utility ID of the rebalancing process. This command shows the current progress of the rebalancing process in terms of estimated percentage complete.
2. Set the priority of this process by executing the **`SET UTIL_IMPACT_PRIRITY FOR <utility_id> TO <priority>`** command. If rebalance still incurs too much overhead or takes an excessive amount of time to complete, an offline approach (namely backup/redirected restore) can be used. The idea is to back up the existing database, restore the database with REDIRECT option, add the new storage path, and continue the restore.

   Similar to adding a new storage path, automatic storage allows us to remove existing storage paths from a database or move the data off the storage paths and rebalance them. For more details about how this can be done, see the DB2 v9.7 Information Center at the following Web page:

   `http://publib.boulder.ibm.com/infocenter/db2luw/v9r7/index.jsp?topic=/com.ibm.db2.luw.doc/welcome.html`

   An important consideration for storage paths or file systems used for automatic storage is that the file systems are uniform in capacity and exhibit similar I/O characteristics. Figure 3-3 on page 91 illustrates how containers of a table space grow in the storage paths in which capacity is uneven.

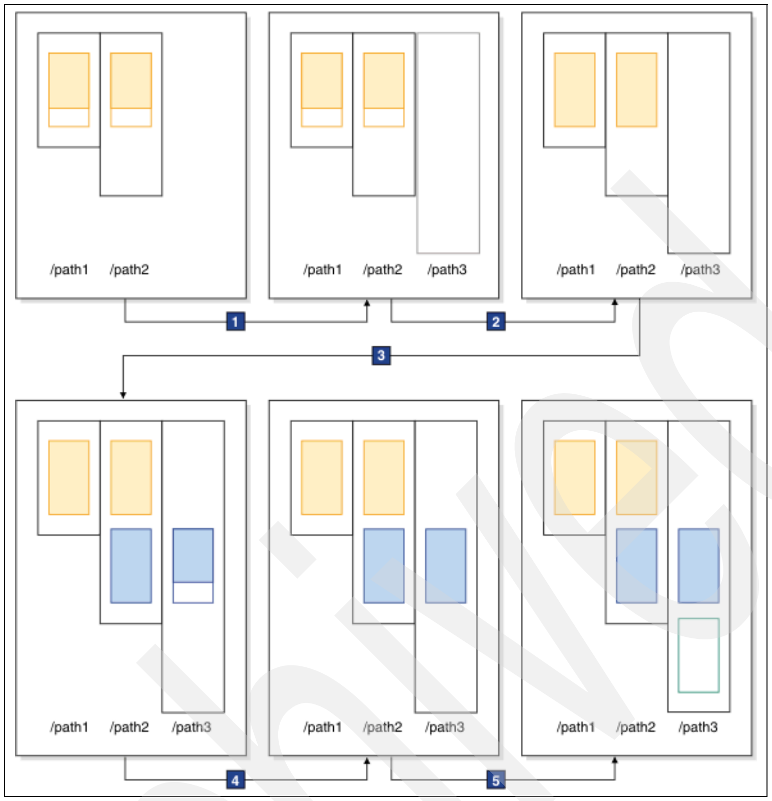


Figure 3-3 How does a table space grow?

3. The table space starts out with two containers (`/path1` and `/path2`) that have not yet reached maximum capacity. A new storage path (`/path3`) is added to the database using the **`ALTER DATABASE`** statement. This new storage path is not yet being used without initiating a rebalance process by **`ALTER TABLESPACE`**.
4. The original containers in `/path1` and `/path2` reach the maximum capacity.
5. Because there is no storage space left in `/path1`, a new stripe set of containers are added to `/path2` and `/path3` and they start to fill up with data.

> **Note:** *Stripe* is defined as the total of all the segments for one pass of all the back-end data disks.

6. The containers in the new stripe set (in `/path2` and `/path3`) reach their maximum capacity.
7. A new stripe set is added only to `/path3` because there is no room for the container in `/path2` to grow.

As observed, we have more data in `/path2` after step 4. This immediately creates a performance hot spot. Hot space can also be introduced if the I/O performance in a storage path is significantly slower than the others. Furthermore, the example shows that without rebalance the new storage path is not used until an existing storage path runs out of space.

The recommended practice is to have all storage paths with the same capacity and I/O characteristics, and rebalance after a new storage path is added. Figure 3-4 depicts the scenario that follows this practice. With rebalance and uniform capacity, the table space grows evenly across all the storage paths. This ensures that parallelism remains uniform and achieves the optimal I/O performance.

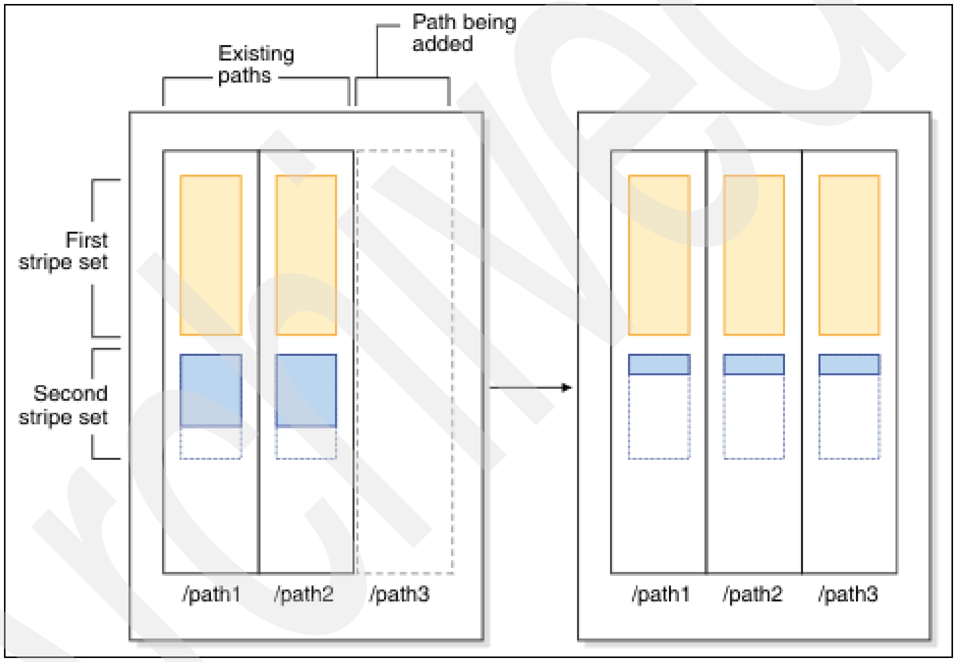


Figure 3-4 Storage grows with uniform capacity storage paths and rebalance

3.2.2 Tablespace design

As mentioned, with automatic storage we do not need to define our containers explicitly when creating table spaces. Nevertheless, automatic storage does not preclude us from defining other types of table spaces as well as their attributes.

First, what table spaces do you need? A simple design is to have one table space for temporary tables, one table space for data, and one table space for all indexes. In a partitioned environment (or DPF environment), a typical approach is to have separate table spaces for partitioned data and non-partitioned data. The next step is to consider what types of table spaces you need, and their attributes. The following sections describe suggested practices and important considerations when designing a table space.

Pagesize

Rows of table data are organized into blocks called *pages*. Pages can be four sizes: 4, 8, 16, and 32 KB. Table data pages do not contain the data for columns defined with LONG VARCHAR, LONG VARGRAPHIC, BLOB, CLOB, DCLOB, or XML data types, unless the LOB or XML document is inlined through the use of INLINE LENGTH option of the column. The rows in a data page, however, contain a descriptor of these columns.

With a different pagesize, the maximum row length can vary, as shown in Table 3-1.

Table 3-1 Page size versus row length:

pagesize	row length
4 KB	4005 bytes
8 KB	8101 bytes
16 KB	16 293 bytes
32 KB	32 677 bytes

Regardless of the page size, the maximum number of rows in a data page is 255 for tables in a REGULAR table space. This number can go higher with a LARGE table space.

For an OLTP environment, a smaller page size is usually preferable, because it consumes less buffer pool space with unwanted rows. 8 KB is recommended. DW workload, on the other hand, accesses a large number of consecutive rows at a time. Thus, a larger page size is usually better because it reduces the number of I/O requests that are required to read a specific number of rows. The recommended value for pagesize is 16 KB. There is, however, an exception to this. If your row length is smaller than the page size divided by the maximum number of rows, there is consumed space on each page. In this situation, a smaller page size might be more appropriate.

LARGE versus REGULAR

For permanent data (such as table data and indexes), we can define our table spaces of type REGULAR or LARGE.

A REGULAR table space allows tables to have up to 255 rows per data page, and can be managed by the database manager (DMS) or system (SMS).

A table in a LARGE table space can support more than 255 rows per data page, which can improve space use on data pages. Indexes on the table are slightly larger because LARGE table spaces make use of large row identifiers (RIDs), which are 2 bytes longer than regular RIDs. In terms of performance, there is not much difference between REGULAR and LARGE table spaces. By default, DMS table spaces are created as LARGE table spaces. For SMS table spaces, however, the only supported table space type is REGULAR table spaces.

DMS versus SMS and automatic storage

A table space can be managed by the DMS, the SMS, or automatic storage. For more information about this, see the following Information Center article:

http://publib.boulder.ibm.com/infocenter/db2luw/v9r7/index.jsp?topic=/com.ibm.db2.luw.admin.dbobj.doc/doc/c0055446.html

Of the three types of table spaces, automatic storage table spaces are the easiest to set up and maintain, and are recommended for most applications. They are particularly beneficial in the following situations:

- ► You have larger tables or tables that are likely to grow quickly.
- ► You do not want to make regular decisions about how to manage container growth.
- ► You want to store other types of related objects (for example, tables, LOBs, indexes) in other table spaces to enhance performance.

DMS table spaces are useful in the following circumstances:

- ► You have larger tables or tables that are likely to grow quickly.
- ► You want to exercise greater control over where data is physically stored.
- ► You want to make adjustments to or control how storage is used (for example, adding containers)
- ► You want to store other types of related objects (for example, tables, LOBs, indexes) in other table spaces to enhance performance.

SMS table spaces are useful in the following situations:

- ► You have smaller tables that are not likely to grow quickly.
- ► You want to exercise greater control over where data is physically stored.
- ► You want to do little in the way of container maintenance.
- ► You are not required to store other types of related objects (for example, tables, LOBs, indexes) in other table spaces. (For partitioned tables only, indexes can be stored in table spaces separately from table data).

In general, automatic storage table spaces and DMS table spaces have better performance than SMS table spaces.

A DMS table space has a choice of creating its containers as a raw device or a file in a file system. A raw device traditionally provides better performance. Nonetheless, the concurrent I/O (CIO) feature on AIX now has almost completely eliminated the need to use a raw device for performance. (CIO is discussed in “File system caching” on page 97.) File systems also provides superior manageability as compared to raw devices. Therefore, the general recommendation is to use file systems instead of raw devices.

EXTENSIZE

The EXTENTSIZE for a table space specifies the number of pages that is written to a container before skipping to the next container (if there are more than one container in the table space).

To maximize I/O performance, the EXTENTSIZE is set to the number of pages that includes an entire RAID stripe. For example, on a system with a 128 KB segment size that uses RAID5 7+P, the stripe size is 896 KB (128 KB x 7). Note that when calculating the RAID 5 stripe size, the Parity Disk is excluded from the calculation.

So in this example, if the page size used is 16 KB, the extensize in pages is 896/16 = 56. The concept of segment and RAID stripe is discussed further in 3.3.3, “Storage structure: RAID levels” on page 104 and 3.3.4, “Logical drives (LUN) and controller ownership” on page 114.

We need to take into consideration if we have many tables in which the amount of data is far less than the EXTENTSIZE. This is not uncommon in the OLTP environments. Consider an example where we have 40,000 tables, of which almost 20,000 tables are empty. Because the EXTENTSIZE is the minimum allocation for tables, a lot of space is wasted. In such environments, a smaller EXTENSIZE might be preferable. For instance, you might consider the EXTENTSIZE to be equal to the segment size.

PREFETCHSIZE

The PREFETCHSIZE specifies the number of pages to read by a query prior to the pages being referenced by the query, so that the query does not need to wait for IO. For example, suppose you have a table space with three containers. If you set the PREFETCHSIZE to be three times the EXTENTSIZE, the database manager can do a big-block read from each container in parallel, thereby significantly increasing I/O throughput. This assumes that the best practice is followed in such a way that each container is resided on a separate physical device.

The recommendation is to leave it as AUTOMATIC. This way, DB2 updates the prefetchsize automatically when there is a change in the number of containers in a table space. The calculation of the PREFETCHSIZE is as follows:

```
number_of_containers × number_of_disks_per_container × extent_size
```

For example, assume the extent size for a database is eight pages, and that there are four containers, each of which exists on a single physical disk. Setting the prefetch size to: 4 × 1 × 8 = 32 results in a prefetch size of 32 pages in total. These 32 pages are read from each of the four containers in parallel.

The default value for number_of_disks_per_container is 1. Nonetheless, this value can be adjusted by the DB2 registry variable DB2_PARALLEL_IO.

OVERHEAD and TRANSFERRATE

When creating a table space, we can define the OVERHEAD and TRANSFERRATE associated the containers in the table space.

- OVERHEAD provides an estimate of the time (in milliseconds) that is required by the container before any data is read into memory.
- TRANSFERRATE provides an estimate of the time (in milliseconds) that is required to read one page of data into memory.

These two values affect how DB2 optimizer selects the optimal access plans for queries. You can use the formula in Example 3-1 to estimate the values for OVERHEAD and TRANSFERRATE.

Example 3-1 Estimate the OVERHEAD and TRANSFERRATE

```
OVERHEAD = average seek time in milliseconds + (0.5 * rotational
latency)
where:
- 0.5 represents the average overhead of one half rotation
- rotational latency (in milliseconds) is calculated for each full
rotation: (1 / RPM) * 60 * 1000 where RPM represents rotation per
minute.

For example, assume that a disk performs 7200 RPM. Using the
rotational-latency formula:
   (1 / 7200) * 60 * 1000 = 8.328 milliseconds
This value can be used to estimate the overhead as follows, assuming an
average seek time of 11 milliseconds:
   OVERHEAD = 11 + (0.5 * 8.328)
            = 15.164
```

```
As to TRANSFERRATE, if each tablespace container is a single physical
disk, you can use the following formula to estimate the transfer cost
in milliseconds per page:

TRANSFERRATE = (1 / spec_rate) * 1000 / 1024000 * pagesize
where:
- spec_rate: represents the disk specification for the transfer rate
(in megabytes per second)

For example, suppose the specification rate for a disk is 3 megabytes
per second. Then:
   TRANSFERRATE = (1 / 3) * 1000 / 1024000 * 4096
                = 1.333248
or about 1.3 milliseconds per page.

If the table space containers are not single physical disks, but are
arrays of disks (such as RAID), you must take additional considerations
into account when estimating the TRANSFERRATE.

If the array is relatively small, you can multiply the spec_rate by the
number of disks, assuming that the bottleneck is at the disk level.
However, if the array is large, the bottleneck might not be at the disk
level, but at one of the other I/O subsystem components, such as disk
controllers, I/O busses, or the system bus. In this case, you cannot
assume that the I/O throughput capacity is the product of the spec_rate
and the number of disks. Instead, you must measure the actual I/O rate
(in megabytes) during a sequential scan. For example, a sequential scan
resulting from select count(*) from big_table could be several
megabytes in size. In this case, divide the result by the number of
containers that make up the table space in which BIG_TABLE resides. Use
this result as a substitute for spec_rate in the formula given above.
```

Containers assigned to a table space can reside on other physical disks. For the best results, all physical disks used for a given table space must have the same OVERHEAD and TRANSFERRATE characteristics. If these characteristics are not the same, use average values when setting OVERHEAD and TRANSFERRATE.

File system caching

The operating system, by default, caches file data that is read from and written to disk. For example, a typical read operation involves reading the data from a disk into the file system cache, and then copying the data from the cache to the application buffer (in our case, DB2 bufferpool). A similar process (but in an opposite direction) is applied to a write operation. In a few cases, this double buffering effect (for example, caching at both file system cache and DB2 bufferpool levels) causes performance degradation, because extra CPU cycles need to be incurred. Because we waste memory to buffer the same page twice, the potential paging can further negatively affect the DB2 application performance.

AIX provides a feature to disable file system caching and avoid double caching. This feature is known as Concurrent I/O (CIO). Another advantage of CIO is that this feature might help reduce the memory requirements of the file system cache, making more memory available for other uses. In DB2, we can enable CIO on table spaces by specifying the clause NO FILE SYSTEM CACHING in the CREATE TABLESPACE or ALTER TABLESPACE statement. (Starting from DB2 v9.5, NO FILE SYSTEM CACHING is the default for any DMS table space created.)

LOBS (Large Objects) are not cached in DB2 bufferpool. For table spaces that contain a fair amount of LOBS data, performance might be improved by enabling file system caching.

3.2.3 Other considerations for DW environments

There are two more considerations for DW environments to discuss, namely placing the tables into table spaces and compressing the data.

How to place tables into table spaces

In a DPF environment (especially for DW workloads), refer to the following guidelines when deciding how to place tables into table spaces:

- Large tables

  It is recommended that you place these tables in a separate table space across all data partitions. These tables might benefit from the use of a separate multi-partition table space for the indexes associated with the tables. Range-partitioned tables might benefit from using one table space per range.

- Medium-sized tables

  These tables are logically grouped together and placed into table spaces with five or more tables in each table space. These table spaces are placed across all data partitions. Tables that are placed in the same table space have a logical relationship to one another (such as all coming from the same department or requiring the same backup time frame). The indexes for these tables can be placed within the data table space or separated into another shared table space for indexes.

- Small tables

  Many tables are placed into one single-partition table space typically on the catalog partition. The indexes associated with these table spaces generally reside within the data table space.

- Replicated tables

  The single-partition tables that are used for joins with multi-partition tables can be replicated across all data partitions. These replicated tables are placed into a separate table space.

- Materialized query tables (MQTs) and other database objects

  These type of objects are typically placed in separate table spaces that are placed across all data partitions.

Compression

Data compression provides several benefits. The most significant benefit is lowered overall space requirements. Typical compression ratios for data tables are between 50% and 65%, although the compression ratio can be much higher or lower depending on the data. Another benefit of data compression is the potential for improved performance. Processing compressed data uses more processor cycles than uncompressed data, but requires less I/O.

Row compression, a feature available since DB2 V9.1, allows data tables to be compressed. In DB2 V9.7, two new compression techniques are introduced:

- Index compression

  Index compression compresses indexes (including indexes on declared or created temporary tables). If a data table is compressed, new indexes created on this table are automatically compressed. You can always enable or disable index compression on any indexes explicitly.

- Temporary table compression.

  Temporary table compression, on the other hand, compresses temporary tables, such as created global temporary tables (CGTT) and declared global temporary tables (DGTT). Unlike row or index compression, temporary table compression is always on.

For more details about how row and index compression can be used, see the DB2 Information Center at the following Web page:

http://publib.boulder.ibm.com/infocenter/db2luw/v9r7/index.jsp?topic=/com.ibm.db2.luw.doc/welcome.html

Important: You must apply for the IBM DB2 Storage Optimization Feature license to use any of the compression features.

3.3 Storage hardware

In the previous section, we described general considerations in DB2 storage design. In this section, we look at storage hardware and discuss how storage hardware can be configured and tuned to meet database storage requirements. For illustrative purposes, we focus on the IBM System Storage™ DS5000 series.

3.3.1 Introduction to the IBM Storage DS5000 Storage Server

IBM has brought together a broad selection of storage servers into one family, known as the DS family, to help small to large enterprises select the right solutions for their needs. The DS family combines the high-performance IBM System Storage DS8000 series of enterprise servers with the IBM System Storage DS4000/5000 series of mid-range systems, and other line-of-entry systems, namely IBM System Storage DS3000 series.

For the mid-range systems, the DS5000 series, in particular, provides an increase in performance by using the latest technology and increased storage capacity of the high-end DS4000® series. Our Infosphere warehouse solution also makes use of the DS5000 series together with DB2 to provide data warehouse solution for large enterprise.

The DS5000 series of storage servers uses Redundant Array of Independent Disks (RAID) technology. RAID technology is used to protect the user data from disk drive failures. DS5000 storage servers contain Fibre Channel (FC) interfaces to connect both the host systems and disk drive enclosures. Its host ports support existing Gbps FC SANs (storage area networks), as infrastructure changes ports can be changed by adding, replacing, or mixing host interfaces to support 8 Gbps FC and 10 Gbps iSCSI.

The DS5000 Storage Manager software, which comes with the hardware, can be used to configure, manage, and troubleshoot the DS5000 storage servers. We use this software to configure RAID arrays and logical drives, assign logical drives to hosts, replace and rebuild failed disk drives, expand the size of the arrays and logical drives, and convert from one RAID level to another.

In the next sections, we talk about important aspects of the DS5000 series that deserve special attention. More importantly, a few of them have an impact on performance of our database systems. For other details on the DS5000 series, readers are encouraged to refer to IBM RedBooks publication *Introduction to the IBM System Storage DS5000 Series*, SG24-7676.

3.3.2 Physical components considerations

In this section we discuss the considerations for physical components.

Cables and connectors

In FC technology, frames are moved from source to destination using gigabit transport, which is a requirement to achieve fast transfer rates. To communicate with gigabit transport, both sides have to support this type of communication.

The FC standard specifies a procedure for speedy auto-detection. Therefore, if a 2 Gbps port on a switch or device is connected to a 1 Gbps port, it must negotiate down and the link runs at 1 Gbps. If there are two 2 Gbps ports on either end of a link, the negotiation runs the link at 2 Gbps if the link is up to specifications.

A link that is too long can end up running at 1 Gbps, even with 2 Gbps ports at either end, so be aware of your distances and make sure your fiber is good. The same rules apply to 4 Gbps devices relative to 1 Gbps and 2 Gbps environments. The 4 Gbps devices have the ability to negotiate back down to either 2 Gbps or 1 Gbps, depending upon the attached device and link quality.

Host Based Adapter

There are eight (DS5100 system) to sixteen (DS5300 system) host connections. Each connection can operate at 4 Gbps, but also auto-negotiates to support 2 Gbps and 1 Gbps connections, as described previously. Each host is associated with a Host Based Adapter (HBA). Having sixteen independent host ports allows us to establish fully redundant direct connections to up to eight hosts. For example,

Figure 3-5 on page 102 shows that the host ports are connected to eight sets of FC HBA.

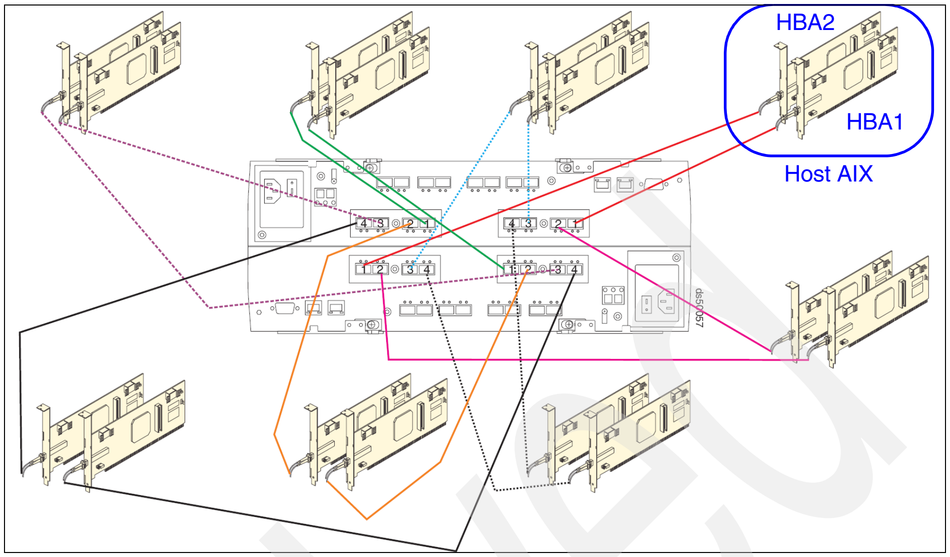


Figure 3-5 Directly connected hosts to the DS5000 storage server

When multiple HBAs are installed on a host, multipath driver might be used to achieve redundancy (failover) or load sharing. We revisit this topic later in this section.

Drives

The speed and the type of the drives[2] used impacts the performance. Typically, the faster the drive, the higher the performance. This increase in performance comes at a cost. The faster drives typically cost more than the lower performance drives. FC drives outperform the SATA drives. In particular, the DS5000 storage server supports the following types of FC drives:

- 4 Gbps FC, 146.8 GB / 15K Enhanced Disk Drive Module
- 4 Gbps FC, 300 GB / 15K Enhanced Disk Drive Module
- 4 Gbps FC, 450 GB / 15K Enhanced Disk Drive Module

Note that a FC 15K drive rotates 15,000 times per minute. Also, its read and write bandwidth are 76 Mbps and 71 Mbps, respectively.

The speed of the drive is the number or revolutions per minute (RPM). A 15 K drive rotates 15,000 times per minute. With higher speeds, the drives tend to be denser, as a large diameter plate driving at such speeds is likely to wobble. With the faster speeds comes the ability to have greater throughput.

[2] The words “drive” and “disk” are used interchangeably in this chapter.

Seek time is the measure of how long it takes for the drive head to move to the correct sectors on the drive to either read or write data. It is measured in milliseconds (ms). The faster the seek time, the quicker data can be read from or written to the drive. The average seek time reduces when the speed of the drive increases. Typically, a 7.2 K drive has an average seek time of around 9 ms, a 10 K drive has an average seek time of around 5.5 ms, and a 15 K drive has an average seek time of around 3.5 ms. Together with RPM, the seek time is used to determined the value of OVERHEAD for our DB2 table spaces (see 3.2.2, “Tablespace design” on page 92).

Command queuing allows for multiple commands to be outstanding to the disk drive at the same time. The drives have a queue where outstanding commands can be dynamically rescheduled or re-ordered, along with the necessary tracking mechanisms for outstanding and completed portions of workload. The FC disks currently have a command queue depth of 16.

Hot spare drive

A hot spare drive is like a replacement drive installed in advance. Hot spare disk drives provide additional protection that might prove to be essential in case of a disk drive failure in a fault tolerant array.

There is no definitive recommendation as to how many hot spares you must install, but it is common practice to use a ratio of one hot spare for 28 drives.

We recommend that you also split the hot spares so that they are not on the same drive loops (see Figure 3-6 on page 104).

Tip: When assigning disks as hot spares, make sure they have enough storage capacity. If the failed disk drive is larger than the hot spare, reconstruction is not possible. Ensure that you have at least one of each size or all larger drives configured as hot spares.

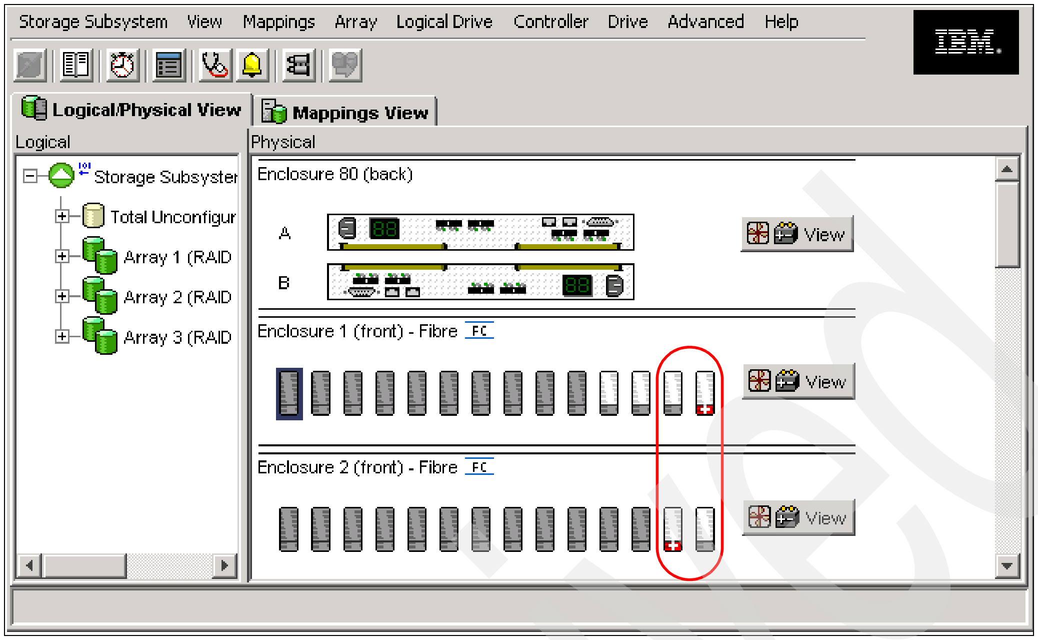


Figure 3-6 Hot spare coverage with alternating loops

3.3.3 Storage structure: RAID levels

An array is a set of drives that the system logically groups together to provide one or more logical drives to an application host or cluster. RAID is the technology that we use to form an array. When defining arrays, you often have to compromise among capacity, performance, and redundancy.

I/O characteristics between OLTP and DW workloads

To better understand what it is meant by the performance impact of other RAID levels, we first need to understand the I/O characteristics that OLTP and DW workloads or applications exhibit.

OLTP (Online Transaction Processing)

OLTP workloads, often described as transaction-based environments, are typically I/O intensive. They require a high number of I/O per seconds (IOPS). Their I/O operations are geared towards random reads and writes, and use a small random data block pattern to transfer data. The latter benefits by having more back-end drives, because a large number of back-end drives enable more host I/Os to be processed simultaneously, as read cache is far less effective, and the misses need to be retrieved from disk.

DW (Data Warehouse)

DW workloads typically require massive amounts of data sent and generally use large sequential blocks to reduce disk latency. This environment is also known as a throughput-based environment. Read operations make use of the cache to stage greater chunks of data at a time, to improve the overall performance. Throughput rates are heavily dependent on the storage server's internal bandwidth.

RAID levels

In this section, we discuss the RAID levels and explain why we choose a particular setting in a particular situation. You can draw your own conclusions.

RAID 0: For performance, but generally not recommended

RAID 0 (Figure 3-7) is also known as data striping. It is well-suited for program libraries requiring rapid loading of large tables, or more generally, applications requiring fast access to read-only data or fast writing. RAID 0 is designed to increase performance. There is no redundancy, so any disk failures require reloading from backups. Select RAID 0 for applications that benefit from the increased performance capabilities of this RAID level. Never use this level for critical applications that require high availability.

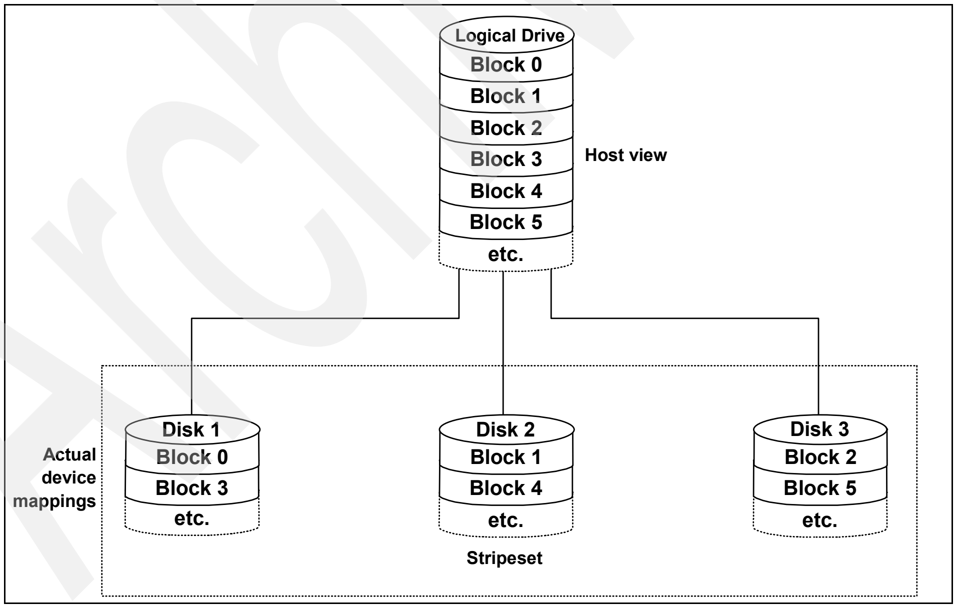


Figure 3-7 RAID 0

RAID 1: For availability/good read response time

RAID 1 (Figure 3-8) is also known as disk mirroring. It is most suited to applications that require high data availability, good read response times, and where cost is a secondary issue. The response time for writes can be somewhat slower than for a single disk, depending on the write policy. The writes can either be executed in parallel for speed or serially for safety. Select RAID 1 for applications with a high percentage of read operations and where the cost is not the major concern.

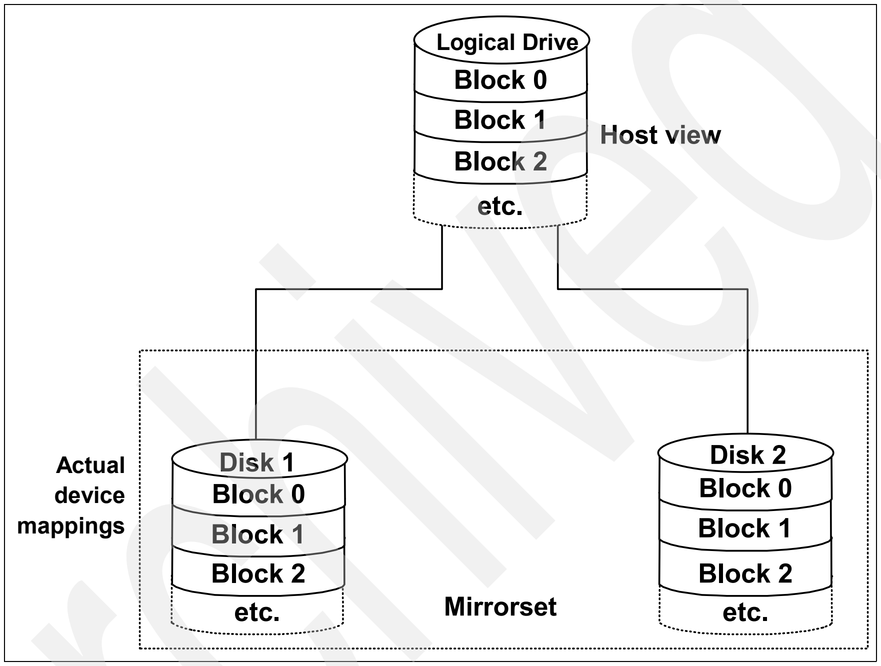


Figure 3-8 RAID 1

Because the data is mirrored, the capacity of the logical drive when assigned RAID 1 is 50% of the array capacity.

The following list details recommendations when using RAID 1:

- Use RAID 1 for the disks that contain your operating system. It is a good choice, because the operating system can usually fit on one disk.
- Use RAID 1 for transaction logs. Typically, the database server transaction log can fit on one disk drive. In addition, the transaction log performs mostly sequential writes. Only rollback operations cause reads from the transaction logs. Therefore, we can achieve a high rate of performance by isolating the transaction log on its own RAID 1 array.

- Use write caching on RAID 1 arrays. Because a RAID 1 write does not complete until both writes have been done (two disks), performance of writes can be improved through the use of a write cache. When using a write cache, be sure it is battery-backed up.

Note: RAID 1 is actually implemented only as RAID 10 (see "RAID 10: Higher performance than RAID 1" on page 109) on the DS5000 storage server products.

RAID 5: High availability and fewer writes than reads

RAID 5 (Figure 3-9) stripes data and parity across all drives in the array. RAID 5 offers both data protection and increased throughput. When you assign RAID 5 to an array, the capacity of the array is reduced by the capacity of one drive (for data-parity storage). RAID 5 gives you higher capacity than RAID 1, but RAID level 1 offers better performance.

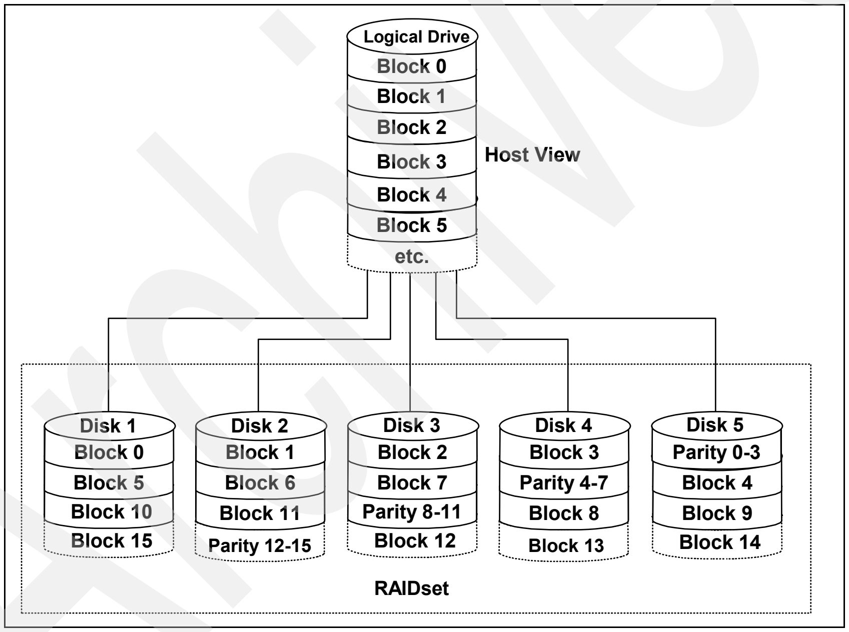


Figure 3-9 RAID 5

RAID 5 is best used in environments requiring high availability and fewer writes than reads.

RAID 5 is good for multi-user environments, such as database or file system storage, where typical I/O size is small, and there is a high proportion of read activity. Applications with a low read percentage (write-intensive) do not perform as well on RAID 5 logical drives because of the way a controller writes data and redundancy data to the drives in a RAID 5 array. If there is a low percentage of read activity relative to write activity, consider changing the RAID level of an array for faster performance.

Use write caching on RAID 5 arrays, because RAID 5 writes are not completed until at least two reads and two writes have occurred. The response time of writes is improved through the use of write cache (be sure it is battery-backed up). RAID 5 arrays with caching can give as good as performance as any other RAID level, and with a few workloads, the striping effect gives better performance than RAID 1.

RAID 6: High availability with additional fault tolerance

RAID 6 (Figure 3-10) is a RAID level employing n+2 drives, which can survive the failure of any two drives. RAID 6 stripes blocks of data and parity across an array of drives. It calculates two sets of information for each block of data (p+q). For the purposes of RAID 6 p+q, they can be used to generate up to two missing values from a set of data. The key to this method is the q, which is a codeword based upon Reed-Solomon error correction. As such, q is more like a CRC than parity. The calculation of q is complex. In the case of the DS5000 storage server, this calculation is made by the hardware.

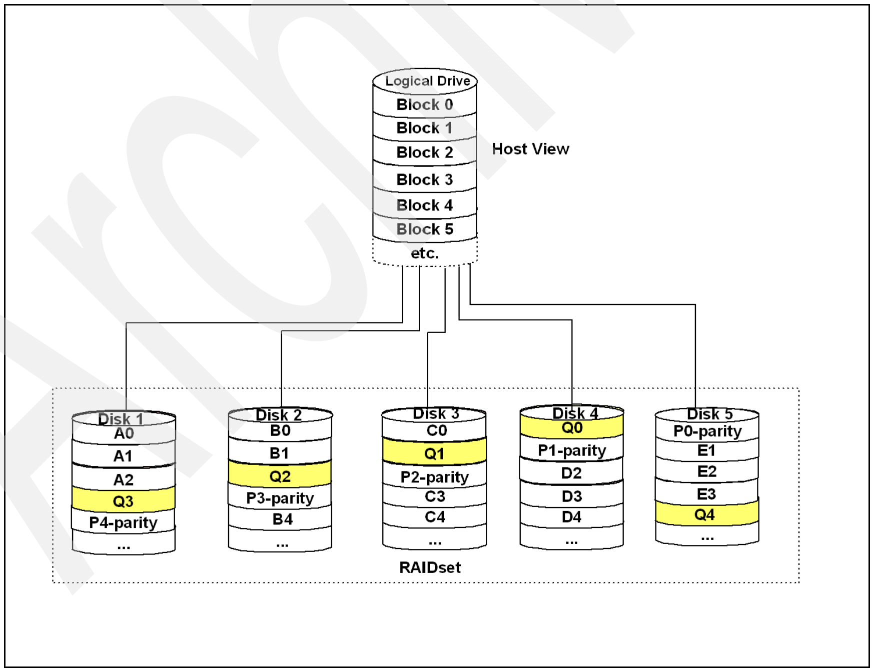


Figure 3-10 RAID 6

By storing two sets of distributed parities, RAID 6 is designed to tolerate two simultaneous disk failures. Furthermore, unlike RAID 5, RAID 6 is also protected from data loss during array re-construction.

Due to the added impact of more parity calculations, RAID 6 is slightly slower than RAID 5 in terms of writing data. Nevertheless, there is essentially no impact on read performance when comparing between RAID 5 and RAID 6 (provided that the number of disks in the array is equal).

RAID 10: Higher performance than RAID 1

RAID 10 (refer to Figure 3-11), also known as RAID 1+0, implements block interleave data striping and mirroring. In RAID 10, data is striped across multiple disk drives. Those drives are then mirrored to another set of drives.

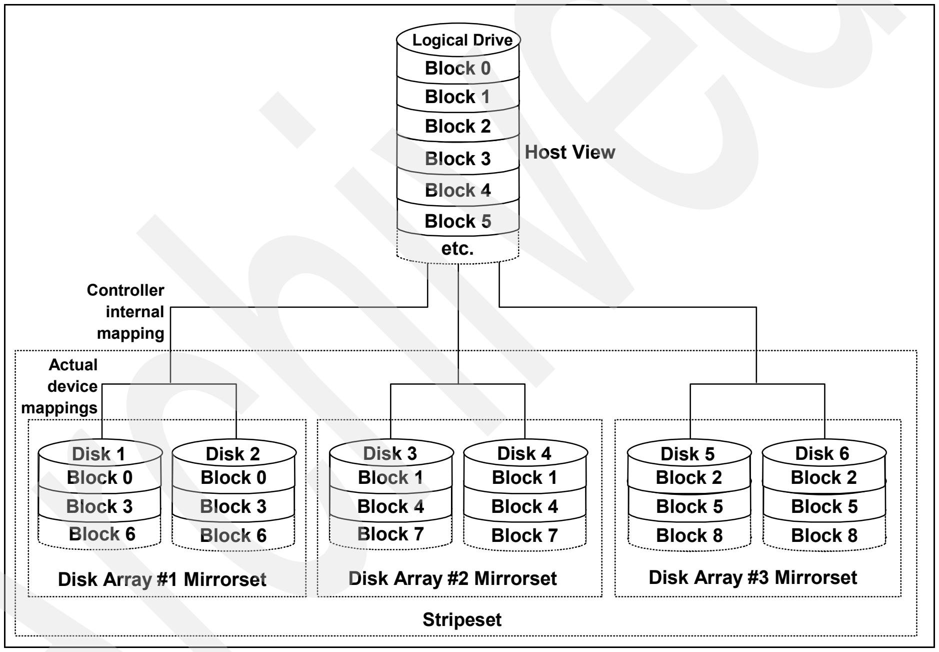


Figure 3-11 RAID 10

The performance of RAID 10 is approximately the same as RAID 0 for sequential I/Os. RAID 10 provides an enhanced feature for disk mirroring that stripes data and copies the data across all the drives of the array. The first stripe is the data stripe. The second stripe is the mirror (copy) of the first data stripe, but it is shifted over one drive. Because the data is mirrored, the capacity of the logical drive is 50% of the physical capacity of the hard disk drives in the array.

The recommendations for using RAID 10 are as follows:

- Use RAID 10 whenever the array experiences more than 10% writes. RAID 5 does not perform as well as RAID 10 with a large number of writes.
- Use RAID 10 when performance is critical. Use write caching on RAID 10. Because a RAID 10 write is not completed until both writes have been done, write performance can be improved through the use of a write cache (be sure it is battery-backed up).

When comparing RAID 10 to RAID 5:

- RAID 10 writes a single block through two writes. RAID 5 requires two reads (read original data and parity) and two writes. Random writes are significantly faster on RAID 10.
- RAID 10 rebuilds take less time than RAID 5 rebuilds. If a real disk fails, RAID 10 rebuilds it by copying all the data on the mirrored disk to a spare. RAID 5 rebuilds a failed disk by merging the contents of the surviving disks in an array and writing the result to a spare.
- RAID 10 is the best fault-tolerant solution in terms of protection and performance, but it comes at a cost. You must purchase twice the number of disks that are necessary with RAID 0.

A comparison of RAID levels

Table 3-2 compares the RAID levels.

Table 3-2 RAID levels comparison

RAID	Description	Workload	Advantage	Disadvantage
0	Stripes data across multiple drives.	OLTP DW	Performance, due to parallel operation of the access.	No redundancy. If one drive fails, the data is lost.
1	The disk's data is mirrored to another drive.	OLTP	Performance, as multiple requests can be fulfilled simultaneously.	Storage costs are doubled.
10	Data is striped across multiple drives and mirrored to the same number of disks.	OLTP	Performance, as multiple requests can be fulfilled simultaneously. Most reliable RAID level on the DS5000 storage server.	Storage costs are doubled.
5	Drives operate independently with data and parity blocks distributed across all drives in the group.	OLTP DW	Good for reads, small IOPS, many concurrent IOPS, and random I/Os.	Writes are particularly demanding.

RAID	Description	Workload	Advantage	Disadvantage
6	Stripes blocks of data and parity across an array of drives and calculates two sets of parity information for each block of data.	OLTP DW	Good for multi-user environments, such as database, where typical I/O size is small, and in situations where additional fault tolerance is required.	Slower in writing data, complex RAID controller architecture.

Array configuration

Before you can start using the physical disk space, you must configure it. Based on the previous recommendation in RAID levels, you divide your (physical) drives into arrays accordingly and create one or more logical drives inside each array.

In simple configurations, you can use all of your drive capacity with one array and create all of your logical drives in that unique array. However, this presents the following drawbacks:

- If you experience a (physical) drive failure, the rebuild process affects all logical drives, and the overall system performance goes down.
- Read/write operations to logical drives are still being made to the same set of physical hard drives.

The array configuration is crucial to performance. You must take into account all the logical drives inside the array, as all the logical drives inside the array impact the same physical disks. If you have two logical drives inside an array and they both are high throughput, there might be contention for access to the physical drives as large read or write requests are serviced. It is crucial to know the type of data that each logical drive is used for and try to balance the load so contention for the physical drives is minimized. Contention is impossible to eliminate unless the array only contains one logical drive.

Number of drives

The more physical drives you have per array, the shorter the access time for read and write I/O operations.

You can determine how many physical drives is associated with a RAID controller by looking at disk transfer rates (rather than at the megabytes per second). For example, if a disk drive is capable of 75 nonsequential (random) I/Os per second, about 26 disk drives working together can, theoretically, produce 2000 nonsequential I/Os per second, or enough to hit the maximum I/O handling capacity of a single RAID controller. If the disk drive can sustain 150 sequential I/Os per second, it takes only about 13 disk drives working together to produce the same 2000 sequential I/Os per second and keep the RAID controller running at maximum throughput.

Tip: Having more physical drives for the same overall capacity gives you:

- Performance

  By doubling the number of the physical drives, you can expect up to a 50% increase in throughput performance.

- Flexibility

  Using more physical drives gives you more flexibility to build arrays and logical drives according to your needs.

- Data capacity

  When using RAID 5 logical drives, more data space is available with smaller physical drives because less space (capacity of a drive) is used for parity.

Enclosure loss protection

Enclosure loss protection is a good way to make your system more resilient against hardware failures. Enclosure loss protection means that you spread your protection arrays across multiple enclosures rather than in one enclosure so that a failure of a single enclosure does not take a whole array offline.

Figure 3-12 on page 113 shows an example of the enclosure loss protection. If enclosure number 2 were to fail, the array with the enclosure loss protection still functions (in a degraded state), as the other drives are not affected by the failure.

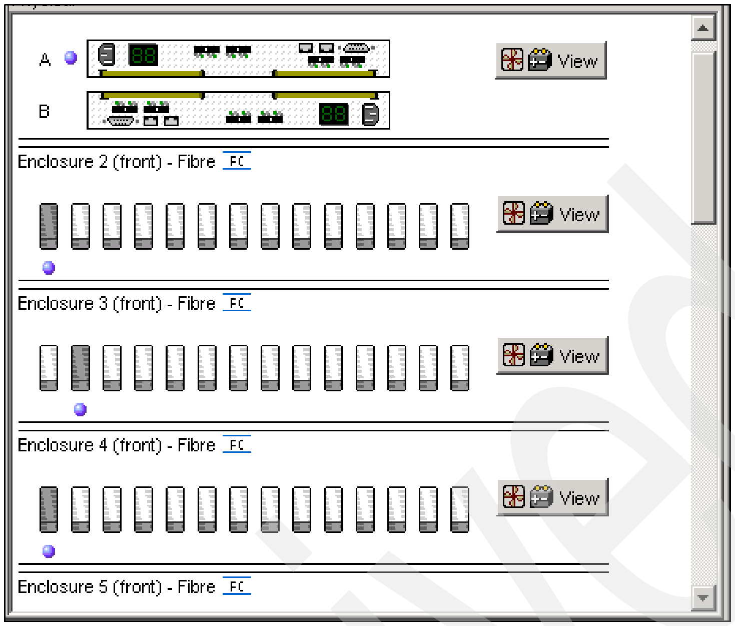

A
B
View
Enclosure 2 (front) - Fibre FC
View
Enclosure 3 (front) - Fibre FC
View
Enclosure 4 (front) - Fibre FC
View
Enclosure 5 (front) - Fibre FC

Figure 3-12 Enclosure loss protection

DB2 database considerations

Further considerations on configuring an array are needed for a DB2 database.

OLTP workloads

OLTP environments contain a fairly high level of reads and a considerable amount of writes. In most cases, it has been found that laying out the tables across a number of logical drives that were created across several RAID 5 arrays of 8+1 parity disks, and configured with a segment size of 64 KB or 128 KB, is a good starting point to begin testing. This configuration, coupled with host recommendations to help avoid offset and striping conflicts, seems to provide a good performance start point to build from. Remember that high write percentages might result in a need to use RAID 10 arrays rather than the RAID 5. This is environment-specific and requires testing to determine. A rule of thumb is that if there are greater than 25–30% writes, then you might want to look at RAID 10 over RAID 5.

Suggested practice: Spread the containers across as many drives as possible, and ensure that the logical drive spread is evenly shared across the DS5000 storage server's resources. Use multiple arrays where larger containers are needed.

DW workloads

In DW environments, write activity is relatively low, as are the random I/Os. One recommendation is that for each data partition, we allocate eight drives (146 GB FC drives) and group them into one 7 + 1 RAID 5 array.

Logs and archives

The DB2 logs and archive files generally are high write workloads, and sequential in nature. We recommend that they be placed on RAID 10 logical drives.

As these are critical files to protect in case of failures, we recommend that you keep two full copies of them on separate disk arrays in the storage server. This is to protect you from the unlikely occurrence of a double disk failure, which can result in data loss. Also, as these are generally smaller files and require less space, we suggest that two separate arrays of 1+1 or 2+2 RAID 1 be used to hold the logs and the mirror pair separately.

3.3.4 Logical drives (LUN) and controller ownership

A logical drive, sometimes referred to as LUNs (LUN stands for Logical Unit Number and represents the number a host uses to access the logical drive), are defined over an array and have a defined RAID level and capacity. More specifically, a logical driver stripes across all data disks in the array, which can be equal to the entire array, or a portion of the array. The latter means that multiple logical drives are allowed to reside on the same array. However, they have to share I/Os bandwidth on the same set of disks. Our recommendation is to define one logical drive (or LUN) over an array.

Each logical drive has a preferred controller of ownership. This controller normally handles all I/O requests for this particular logical drive. In other words, each logical drive is owned by only one controller. The alternate controller takes over and handles the I/O requests in case of a failure along the I/O path. When defining logical drives, the system normally alternates ownership between the two controllers as they are defined.

As a suggested practice, configuring logical drives to spread evenly among the two controllers (in a DS5000 storage server) allows better load balancing and controller use.

The segment size is the maximum amount of data that is written or read from a disk per operation before the next disk in the array is used. For OLTP environments, we suggest that the segment size be 64 KB to 128 KB.

With DW workloads, the focus is on moving high throughput in fewer I/O, and the nature of these workloads is generally sequential in nature. As a result, you want to have larger segments (128 KB or higher) instead to get the most from each stripe. When creating a logical drive, we specify the stripe width to be the amount of segments size multiplied by the number of disks in the logical drive (or array). More importantly, stripe width is used to determine our EXTENTSIZE during the creation of our table spaces in DB2 (refer to 3.2.2, “Tablespace design” on page 92 for more details).

3.4 Tuning storage on AIX

Before starting the creation of volume groups or logical volumes, configure and tune the storage at the AIX level.

3.4.1 Multipath driver

AIX offers a multipath driver, called Multiple Path I/O (MPIO). It allows I/O load balancing and automatic path failover. With MPIO, a device can be uniquely detected through one or more physical connections, or paths. A path-control module (PCM) provides the path management functions.

An MPIO-capable device driver can control more than one type of target device. A PCM can support one or more specific devices. Therefore, one device driver can be interfaced to multiple PCMs that control the I/O across the paths to each of the target devices.

The AIX PCM has a health-check capability that can be used to do the following tasks:

- Check the paths and determine which paths are currently usable for sending I/O.
- Enable a path that was previously marked failed because of a temporary path fault (for example, when a cable to a device was removed and then reconnected).
- Check currently unused paths that are used if a failover occurred (for example, when the algorithm attribute value is failover, the health check can test the alternate paths).

MPIO is part of the AIX operating system and does not need to be installed separately. The required AIX 6.1 level is TL0 (IZ13627).

3.4.2 hdisk tuning

On AIX, each logical drive or LUN is called a physical volume (PV) and often referred to as hdisk*x* (where *x* is a unique integer on the system).

There are three parameters that are important to set for hdisks for performance:

- queue_depth
- max_transfer
- max_coalesce

queue_depth

The maximum queue depth or queue_depth is the maximum number of concurrent operations in progress on the physical device. Excess requests beyond queue_depth are held on a pending queue in the disk driver until an earlier request completes. Setting the queue_depth for every hdisk in the system to the appropriate value is important for system performance. The valid values for queue_depth are from 1 to 256.

Typically, disk vendors supply a default queue_depth value appropriate for their disks in their disk-specific ODM device support fileset. If you know this number you can use the exact number for the queue_depth calculation. If not, you can use a number between four and 16, which is the standard queue depths for drives. FC drives usually have a queue depth of 16. You also need to know how many disks you have per hdisk. For example, if you have a RAID 5 7+P configuration, you have eight disks per hdisk so you can start with a queue_depth for the hdisk of 16 * 8 = 128. You can monitor the queues using **iostat -D**. OLTP environments usually require higher queue_depth to perform better than DW environments. To set a queue_depth of 64, issue the following command:

```
#chdev -l hdiskX -a queue_depth=64 -P
```

To make the attribute changes permanent, use the **-P** option to update the AIX ODM attribute. To make this change there must either be no volume group defined on the hdisk, or the volume group that the hdisk belongs to must be varied off. You need to reboot the server to make this change take effect.

When setting this value, keep the following in mind:

- Setting queue_depth too high might result in the storage device being overwhelmed. This can result in a range of issues from higher I/O latency (lower performance) or I/Os being rejected or dropped by the device (errors in the AIX errpt and greatly reduced performance).
- Setting queue_depth too high for FC disks might overrun the maximum number of commands (**`num_cmd_elems`**) allowed outstanding by the FC driver (see 3.4.3, “Fibre Channel adapters configuration” on page 118 on how to tune **`num_cmd_elems`**).
- A combination of a large number of large I/Os for FC disks might overrun the PCI bus address space available for the FC port.
- queue_depth is not honored for disks that do not support SCSI Command Tagged Queueing (CTQ). Also, queue_depth is not honored for disks whose q_type attribute is set to none or whose SCSI INQUIRY responses indicate lack of support for CTQ.

max_transfer

The maximum transfer size (max_transfer) sets the limit for the maximum size of an individual I/O to the disk. Although applications might make larger I/O requests, those requests are broken down into multiple max_transfer-sized requests before they are handed to the disk driver.

For OLTP workloads, most I/Os are for discontiguous 4 or 8 K pages (depending on the database page size). If it is a DW workload, then most likely there are lots of scans and I/O sizes are larger. In this case a larger max_transfer means fewer round trips if the I/Os are large and contiguous. We recommend leaving the default max_transfer of 256 KB (0x40000) for OLTP workloads and set it to 1 MB (0x100000) for DW workloads. To set a max_transfer of 1 Mb issue the following command:

```
#chdev -l hdiskX -a max_transfer=0x100000 -P
```

When setting this value, keep the following in mind:

- A combination of high queue_depth and large I/Os for FC disks might overrun the PCI bus address space available for the FC port.
- The effective maximum transfer size for a disk is the minimum of the disk's max_transfer tunable and the actual maximum transfer size of the underlying adapter for all of the paths. That is, a disk cannot have a larger maximum transfer size than the underlying adapters (see max_xfer_size in 3.4.3, “Fibre Channel adapters configuration” on page 118).

max_coalesce

The max_coalesce value sets the limit for the maximum size of an individual I/O that the disk driver creates by coalescing smaller, adjacent requests. We recommend setting this value equal to max_transfer when possible. To set a max_coalesce of 1 Mb issue the following command:

```
#chdev -l hdiskX -a max_coalesce=0x100000 -P
```

3.4.3 Fibre Channel adapters configuration

There are two parameters that are important to tune for Host Based Adapters (HBA) for performance:

- num_cmd_elems
- max_xfer_size

num_cmd_elems

The num_cmd_elems value sets a limit for the maximum number of SCSI I/O requests that can be active for the FC port at one time. The actual maximum number of commands that can be issued at one time is the minimum of num_cmd_elems and the aggregate queue_depth of the devices using the port. Supported values are between 20 and 2048.

Any excess requests beyond num_cmd_elems are held on a pending queue in the FC protocol driver until an earlier request completes. The fcstat No Command Resource Count statistic tracks the number of times that a request cannot be issued due to lack of num_cmd_elems.

We recommend setting this value to its maximum of 2048 and tuning down if necessary. To do so, use the **chdev** command. For example to set this value to 2048 issue:

```
#chdev -l fcs0 -a num_cmd_elems=2048 -P
```

When setting this value keep the following in mind that a combination of a large number of large I/Os for Fibre Channel disks might overrun the PCI bus address space available for the Fibre Channel port.

max_xfer_size

The max_xfer_size value sets the limit for the maximum size of an individual I/O sent by the port. This value also influences the amount of PCI bus address space allocated for DMA mapping of I/O buffers. The default setting of 0x100000 (1 MB) causes the FC driver to request the normal amount of PCI bus address space. Any larger setting causes the FC driver to request a larger amount of PCI bus address space. There are only two PCI bus address space sizes that the FC driver requests. It does not request more and more as you increase max_xfer_size further.

Any excess requests beyond the limit of the drivers ability to map for DMA is held on a pending queue in the adapter driver until earlier requests complete. Each time a request is set aside because it cannot be mapped for DMA a PCI_DMA error is logged. The fcstat No DMA Resource Count statistic tracks the number of times that a request cannot be issued due to lack of PCI address space for DMA.

We suggest leaving the default max_xfer_size for both OLTP and DW. As an example, if you were to set this value to 2 Mb issue:

```
#chdev -l fcs0 -a max_xfer_size=0x200000 -P
```

When setting this value, keep the following issues in mind:

- The effective maximum transfer size for the port is the minimum of the port's max_xfer_size and the overlying device's maximum transfer size. The FC driver does not coalesce smaller requests into larger ones.
- Some PCI slots on a few machines have an overall limit on the amount of PCI bus address space available for DMA mapping. On those machines it might not be possible to configure both ports of a dual-port FC adapter with increased max_xfer_size. If that happens, the Fibre Channel adapter driver logs an error and one or both of the adapter ports remain in the Defined state.

3.5 LVM configuration: Volume groups and logical volumes

The Logical Volume Manager (LVM) is a set of operating system commands, library subroutines, and other tools that allow the user to establish and control logical volume storage. The LVM controls disk resources by mapping data between a more simple and flexible logical view of storage space and the actual physical disks. The LVM does this using a layer of device-driver code that runs above traditional disk device drivers. This logical view of the disk storage is provided to the applications, and is independent of the underlying physical disk structure.

To manage the disk storage the LVM uses a hierarchy of structures that have a clearly defined relationship between them. The lowest element in this structure is the PV. We have seen in previous sections how we started building the storage layout (in Figure 3-1 on page 86 and Figure 3-2 on page 87) by taking the disks, and creating arrays and LUNs. Each LUN is seen by the LVM as a physical volume and has a name, usually /dev/hdisk*x* (where *x* is a unique integer on the system).

Every physical volume in use belongs to a volume group (VG) unless it is being used as a raw storage or a readily available spare (also known as Hot Spare). A VG is a collection of one or more PVs. Within each volume group, one or more logical volumes (LVs) are defined. LVs are the way to group information located on one or more PVs. LVs are an area of disk used to store data that appears to be contiguous to the application, but can be non-contiguous on the actual PV. It is this definition of a LV that allows them to be extended, relocated, span multiples PVs, and have their contents replicated.

Now that we have a basic understanding of the LVM, we look at creation and configuration in detail.

3.5.1 Creating and configuring VGs

As discussed previously, a VG is a collection of PVs. When you install a new system, one VG (the root VG, called rootvg) is created. New VGs can be created with the **`mkvg`** command. It is important to remember that the rootvg has attributes that differs from the user defined volume groups. Particularly, it cannot be imported. We recommend you organize PVs into VGs separate from rootvg because of the following reasons:

- For safer and easier maintenance.
  - Operating system updates, reinstallations, and crash recoveries are safer because you can separate user file systems from the operating system so that user files are not jeopardized during these operations.
  - Maintenance is easier because you can update or reinstall the operating system without having to restore user data. For example, before updating, you can remove a user-defined VG from the system by un-mounting its file systems and deactivating it (using **`varyoffvg`**). If necessary, the volume can be exported (using **`exportvg`**). After updating the system software, you can reintroduce the user-defined VG (using **`importvg`**), activate it (**`varyonvg`**), and then remount its file systems.
- For physical partition sizes. All PVs within the same VG must have the same physical partition size. To have PVs with other physical partition sizes, we need to place each size in a separate VG. We talk about what the recommended physical partition size is for the VGs related to DB2 in “Physical partition size” on page 121.
- For high availability (HA) purposes. Depending on the HA solution chosen, we need to have the VG created in such a way that more than one system can access those VGs. More on this topic in “Other considerations” on page 124.

We want to create as few VGs as possible to make managing the resources easier, but enough to isolate its data if needed. For a typical OLTP environment we suggest creating one VG for data (permanent, temporal and backup), one VG for the active logs, and one VG for the archival logs. For a DW environment we suggest one VG for data (permanent, temporal, and backup) and active logs, and one VG for archival logs.

The reason for having one fewer VG in DW environments is that we want the active logs distributed in the database partitions. Because active logs are not as performance critical as in an OLTP environment, they do not require dedicated disks for them.

Next, we discuss performance decisions that we need to make before the actual creation.

Physical partition size

When you add a PV to a VG, the PV is partitioned into contiguous, equal size units of space called physical partitions (PPs). A PP is the smallest unit of storage allocated. PVs inherit the VGs physical partition size, which can only be set when the VG is created.

The LVM limits the number of physical partitions that a PV can have. Those limits depend on the type of VG chosen:

- Original
- Big
- Scalable

For Original or Big VGs, the maximum number of PP per PV is 1016. That limit can be changed by a factor to make it larger. The factor is between 1 and 16 for Original VGs and 1 and 64 for Big VGs. The maximum number of PPs per PV for a VG changes to 1016 * factor. When using this factor the maximum number of PVs for that VG is reduced by MaxPVs / factor. For example, an Original VG can have up to 32 PVs. If we decide to increase the number of PPs by a factor of 4, we are able to have 1016 * 4 = 4064 PPs per PV for that VG, but now we only can have 32 / 4 = 8 PVs in that VG. For Big VGs it is the same, except that the largest number of PVs we can handle is 128, so in this example we end up with 32 PVs as the maximum.

Scalable VGs can accommodate up to 1024 Pvs, 256 LVs and 32,768 PPs by default. The number of LVs and PPs can be beyond the default values. In particular, PPs can be increased up to 2,097,152 (Maximum PPs per VG is expressed in units of 1024 PPs, the maximum being 2048, that is 1024 * 2048 = 2,097,152 PPs).

The disadvantage is than increasing these defaults can significantly increase the size of the VG Description Area (VGDA)[3].

These values must only be increased as needed because they cannot be decreased. Meanwhile, as the VGDA space increases all VGDA update operations (creating a LV, changing a LV, adding a PV, and so on) can take longer and longer to run.

Another consideration for the choice of VG type is that the LVM with Original and Big VGs reserves by default the first 512 bytes of the volume for the LV control block. Therefore, the first data block starts at an offset of 512 bytes into the volume. Care is taken when laying out the segment size of the logical drive to enable the best alignment. You can eliminate the LV control block on the LV by using a Scalable VG, or by using the **`-T 0`** option for Big VGs.

Our suggestions is to use Scalable VGs. This allows accommodating large PV sizes, large number of PVs in the VG and allowing for future growth without the risk of hitting the limitations that Original or Big VGs have. Furthermore, we eliminate the risk of alignment problems as explained before.

Now that we have decided to use Scalable VGs, we need to choose the PP size and the maximum number of PPs for the VG. The first step is to know how much physical storage we need to accommodate under that VG. The result of PP Size * Maximum Number of PP per VG can accommodate at least that capacity. A best practice for Original or Big VGs for PP size is to go as small as possible given the maximum PPs per PV restrictions. The reason for that is that PP size has a performance impact when doing mirroring at the LVM level or in certain cases when doing LV stripping. Because we do not recommend the usage of mirroring at the LVM level or LV stripping of differently-sized LVs, then the choice of a PP size only influences the minimum size allocation on the PV.

This parameter does not influence the IO performance, so it is not performance critical to go small. So the only restriction that the PP size has in our configuration is that we are not able to create a file system smaller than a PP. Also, by going small on the PP size we might need to increase the maximum number of PP per VG and that negatively impacts the performance of VGDA operations, as explained before, so we do not want to choose a large number in that area.

A good size for PPs is 64 Mb. Then, depending on the capacity of all the storage in the volume group, you can calculate what the maximum number of PP per VG is. For example, if we want to use 64 Mb PPs and a maximum number of PP per

[3] The VGDA contains information about the VG, the LVs that reside in the VG, and the PVs that make up the volume group

VG of 512 (that is 512 * 1024 = 524,288 PPs), we can accommodate up to 32 TB of capacity in that VG. You can use the following command to create such a VG:

```
mkvg -S -y <vgName> -s 64 -P 1024 f <hdiskList>
```

Logical track group (LTG) size

When the LVM receives a request for an I/O, it breaks the I/O down into what is called logical track group (LTG) sizes before it passes the request down to the device driver of the underlying disks. The LTG is the maximum transfer size of a logical volume and is common to all the logical volumes in the volume group.

value is set by the **varyonvg** command using the **-M** flag. When the LTG size is set using the **-M** flag, the **varyonvg** and **extendvg** commands might fail if an underlying disk has a maximum transfer size that is smaller than the LTG size. To obtain the maximum supported LTG size of your hard disk, you can use the **lquerypv** command with the **-M** flag. The output gives the LTG size in KB, as shown in the following example.

```
# /usr/sbin/lquerypv -M hdisk0
```

The **lspv** command displays the same value as MAX REQUEST, as shown in Example 3-2.

Example 3-2 lspv command

```
# lspv hdisk0
PHYSICAL VOLUME: hdisk0 VOLUME GROUP: rootvg
PV IDENTIFIER: 000bc6fdbff92812   VG IDENTIFIER
000bc6fd00004c00000000fda469279d
PV STATE: active
STALE PARTITIONS: 0 ALLOCATABLE: yes
PP SIZE: 16 megabyte(s) LOGICAL VOLUMES: 9
TOTAL PPs: 542 (8672 megabytes) VG DESCRIPTORS: 2
FREE PPs: 431 (6896 megabytes) HOT SPARE: no
USED PPs: 111 (1776 megabytes) MAX REQUEST: 256 kilobytes
FREE DISTRIBUTION:      108..76..30..108..109
USED DISTRIBUTION:      01..32..78..00..00
```

You can list the value of the LTG in use with the **lsvg** command, as shown in Example 3-3.

Example 3-3 lsvg command

```
# lsvg rootvg
VOLUME GROUP:        rootvg         VG IDENTIFIER:

000bc6fd00004c00000000fda469279d
VG STATE:              active         PP SIZE: 16 megabyte(s)
VG PERMISSION:         read/write TOTAL PPs: 542 (8672 megabytes)
MAX Lvs: 256 FREE PPs: 431 (6896 Megabytes)
LVs:    9    USED PPs: 111 (1776 megabytes)
OPEN LVs: 8 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): 256 kilobyte(s) AUTO SYNC: no
HOT SPARE: no BB POLICY: relocatable
```

Note that the LTG size for a VG is displayed as dynamic in the **lsvg** command output.

Other considerations

Depending on the High Availability solution being considered, other important considerations when creating volume groups are taken into account. Of particular importance is the Concurrent Volume Group flag. The concurrent access VG is a VG that can be accessed from more than one host system simultaneously. This option allows more than one system to access the PVs. Then, through a concurrent capable VG, they can now concurrently access the information stored on them. Initially, this was designed for high-availability systems, with the high-availability software controlling the sharing of the configuration data between the systems. However, the application must control the concurrent access to the data. Another consideration is the major number for a VG. The major number is a numerical identifier for the VG. It is recommended in multi-host environments that the major numbers are consistent across all hosts.

3.5.2 Creating and configuring logical volumes

An LV is a portion of a PV viewed by the system as a single unit. LVs consist of LPs, each of which maps to one or more PPs. The LV presents a simple contiguous view of data storage to the application while hiding the more complex and possibly non-contiguous physical placement of data.

LVs can only exist within one VG. They cannot be mirrored or expanded onto other VGs. The LV information is kept in the VGDA and the VGDA only tracks information pertaining to one VG.

The creation of LVs allows many levels of control by the user:

- No control
- Physical volume specific control
- Partition location control

The Logical Volume Manager subsystem provides flexible access and control for complex physical storage systems. LVs can be mirrored and stripped with other strip sizes. There are other types of LVs:

- boot: Contains the initial information required to start the system.
- jfs: Journaled File System
- jfslog: Journaled File Systems log
- jfs2: Enhanced Journaled File System
- jfs2log: Enhanced Journaled File System log
- paging: Used by the virtual memory manager to swap out pages of memory

Following Figure 3-1 on page 86 and Figure 3-2 on page 87 we are creating each LV associated with one and only one hdisk. Also, as we discuss in the next section, we are creating a jfs2 file system on each of those LVs, so the jfs2 type is specified when creating the LVs. When specifying the size of the LV, remember that a LV is made of logical partitions and each of those logical partitions corresponds in size to a PP on the PV (in this case only one because we do not have mirroring). Thus, when specifying the LV size, do so in multiples of PP size to void wasted space. The size can be specified in logical partition units (the default) or KB/MB/GB.

The command to use to create each LV is as follows:

```
mklv -t jfs2 -y <lvName> <vgName> <lvSize> <hdiskName>
```

For example, to create one 512 GB LV on vg1 on hdisk1 issue:

```
mklv -t jfs2 -y lv1 vg1 512GB hdisk1
```

Tip: It is a best practices to create a LV, because you can give it a specific name that is relevant to what it contains. If you do not create a LV, AIX creates it with a generic name, if it does not exist when creating a file system.

For OLTP systems (see Figure 3-1 on page 86) we suggest having two LVs per hdisk, one for data and one for backups. This allows parallelism when taking a backup of the database by specifying all the backup file systems as targets.

For DW systems (see Figure 3-2 on page 87), we recommend having three LVs per hdisk, one for data, one for backups and one for active logs/database directory. This allows for data partition isolation.

We might need to create a striped LV if we want to increase capacity of the active logging or archival logging file systems. We recommend you use the exact same type of hdisk and use fine-grain stripping. We suggest a strip size of 32 K or 64 K. Use the **-S** flag to indicate the stripSize when creating the LV.

```
mklv -t jfs2-y lvlogs -S 64K vg2 512GB hdisk7 hdisk8
```

3.6 File systems

A *file system* is a hierarchical structure (file tree) of files and directories. This type of structure resembles an inverted tree with the roots at the top and branches at the bottom. This file tree uses directories to organize data and programs into groups, allowing the management of several directories and files at one time. A file system resides on a single LV. Every file and directory belongs to a file system within a LV.

Because of its structure, a few tasks are performed more efficiently on a file system than on each directory within the file system. For example, you can back up, move, or secure an entire file system. You can make a point-in-time image of a JFS file system or a JFS2 file system, called a snapshot.

To be accessible, a file system must be mounted onto a directory mount point using the **mount** command. When multiple file systems are mounted, a directory structure is created that presents the image of a single file system. It is a hierarchical structure with a single root. This structure includes the base file systems and any file systems you create.

You can access both local and remote file systems using the **mount** command. This makes the file system available for read and write access from your system. Mounting or unmounting a file system usually requires system group membership. File systems can be mounted automatically, if they are defined in the `/etc/filesystems` file. You can unmount a local or remote file system with the **`unmount`** command, unless a user or process is currently accessing that file system.

Both JFS and JFS2 file systems are built into the base operating system. However it is not recommended to use JFS. You can also use other file systems on AIX such as Veritas or GPFS, but for this Redbooks publication, we focus on JFS2. This file system uses database journaling techniques to maintain its structural consistency. It allows the file system log to be placed in the same logical volume as the data, instead of allocating a separate logical volume for logs for all file systems in the VG.

Because write operations are performed after logging of metadata has been completed, write throughput is highly affected by where this logging is being done. Therefore, we suggest using the INLINE logging capabilities. We place the log in the LV with the JFS2 file system. We do not suggest setting any specific size for the log. The INLINE log defaults to 0.4% of the LV size if logsize is not specified. This is enough in most cases.

Another consideration for file systems are the mount options. They can be specified in the **`crfs`** command with the **`-a`** option parameter or they can be specified at mount time. We recommend specifying them at the **`crfs`** time to ensure the proper options are used every time the file system is mounted. We suggest the use of the release behind when read option (rbr option).

When sequential reading of a file in the file system is detected the real memory pages used by the file are released after the pages are copied to internal buffers. This solution addresses a scaling problem when performing sequential I/O on large files whose pages are not reaccessed in the near future. When writing a large file without using release-behind, writes go fast whenever there are available pages on the free list.

When the number of pages drops to the value of the minfree parameter, the virtual memory manager uses its Least Recently Used (LRU) algorithm to find candidate pages for eviction. As part of this process, the virtual memory manager needs to acquire a lock that is also being used for writing. This lock contention might cause a sharp performance degradation.

A side effect of using the release-behind mechanism is an increase in CPU use for the same read or write throughput rate compared to without using release-behind. This is due to the work of freeing pages, which is normally handled at a later time by the LRU daemon. Note that all file page accesses result in disk I/O because file data is not cached by the virtual memory manager.

To create a file system, use the following commands:

```
mkdir -p <mountPoint>
crfs -v jfs2 -d <lvName> -m <mountPoint> -a logname=INLINE -a
options=rbr -A yes -p rw
```

3.7 Conclusion

This chapter described the storage principles guiding the storage layout, the mapping of physical storage to logical storage, the layout of file systems, and how DB2 uses those. We looked into OLTP and DW environments and what the best practices are in each case. In general, we must design to spread our hardware evenly, push as much functionality as possible down to the hardware, have a simple LVM design and let DB2 handle the storage automatically. When not sure how to configure certain parameters, the defaults are always a good start. Then, proper monitoring helps to adjust. It is important to check the performance as the system is being built as after the whole storage is laid out. One change can mean a lot of re-work needed, in particular when it comes to the lower layers of the design.

4 Monitoring

Today's environments range from stand-alone systems to complex combinations of database servers and clients running on multiple platforms. In any type of system, the common key for successful applications is performance. Although performance might initially be good, as time goes on, the system might need to serve more users, store more data, and process more complex queries. Consequently, the increased load level on the system affects its performance. This can be the time to upgrade to more powerful equipment. However, before investing in equipment, you might be able to improve performance by monitoring and tuning your environment.

This chapter provides an overview of the tasks involved in monitoring and tuning DB2, AIX, and storage to obtain optimal performance and identify bottlenecks. It describes how to monitor the system resource usage using AIX commands (such as nmon, iostat, vmstat, and so forth). It also discusses the methods available to monitor the database activity, using tools such as the Snapshot™ Monitor, Activity Monitor and **db2pd**. Based on the information provided by these commands and tools, you can make informed decisions about what actions need to be taken to tune the database environment. Later in the chapter we include a few scenarios in which we can use this monitoring information to determine where the bottleneck can be.

This chapter has the following sections:

4.1 Understanding the system

It is important to have a good understanding of your system to figure out what tools to use in the monitoring of bottlenecks and where they might lie. For example, look at Figure 4-1, which details the several components that might typically exist in today's OLTP systems and data warehouses. After you know the architecture, it becomes easier to perform calculated tests at the various levels to determine where the problem can lie and what evidence to look for which suggests a possible cause and consequently a solution.

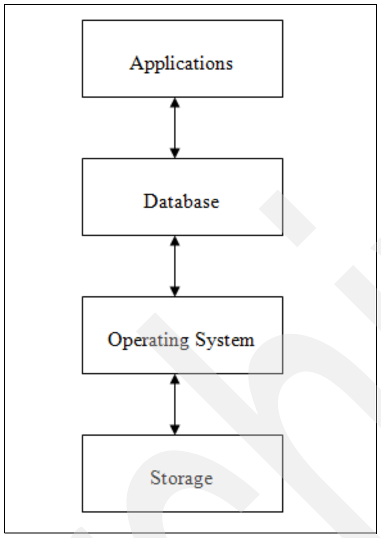


Figure 4-1 Components involved

4.2 Benchmarking

After following the various best practices to configure the system, it is then desirable to run a benchmark and establish the baseline. That is, for a given workload X the response time for the various queries or jobs is Y. This applies to both OLTP and data warehouses such that you can establish a baseline that dictates how long a particular query or report takes. It is important to establish a baseline and collect evidence of the system performing at this level, because if the system performs at a substandard level in the future, you have a comparison data that can aid in determining what can have lead to the performance degradation.

The last point worth considering is of the analogy of comparing apples to apples, For example, after migrating the various applications from the testing environment to the production environment, the performance might be better or worse. In such a situation a holistic picture must be taken to compare the workload, configuration, hardware, and the like.

4.3 Determine the possible causes

There are many possible causes of performance problems. In the following sections, we describe those most frequently encountered.

4.3.1 Poor application design

When you experience performance problems, in many cases these stem from poor application design and inefficient programs. The underlying database, operating system, or storage itself might not have any problems at all. For example, SQL statements written inappropriately can degrade overall performance, even though the system might be finely tuned and configured. These type of problems are beyond the scope of this chapter.

4.3.2 Poor system and database design

Poor design of your system or databases can be also a reason for performance problems. Inefficient disk storage layout, or failing to consider performance when designing and implementing your database, degrades performance and can be difficult to fix after your production system has been started. Chapter 2, “AIX configuration” on page 35 and Chapter 3, “Storage layout” on page 85 describe tips to take into consideration when designing disk storage layout and databases.

4.3.3 System resource shortages

System resource shortages can cause bottlenecks in your database system:

- CPU

  Too many users, or running applications on a CPU, might cause system degradation.

- Memory

  Every process uses physical memory. If you have insufficient memory, you might find that applications fails, or your system starts thrashing.

- Disk I/O

  I/O performance can play an important part in a database system. Too much activity on a single disk or I/O bus might cause performance problems.
- Network

  Unless you are in a stand-alone environment, the network plays an important role. If the network is too slow, this might appear to the client as a performance problem at the database server.

It is these latter shortages and poor system and database design that are the focus of this chapter.

4.4 Planning monitoring and tuning

When you carry out a performance monitoring and tuning project, the worst possible approach is to change the value of many parameters without having any idea of what is causing the performance problem. Barring miracles, performance only gets worse, and you never know which parameter was the cause. So, even if you are anxious for results, you benefit from following a more methodical approach:

1. Find which applications or system resources are not performing well.
2. Measure the current performance and set the performance goal.
3. Monitor the system and identify where the bottleneck is.
4. Decide where you can afford to make trade-offs, and which resources can bear an additional load.
5. Change only one performance parameter to relieve the bottleneck.
6. Execute the applications again to monitor your system, and check if the performance meets your goal.
7. If the performance does not meet the goal, go back to step 3.

Note: Do not forget the Pareto's Principle, also known as the 80-20 rule. In our context, 80% of the performance benefits can be the result of tuning 20% of the various parameters.

4.5 Monitoring tools for DB2

DB2 provides a suite of monitoring tools that can be effectively used to diagnose performance problems. The collective power of AIX and DB2 diagnostics can be used to resolve performance issues quickly. In this section, we discuss the most useful DB2 performance diagnostics, ones that we can rely on to determine the overall health of the database.

DB2 Monitoring Tools can be broadly classified into the following two categories:

- Point-in-time monitoring
- Traces

4.5.1 Point-in-time monitoring tools

The monitoring tools in this category are snapshot monitors, workload management (WLM) aggregate functions starting DB2 Version 9.5, and in-memory metrics in DB2 9.7. These tools provide us with a picture of the database at a given point-in-time. They provide summarized data, where counters, timers, and histograms maintain running totals of activity in the system. It is strongly recommended to collect them on a regular basis for ongoing monitoring. They are light in weight and can be useful in providing a picture of database performance. If a performance problem arises, you can go back and understand what has been happening and what has changed.

4.5.2 Traces

Tools in this category provide motion picture monitoring, which records the execution of a series of individual activities. This is achieved with trace-like mechanisms, such as event monitors (especially statement event monitors) and WLM activity monitors. These tools provide a much more detailed picture of system activity and therefore, produce huge volumes of data. This imposes a much greater overhead on the system. They are more suitable for exception monitoring when you need in-depth understanding of what is causing a performance issue.

The recent trend in DB2 releases has been to move towards providing SQL access to data generated by the monitoring tools. This makes monitoring more manageable, as it allows you to redirect the output of snapshots and event monitors back into tables. You can use the power of SQL to delve through the historical monitoring data and get insights into your system performance.

Let us look at the most commonly used point-in-time diagnostics in DB2: Snapshots and db2pd.

4.5.3 Snapshots

Snapshots are the most commonly used performance diagnostics. DB2 provides a set of administrative views that provides relational access to snapshot data. Administrative views are available in SYSIBMADM schema. Examples of administrative views are:

- SNAPDB,
- SNAPBP
- SNAPDBM
- SNAPAPPL
- BP_READIO

To get a complete list, use the following SQL:

```
select tabname from syscat.tables where tabschema='SYSIBMADM'
```

The instance-level monitor switches must be turned on if snapshot table functions and administrative views are to access the data. For example, DFT_MON_BUFPOOL enables collection of buffer pool monitoring data.

All snapshot elements provide good information but let us look at the most useful ones: the Key Performance Indicators.

DB2 Key Performance Indicators (KPIs) and Rules of Thumb (RoT)

Most of these KPIs are derived from database snapshot (**`GET SNAPSHOT FOR DATABASE`** command). We use the columns of administrative view, SYSIBMADM.SNAPDB for most of the metrics. If you are running in a multi-partition environment, you must include DBPARTITIONNUM in your monitoring SELECT statements, to distinguish the rows that you get back for each partition.

Buffer Pool Hit Ratio

This metric measures percentage of reads that got satisfied in the buffer pool and did not require physical reads. It is calculated as shown in Example 4-1.

Example 4-1 Buffer Pool Data/Index Hit Ratio

```
Data Hit Ratio

100 * (POOL_DATA_L_READS – POOL_DATA_P_READS) /POOL_DATA_L_READS

Index Hit Ratio

100 * (POOL_INDEX_L_READS – POOL_INDEX_P_READS) /POOL_INDEX_L_READS
```

For OLTP, the RoT is to have Data Hit Ratio of at least 80% and Index Hit Ratio of at least 95%.

Data warehouse workloads typically do large sequential table scans and therefore do not have a high Buffer pool Hit Ratio. Buffer pool Temporary Data/Index Hit Ratio is relevant for data warehouse workloads, as they are characterized by complex queries that involve large sorts and hash joins. See Example 4-2.

Example 4-2 Buffer pool Temporary Data/Index Hit Ratio

```
Temporary Data Hit Ratio

100 * (POOL_TEMP_DATA_L_READS - POOL_TEMP_DATA_P_READS) /

POOL_TEMP_DATA_L_READS

Temporary Index Hit Ratio

100 * (POOL_TEMP_INDEX_L_READS - POOL_TEMP_INDEX_P_READS) /

POOL_TEMP_INDEX_L_READS
```

Ideally we want this to be high to avoid temporary data physical I/O. However, large scans and therefore, a low hit ratio is unavoidable.

Average Buffer pool I/O Response Time

This is a measure of the I/O response time and can be used in conjunction with AIX monitoring tools to determine if there is a potential I/O Bottleneck. Keeping modern storage subsystems in mind, average buffer pool read/write time is ~10 milliseconds.

Example 4-3 Overall Average Read Time

```
Overall Average Read Time(in ms)

POOL_READ_TIME /

(POOL_DATA_P_READS + POOL_INDEX_P_READS + POOL_TEMP_DATA_P_READS +
POOL_TEMP_INDEX_P_READS)
```

It can be useful to determine the average asynchronous and synchronous read time. Because OLTP is characterized by random reads, all the reads are synchronous. Synchronous reads are done by the agents and significantly affect the latency of transactions. See Example 4-4 on page 137.

Example 4-4 Average Asynchronous/Synchronous Read Time

```
Average Asynchronous Read Time (in ms)

POOL_ASYNC_READ_TIME / (POOL_ASYNC_DATA_READS + POOL_ASYNC_INDEX_READS)

Average Synchronous Read Time (in ms)

(POOL_READ_TIME - POOL_ASYNC_READ_TIME)/
((POOL_DATA_P_READS + POOL_INDEX_P_READS + POOL_TEMP_DATA_P_READS +
POOL_TEMP_INDEX_P_READS) - POOL_ASYNC_DATA_READS +
POOL_ASYNC_INDEX_READS))
```

Similarly, you can get the average response time for buffer pool writes, as shown in Example 4-5.

Example 4-5 Average Write Time

```
Average Write Time (in ms)
POOL_WRITE_TIME / (POOL_DATA_WRITES + POOL_INDEX_WRITES +
POOL_XDA_WRITES)
```

We can further breakup the Write Response time into Average Asynchronous Write Time and Synchronous Write time, as shown in Example 4-6.

Example 4-6 Average Asynchronous/Synchronous Write Time

```
Average Asynchronous Write Time (in ms)

POOL_ASYNC_WRITE_TIME / (POOL_ASYNC_DATA_WRITES +
POOL_ASYNC_INDEX_WRITES + POOL_ASYNC_XDA_WRITES)

Average Synchronous Read Time(in ms)

(POOL_WRITE_TIME - POOL_ASYNC_WRITE_TIME) / ((POOL_DATA_WRITES +
POOL_INDEX_WRITES + POOL_XDA_WRITES) –
  (POOL_ASYNC_DATA_WRITES + POOL_ASYNC_INDEX_WRITES +
POOL_ASYNC_XDA_WRITES)
)
```

You can query the SYSIBMADM.BP_READ_IO and SYSIBMADM.BP_WRITE_IO to obtain these I/O metrics for all buffer pools. See Example 4-7.

Example 4-7 Query SYSIBMADM.BP_READ_IO & SYSIBMADM.BP_WRITE_IO

```
SELECT AVERAGE_READ_TIME_MS, AVERAGE_ASYNC_READ_TIME_MS,
AVERAGE_SYNC_READ_TIME_MS FROM SYSIBMADM.BP_READ_IO;
```

Transaction Log Response Time

Log response time significantly affects the latency of transactions in an OLTP workload. It is therefore desirable to have low log I/O latency, not exceeding 10 milliseconds. Consider enabling storage write cache to reduce the log write latency. See Example 4-8.

Example 4-8 Average Log Write Time

```
Average Log Write Time (in ms)

(LOG_WRITE_TIME_S*1000 + LOG_WRITE_TIME_NS/1000.0) / NUM_LOG_WRITE_IO
```

A high number of log buffer full, NUM_LOG_BUFFER_FULL, suggests that the log buffer had to be flushed out to the disk as it became full. Increase the Log Buffer Size to avoid this situation.

Page Cleaning

In OLTP, page cleaners play a crucial role. They asynchronously write dirty pages to the disk. This ensures that there are clean slots available for use when a new page needs to be read into the buffer pool. We want to make sure that most of the writes are being done asynchronously by the page cleaners. If not, it is the agent that is doing the writes which is expensive. See Example 4-9.

Example 4-9 Page Cleaning Ratio

```
Page Cleaning Ratio

100 * (POOL_ASYNC_DATA_WRITES + POOL_ASYNC_INDEX_WRITES +
POOL_ASYNC_XDA_WRITES) / (POOL_DATA_WRITES + POOL_INDEX_WRITES +
POOL_XDA_WRITES
```

A 95% page cleaning ratio is considered good.

No victim buffers available

POOL_NO_VICTIM_BUFFER counts the number of times the free page list was empty when a clean page was needed. Consider this as an early warning of inefficient cleaning. If this number gets large, try increasing the number of cleaners, or decreasing CHNGPGS_THRES or SOFTMAX to trigger cleaning at a higher frequency

Dirty page steals

Having no victim buffers available and dirty page steals go hand-in-hand, as dirty steals are indicative of no suitable victim pages being found. No victim buffers available is more sensitive than Dirty Page Steals.

POOL_DRTY_PG_STEAL_CLNS counts the number of times a dirty page has to be cleaned before a read. This results in synchronous writes that can be detrimental for throughput. A dirty page steal rate greater than 1 per 100 transactions can be considered non-trivial. This counter is low, ideally zero.

Dirty page threshold / LSN Gap Triggers

There are three ways to trigger cleaners. We need cleaners to run to have free pages, and we want to avoid dirty page steals. Dirty page threshold triggers, POOL_DRTY_PG_THRSH_CLNS and LSN gap triggers, POOL_LSN_GAP_CLNS are considered good cleaner triggers. If their number gets too large, we might be over-cleaning. Increasing CHNGPGS_THRESH or SOFTMAX decreases the cleaner frequency.

Consider enabling DB2 Alternate Page Cleaning (db2set DB2_USE_ALTERNATE_PAGE_CLEANING=ON). This proactively cleans the dirty pages from the buffer pool such that there is a constant writing of dirty pages that can alleviate the I/O spikes associated with the triggers.

Prefetch Ratio

This represents the percentage of asynchronous reads done by the prefetcher and is particularly relevant for sequential scans. RoT is to keep this close to 100%. Prefetch Ratio is high for data warehouse workloads

```
(POOL_ASYNC_DATA_READS + POOL_ASYNC_INDEX_READS) / (POOL_DATA_P_READS +
POOL_INDEX_P_READS + POOL_TEMP_DATA_P_READS + POOL_TEMP_INDEX_P_READS)
```

Rows Read/Row Selected Ratio

This is a useful metric. It determines how many rows had to be read to return the selected rows. A high ratio is an indication of large table scans and indicates possibility of creating indexes. This is particularly relevant for OLTP, where we do not want this ratio to be greater than 10–20. You can drill down to Statement Snapshots to find the culprit SQLs. Look for SQLs that have Rows Read much higher than the Number of Executions.

```
ROWS_READ / ROWS_SELECTED
```

Note that repeated in-memory table scans can also consume significant CPU.

Package Cache Hit Ratio

This determines the fraction of SQL prepares that cannot be satisfied by dynamic SQL cache. It is calculated as:

```
(1-(pkg_cache_inserts / pkg_cache_lookups))*100%
```

A low package cache ratio is either because we are re-compiling the same SQL statement with other literal values or because of low package cache size. RoT is to have it close to 100% at steady state.

Sort Metrics

When using private sort memory, SHEAPTHRES is a soft limit. If the SHEAPTHRES limit is exceeded, new sorts get significantly less than their optimal amount of memory, degrading query performance. Database Manager Snapshot can be used to determine if we exceeded the SHEAPTHRES. If the Private Sort heap high water mark is greater than SHEAPTHRES, consider increasing SHEAPTHRES.

Another important sort metrics is the number of Sort overflows, SORT_OVERFLOWS. It represents how many sorts spilled to the disk. It is good to have near to 0 sort overflows for OLTP workloads.

Lock Metrics

Contention issues, deadlocks, and lock timeouts can be attributed to poor application design. They can drastically affect the response time and throughput.

- LOCK_ESCALS
- LOCK_TIMEOUTS
- DEADLOCKS

Lock escalations (measured by LOCK_ESCALS) happen when we run out of space on the locklist. Increase LOCKLIST to avoid escalations. Lock timeouts are not necessarily fatal but they do decrease the concurrency. LOCK_TIMEOUT is set to catch problem situations. You can further delve into an application snapshot to determine the percentage of applications in Lock-wait state to determine if there are any locking issues.

A deadlock is created when two applications are each locking data needed by the other, resulting in a situation when neither application can continue to process.

The best tool for collecting information about a deadlock is a detailed deadlock event monitor, which must be defined and enabled before the problem occurs. If you do not have the default deadlock event monitor running, you must create and enable a detailed deadlock event monitor.

For information about the default deadlock monitor, see the following Web page:

`http://publib.boulder.ibm.com/infocenter/db2luw/v9/index.jsp?topic=/com.ibm.db2.udb.admin.doc/doc/c0005419.htm`

Dynamic SQL Metrics

Dynamic SQL Snapshot gives us good insights into all dynamic SQLs processed. This helps us determine the hottest statements in the database and see if there is an opportunity to improve their performance. You can query the SYSIBMADM.SNAPDYN_SQL to access dynamic SQL snapshot data.

NUM_EXECUTIONS, NUM_COMPILATIONS, ROWS_READ, TOTAL_SORT_TIME, STMT_SORTS and hit ratios are the key metrics we look at while determining the performance characteristics of dynamic SQLs.

There are instances when you notice a big difference between the execution time and User + System CPU Time.

```
TOTAL_EXEC_TIME - (TOTAL_USR_CPU_TIME + TOTAL_SYS_CPU_TIME)
```

This is because Total Execution time includes all the white space between open and close (fetches, application code, I/O wait and network wait). It is not a good indicator of real work being done.

You can query the SYSIBMADM. TOP_DYNAMIC_SQL view to find the hottest statements in the database and sort the result set by other metrics (for example, number of executions, average execution time and number of sorts per execution).

Important: An important thing to keep in mind when you compare two snapshots is to normalize the metrics based on the amount of activity recorded during the monitoring period. The number of commit statements, COMMIT_SQL_STMTS, is the metric that reflects the number of transactions and is a good indicator of the amount of activity in the database. This gives what we call in the performance world an apples-to-apples comparison.

4.5.4 db2top

Most entries in snapshots are cumulative values and show the condition of the system at a point in time. Manual work is needed to get a delta value for each entry from one snapshot to the next.

The db2top is a tool that uses DB2 snapshot monitoring APIs to retrieve information about the database system. It is used to calculate the delta values for those snapshot entries in real time. This tool provides a GUI under a command line mode, so that users can get a better understanding while reading each entry. This tool also integrates multiple types of DB2 snapshots, categorizes them, and presents them in other windows for the GUI environment.

db2top has been included in the following DB2 versions and later:

- DB2 UDB v8.2 Fixpak 17 and later
- DB2 UDB v9.1 Fixpak 6 and later
- DB2 UDB v9.5 Fixpak 2 and later
- DB2 UDB v9.7 GA and later

The following Web page provides an in-depth discussion on the db2top tool:

http://www.ibm.com/developerworks/data/library/techarticle/dm-0812wang/index.html

4.5.5 db2pd

db2pd is a standalone utility shipped with DB2 starting with V8.2. It provides a non-intrusive method to view database progress and potential problems. You can use the power of db2pd to get insights into the DB2 engine. Some examples of information provided by db2pd are:

- Memory usage
- Agents
- Applications
- Locks
- Buffer pools
- Dynamic package cache
- Static package cache
- Catalog cache
- Logs
- Table and index statistics

In this section, we discuss db2pd options that you can use to diagnose performance problems.

db2pd –edus

Multi-threaded architecture has been introduced on UNIX starting with DB2 9.5. Prior to DB2 9.5, you can see all active DB2 processes using the **ps** command on UNIX. Starting with DB2 9.5, the **ps** command only lists the parent process, db2sysc. You can use **db2pd –edus** to list all DB2 threads, along with their CPU use. It can be useful to determine which DB2 thread is behind a CPU bottleneck. See Example 4-10.

Example 4-10 db2pd –edus

```
Database Partition 0 -- Active -- Up 0 days 00:47:51
List of all EDUs for database partition 0
db2sysc PID: 13360
db2wdog PID: 13358
db2acd  PID: 13374

EDU ID    TID            Kernel TID     EDU Name           USR (s)      SYS (s)
219       47914806143296 13976          db2pclnr (DTW)       0.000000     0.000000
218       47914814531904 13975          db2pclnr (DTW)       0.000000     0.000000
217       47914822920512 13944          db2dlock (DTW)       0.000000     0.000000
216       47914831309120 13943          db2lfr   (DTW)       0.000000     0.000000
215       47914948749632 13942          db2loggw (DTW)       0.760000     2.450000
214       47914827114816 13925          db2loggr (DTW)       0.090000     0.120000
213       47926399199552 13825          db2stmm  (DTW)       0.040000     0.010000
191       47914835503424 13802          db2agent (DTW)      45.280000     7.810000
190       47914839697728 13801          db2agent (DTW)      46.310000     7.610000
189       47914843892032 13800          db2agent (DTW)      44.950000     7.600000
188       47914848086336 13799          db2agent (DTW)      46.000000     7.700000
```

You can use the AIX **ps** command, **ps -mo THREAD -p <db2sysc pid>**, as well to get details about the EDU threads.

db2pd –memsets and –mempools

These can give an insight into memory usage of DB2 and can be useful to look at when you run into memory bottleneck. They can also be used to detect memory leaks in DB2, which are rare.

-memsets is used to gain a quick, detailed view of how much memory each DB2 memory set is using. See Example 4-11 on page 144.

–mempools is used to drill down further into the memory pools that constitute a memory set.

Example 4-11 Memory set

```
db2pd -memset
Database Partition 0 -- Active -- Up 0 days 10:07:04
Memory Sets:
Name        Address           Id         Size(Kb)   Key        DBP
Type   Unrsv(Kb)  Used(Kb)   HWM(Kb)    Cmt(Kb)    Uncmt(Kb)
DBMS        0x0780000000000000 28311566    36288     0x931FF261  0
0      0          14336      17024      17024      19264
FMP         0x0780000010000000 223346762   22592      0x0        0
0      2          0          576        22592      0
Trace       0x0770000000000000 362807300   137550    0x931FF274  0
-1     0          137550     0          137550     0
```

In Example 4-11:

- `Name` is the name of memory set.
- `Address` is the address of the memory set.
- `Id` is the memory set identifier.
- `Size(Kb)` is the size of memory set in kilobytes.
- `Key` is the memory set key (for Unix-based systems only).
- `DBP` is the database partition server that owns the memory set.
- `Type` is the type of memory set.
- `Unrsv(Kb)` is the memory not reserved for any particular pool. Any pool in the set can use this memory if needed.
- `Used(Kb)` shows memory currently allocated to memory pools.
- `Cmt(Kb)` shows all memory that has been committed by the DB2 database and occupies physical RAM, paging space or both.
- `Uncmt(Kb)` shows memory not currently being used and marked by the DB2 database to be uncommitted.

The DBMS Memory set corresponds to INSTANCE_MEMORY dbm cfg. Let us delve further into the memory pools owned by the DBMS memory set. See Example 4-12.

Example 4-12 db2pd -mempools

```
db2pd -mempools
Database Partition 0 -- Active -- Up 0 days 10:17:47
Memory Pools:
Address            MemSet   PoolName   Id    Overhead    LogSz
LogUpBnd    LogHWM      PhySz       PhyUpBnd    PhyHWM      Bnd BlkCnt
CfgParm
0x07800000000012A8 DBMS     fcm         74    0           0
1931054     0           0           1966080     0           Ovf 0
n/a
0x0780000000001160 DBMS     fcmsess     77    65376       1401568
1687552     1401568     1572864     1703936     1572864     Ovf 3
n/a
0x0780000000001018 DBMS     fcmchan     79    65376       259584
507904      259584      393216      524288      393216      Ovf 3
n/a
0x0780000000000ED0 DBMS     fcmbp       13    65376       656896
925696      656896      851968      983040      851968      Ovf 3
n/a
0x0780000000000D88 DBMS     fcmctl      73    111872      1594237
8675472     1594237     1769472     8716288     1769472     Ovf 11
n/a
0x0780000000000C40 DBMS     monh        11    0           3664
368640      144759      196608      393216      327680      Ovf 4
MON_HEAP_SZ
0x0780000000000AF8 DBMS     resynch     62    41216       155320
2752512     155320      262144      2752512     262144      Ovf 2
n/a
0x07800000000009B0 DBMS     apmh        70    4512        1757972
5898240     1915716     1900544     5898240     1966080     Ovf 86
n/a
0x0780000000000868 DBMS     kerh        52    0           1564968
4128768     1951064     1900544     4128768     2228224     Ovf 167
n/a
0x0780000000000720 DBMS     bsuh        71    0           61369
13828096    2276362     262144      13828096    2490368     Ovf 19
n/a
0x07800000000005D8 DBMS     sqlch       50    0           2560840
2621440     2560840     2621440     2621440     2621440     Ovf 203
n/a
```

```
0x0780000000000490 DBMS     krcbh      69    0          145272
131072      145624      196608      131072      196608      Ovf 11
n/a
0x0780000000000348 DBMS     eduah      72    65440      2621384
2621408     2621384     2686976     2621440     2686976     Ovf 1
n/a
0x0780000010000348 FMP      undefh     59    32000      491600
22971520    491600      524288      23003136    524288      Phy 4
n/a
```

- The (Size-Unrsv) of a memory set is equal to the sum of PhyUpBnd of all memory pools owned by the memory set.
- The Used size of a memory set is almost equal to sum of PhySz of memory pools in the memory set.

You can further delve into the logical memory of a memory pool using the db2pd –memblocks option. You can view this information at the database level as well to get overview of the database memory set and memory pools

If the Used size of memory set is much higher than the sum of the physical size of memory pool, it might indicate a memory leak.

db2pd -tcbstats

This option displays information about the size and activity of tables. The Scan column can be particularly useful in determining table scans. A steadily increasing Scan value on a table can indicate a poor plan and a potential requirement to create an index. In this example, there have been 30419 table scans on the LAST_TRADE table. Note that although LAST_TRADE is a small table containing only 643 rows, repeated in-memory table scans can lead to a potential CPU bottleneck. See Example 4-13.

Example 4-13 db2pd -tcbstat

```
TableName          SchemaNm Scans      UDI        RTSUDI              PgReorgs
NoChgUpdts Reads      FscrUpdates Inserts    Updates Deletes    OvFlReads  OvFlCrtes
RowsComp   RowsUncomp CCLogReads StoreBytes BytesSaved
LAST_TRADE         SW     30419      4769       4769                0          0
20789015   0          0         4769 0          0         0          0          0
256621     -          -
```

db2pd –activestatements

This option indicates the dynamic statements agents are currently executing. You can use this to determine the most active statements in your database. In Example 4-14, statement with Anchor ID = 600 seems to be fairly active. **db2pd –dynamic** can help get the statement text corresponding to Anchor ID=600. See Example 4-14.

Example 4-14 db2pd – activestatements

```
Address            AppHandl [nod-index] UOW-ID     StmtID     AnchID StmtUID
EffISO      EffLockTOut EffDegree   EntryTime          StartTime          L
0x00002B9674BF4E20 138      [000-00138] 1907       15         928    1         0
-1          0           Wed Dec  2 06:33:20 Wed Dec  2 06:33:20 W
0x00002B9674BFABE0 118      [000-00118] 1956       19         600    1         0
-1          0           Wed Dec  2 06:33:20 Wed Dec  2 06:33:20 W
0x00002B9674C37D80 131      [000-00131] 1946       38         701    1         0
-1          0           Wed Dec  2 06:33:20 Wed Dec  2 06:33:20 W
0x00002B9674CB0080 144      [000-00144] 1920       23         600    1         0
-1          0           Wed Dec  2 06:33:20 Wed Dec  2 06:33:20 W
0x00002B9674CD9CC0 111      [000-00111] 1886       31         600    1         0
-1          0           Wed Dec  2 06:33:20 Wed Dec  2 06:33:20 W
0x00002B9674C55E40 124      [000-00124] 1901       19         600    1         0
-1          0           Wed Dec  2 06:33:20 Wed Dec  2 06:33:20 W
0x00002B9674C61FC0 130      [000-00130] 1946       13         1011   1         0
-1          0           Wed Dec  2 06:33:20 Wed Dec  2 06:33:20 W
0x00002B9674CFA3A0 143      [000-00143] 1885       11         600    1         0
-1          0           Wed Dec  2 06:33:20 Wed Dec  2 06:33:20 W
0x00002B9674C31FC0 110      [000-00110] 1924       16         760    1         0
-1          0           Wed Dec  2 06:33:20 Wed Dec  2 06:33:20 W
0x00002B9674D0BC00 123      [000-00123] 1898       15         600    1         0
-1          0           Wed Dec  2 06:33:20 Wed Dec  2 06:33:20 W
0x00002B9674C6DB40 136      [000-00136] 1855       15         600    1         0
-1          0           Wed Dec  2 06:33:20 Wed Dec  2 06:33:20 W
```

db2pd –dynamic

`db2pd –dynamic` helps identify the statement text with anchor ID = 600. NumRef indicates how many times a statement has been referenced and hence, indicates how active the dynamic statement. See Example 4-15.

Example 4-15 db2pd –dynamic

```
Database Partition 0 -- Database DTW -- Active -- Up 0 days 00:04:12

Dynamic Cache:
Current Memory Used          4066784
Total Heap Size              4068474
Cache Overflow Flag          0
Number of References         954293
Number of Statement Inserts  12343
Number of Statement Deletes  12072
Number of Variation Inserts  12399
Number of Statements         271

Dynamic SQL Statements:

Address            AnchID StmtUID    NumEnv     NumVar     NumRef     NumExe     Text
0x00002B9674E4F3A0 14     1          1          1          24009      24007
Insert into HISTORY (H_W_ID, H_D_ID, H_C_W_ID, H_C_D_ID, H_C_ID, H_DATE, H_AMOUNT,
H_DATA) values (?, ?, ?, ?, ?, ?, ?, ?)

0x00002B9674E4FEC0 600    1          1          1          250037     250037
Select S_QUANTITY, S_DIST_01,S_DIST_02, S_DIST_03, S_DIST_04, S_DIST_05, S_DIST_06,
S_DIST_07, S_DIST_08, S_DIST_09, S_DIST_10, S_YTD, S_ORDER_CNT, S_REMOTE_CNT, S_DATA
from STOCK where S_W_ID = ?and S_I_ID = ?
```

db2pd –static

Snapshots only monitor the dynamic SQL statements. Although we can use the statement event monitor to track Static SQLs, they generate huge volumes of data, making them impractical for on-going monitoring. **db2pd –static** is a handy option to determine the active Static SQL statements.

See Example 4-16. Section 9 of package PAYS is active as it is being referenced by 15 concurrent users. You can query SYSCAT.STATEMENTS to determine the query that correspond to Section 9 of PAYS package.

Example 4-16 db2pd – static

```
Database Partition 0 -- Database DTW -- Active -- Up 0 days 00:06:32
Static Cache:
Current Memory Used          232831
Total Heap Size              4068474
Cache Overflow Flag          0
Number of References         716105
:Sections:
Address    Schema   PkgName  UniqueID SecNo NumRef    UseCount  StmtType Cursor
W-Hld
0x3FB55138 SW       PAYS     JAxPMQCV 7     1604628   0         6
NO
0x3FB5527C SW       PAYS     JAxPMQCV 8     1604626   0         4
NO
0x3FB553C0 SW       PAYS     JAxPMQCV 9     1604625   15        4
NO
```

This shows how **db2pd** can be effective at times in diagnosing performance issues.

4.6 Monitoring enhancements in DB2 9.7

DB2 9.7 introduces several enhancements in the area of database monitoring. The primary goal of these enhancements is to reduce the time and effort required to detect and resolve common database performance problems. Major enhancements have been made to improve the diagnostics as well as to reduce the performance impact of monitoring, both in the area of point-in-time monitoring and event monitors. We discuss the changes in point-in-time monitoring in the following sections.

4.6.1 In-memory metrics: New monitoring infrastructure

In DB2 9.7, we begin to move away from traditional snapshot towards a new light-weight monitoring infrastructure, In-memory metrics. Key highlights of the new infrastructure are:

- A comprehensive set of Time Spent in DB2 metrics that tells you how and where DB2 is spending time.
- Reduced performance impact of collecting and querying monitoring data.

> **Note:** Tests done in the lab on an OLTP workload suggest that the performance overhead of collecting Snapshots (all switches ON) is 6%, while it is only 3% with In-memory metrics.

- Better control over the granularity of monitoring. You can enable monitoring only for specific service classes.
- A common interface through Package Cache for monitoring both Static and Dynamic SQL

How to use the new monitoring infrastructure

Data for monitoring elements is accumulated and available for querying along multiple perspectives:

- System level
- Activity level
- Database object level

System level monitoring

System level monitoring reflects the effort DB2 spent in processing application requests. It allows you to determine what the system is doing as a whole, as well as for particular categories and subsets of workload.

Request metrics are accumulated in-memory and reported at the following points:

- Service Subclass
- Connection
- Workload Definition
- Unit of Work

As shown in Figure 4-2, a database request is processed by an agent. The agent gathers the in-memory metrics and rolls up the metrics to other aggregation points at regular intervals. (Transaction boundary or every 10 seconds, whichever comes first).

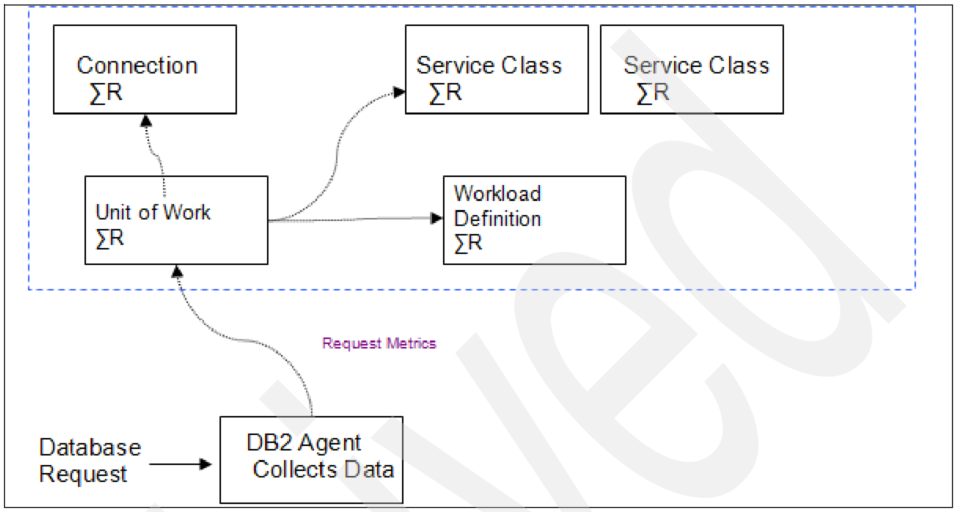


Figure 4-2 How metrics are gathered

Monitoring can be enabled at two levels:

- Database level

```
UPDATE DB CFG FOR <DBNAME> USING MON_REQ_METRICS BASE
```

- Service Super Class level

```
CREATE SERVICE CLASS <SERVICE-CLASS-NAME> COLLECT REQUEST DATA BASE
```

You can also monitor only a set of workloads by enabling monitoring only for the service class to which the workload maps. The database level decides the minimum level of monitoring. For example, if MON_REQ_METRICS=BASE, then request metrics are collected regardless of service class setting. MON_REQ_METRICS is set to BASE for new databases and to NONE for migrated database.

System in-memory metrics are exposed through the following table functions:

- MON_GET_UNIT_OF_WORK, MON_GET_UNIT_OF_WORK_DETAILS
- MON_GET_WORKLOAD, MON_GET_WORKLOAD_DETAILS
- MON_GET_CONNECTION, MON_GET_CONNECTION_DETAILS
- MON_GET_SERVICE_SUBCLASS, MON_GET_SERVICE_SUBCLASS_DETAILS

Let us see a few examples.

List the active connections where we are spending most of our time

We query the MON_GET_CONNECTION table function to get a high level view of where we are spending time. It accepts two arguments:

- Application_handle
- Member

If the application handle is NULL, details are returned for all connections. Member is used to specify the node in DPF environment. Specify -1 for the current node and -2 for all nodes.

TOTAL_RQST_TIME is the total time DB2 spent for all requests for an application. ACT_COMPLETED_TOTAL represents the number of activities (SQL statements) executed by the application. See Example 4-17.

Example 4-17 List the active connection

```
SELECT APPLICATION_HANDLE, ACT_COMPLETED_TOTAL, TOTAL_RQST_TIME
FROM TABLE(MON_GET_CONNECTION(CAST(NULL AS BIGINT), -2))
ORDER BY TOTAL_RQST_TIME DESC
APPLICATION_HANDLE   ACT_COMPLETED_TOTAL  TOTAL_RQST_TIME
-------------------- -------------------- --------------------
                  11               128279               198963
                  14               128377               198813
                   9               128580               198615
                  13               128491               198486
                  12               126753               198403
                   8               125708               198371
                   7               126950               197918
                  10               127289               197889
                  37                    1                58127
                  15                   10                33079
```

In the output, application handle = 37 has executed only one activity, while other applications have completed relatively high number of activities.

What are the transactions (UOW) the connections are currently executing?

We can drill down to the UOW level through MON_GET_UNIT_OF_WORK to determine this. MON_GET_UNIT_OF_WORK accepts application_id and member as arguments. See Example 4-18.

Example 4-18 Drill down to the UOW level

```
SELECT APPLICATION_HANDLE, UOW_ID, TOTAL_RQST_TIME,
WORKLOAD_OCCURRENCE_STATE, UOW_START_TIME FROM
TABLE(MON_GET_UNIT_OF_WORK(NULL,-2))
WHERE APPLICATION_HANDLE IN
( SELECT APPLICATION_HANDLE
  FROM TABLE(MON_GET_CONNECTION(CAST(NULL AS BIGINT), -2) ) AS T
  ORDER BY TOTAL_RQST_TIME DESC FETCH FIRST 10 ROWS ONLY)
ORDER BY TOTAL_RQST_TIME DESC


APPLICATION_HANDLE   UOW_ID      TOTAL_RQST_TIME
WORKLOAD_OCCURRENCE_STATE        UOW_START_TIME
-------------------- ----------- --------------------
-------------------------------- --------------------------
              37           1            65444 UOWEXEC
2009-12-07-08.41.42.321700
              13        5455                0 UOWEXEC
2009-12-07-08.43.27.979914
              12        5500                0 UOWEXEC
2009-12-07-08.43.28.009260
              11        5457                0 UOWEXEC
2009-12-07-08.43.28.030864
              10        5377                0 UOWEXEC
2009-12-07-08.43.28.008118
               9        5445                0 UOWEXEC
2009-12-07-08.43.28.020324
              15          12                0 UOWEXEC
2009-12-07-08.43.27.943657
               8        5509                0 UOWWAIT
2009-12-07-08.43.27.959482
              14        5417                0 UOWEXEC
2009-12-07-08.43.28.025917
               7        5480                0 UOWEXEC
2009-12-07-08.43.28.023564
  10 record(s) selected.
```

Application 37 look suspicious. It seems to be executing a long running transaction. Monitoring the UOWs at regular intervals consistently shows UOW_ID=1 from APPLICATION HANDLE=37 at the top of the list. For other applications, the UOW IDs in a state of execution are constantly changing. We can further drill down to the activity level to determine what the UOW is executing. We discuss this in “Activity level monitoring” on page 155.

If you have configured WLM, you can obtain metrics at the service class level and workload level. Let us see an example:

- Determine the number of completed activities and CPU Time consumed by each service class

  We can query MON_GET_SERVICE_SUBCLASS. See Example 4-19.

Example 4-19 Query the MON_GET_SERVICE_SUBCLASS

```
SELECT varchar(service_superclass_name,30) as service_superclass,
         varchar(service_subclass_name,30) as service_subclass,
       total_cpu_time as total_cpu,
       act_completed_total as SQL_EXECUTED FROM
TABLE(MON_GET_SERVICE_SUBCLASS('','',-2)) AS t
ORDER BY total_cpu desc

SERVICE_SUPERCLASS             SERVICE_SUBCLASS               TOTAL_CPU
SQL_EXECUTED
------------------------------
---------------------------------------------------------------
FINANCE                        TRANSACTCLASS               5673098558
43994853
FINANCE                       REPORTCLASS               1341307246
66
SYSDEFAULTUSERCLASS           SYSDEFAULTSUBCLASS           628763
13
SYSDEFAULTMAINTENANCECLASS     SYSDEFAULTSUBCLASS          113371
0
SRVCLASS                      SYSDEFAULTSUBCLASS               0
0
SYSDEFAULTSYSTEMCLASS         SYSDEFAULTSUBCLASS               0
0

  6 record(s) selected.
```

If you want to view the metrics of a particular class, specify the service super class name and service subclass name as first two arguments of MON_GET_SERVICE_SUBCLASS.

It is good practice to configure WLM such that there is one-to-one mapping between an application and a workload/service sub class. This enables you to determine the performance characteristics of an application by looking at service class/workload monitoring metrics.

Activity level monitoring

This provides reporting from the perspective of the individual activities processed by DB2, such as the execution of individual SQL statements. For SQL statements, the term activity refers only to the execution of the section for that statement. Metrics for other requests or processing that is performed before or after section execution are not included in the activity metrics. See Figure 4-3.

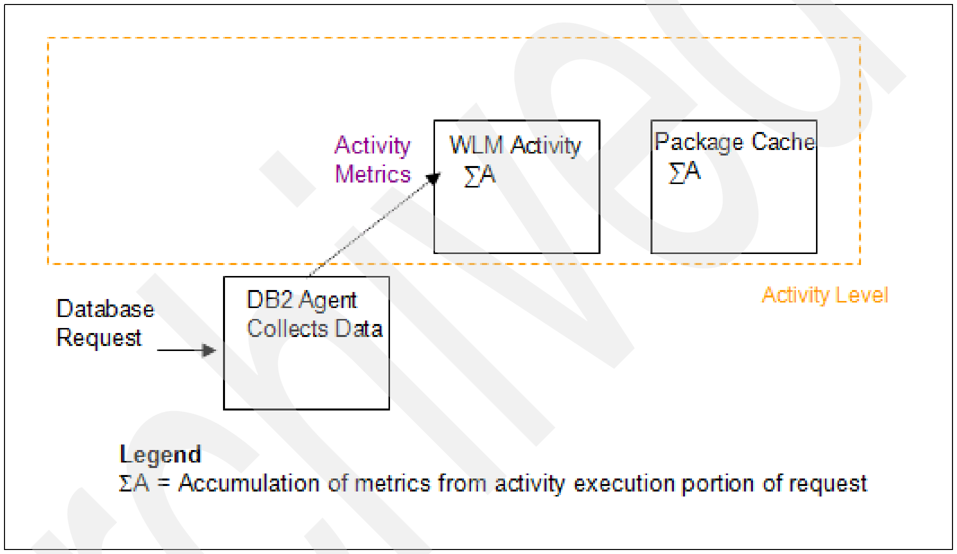


Figure 4-3   Activity level monitoring

A request comes into an agent and is processed. If the request is related to an activity, the agent gathers the metrics from the start of activity execution and, at regular intervals, aggregates them in the activity control block. When the activity completes, those activity execution metrics are propagated to the package cache and aggregated under the specific cached section that was executed (static and dynamic SQL).

Monitoring can be enabled at two levels:

- Database level

```
UPDATE DB CFG FOR <DBNAME> USING MON_ACT_METRICS BASE
```

- Service Super Class level

```
CREATE SERVICE CLASS <SERVICE-CLASS-NAME> COLLECT ACTIVITY METRICS
BASE
```

The database level decides the minimum level of monitoring. For example, if MON_ ACT_METRICS=BASE, then in-memory metrics are collected regardless of service class setting. MON_ ACT_METRCIS is set to BASE for new databases and to NONE for migrated database. Activity metrics are exposed through:

- MON_GET_PKG_CACHE_STMT (Both static and dynamic SQL)
- MON_GET_ACTIVITY_DETAILS (XML)

MON_GET_ACTIVITY_DETAILS returns data about the individual activities in progress when the table function is called. Data is returned in XML format.

MON_GET_PKG_CACHE_STMT returns data for an individual SQL statement section aggregated over all executions of the section. Data is returned in a relational form.

Continuing our previous example, we want to find details about the SQL statements that the problematic UOW is currently executing.

MON_GET_ACTIVITY_DETAILS returns metrics about a specific activity identified by its application handle, unit of work ID, and activity ID

We can determine the activity_id using the WLM Table function, WLM_GET_WORKLOAD_OCCURRENCE_ACTIVITIES_V97, as shown in Figure 4-20 on page 182.

Example 4-20 Determine the activity_id

```
SELECT application_handle, activity_id, uow_id, local_start_time
FROM TABLE (WLM_GET_WORKLOAD_OCCURRENCE_ACTIVITIES_V97(37, -1) ) AS T

APPLICATION_HANDLE   ACTIVITY_ID UOW_ID      LOCAL_START_TIME

-------------------- ----------- ----------- --------------------------

                  37           2           1 2009-11-23-12.26.10.824045
```

Next, we use the activity_id to determine which SQL statement corresponds to this activity and to determine where it is spending time. See Example 4-21.

Example 4-21 Determine which SQL statement corresponds to this activity

```
SELECT actmetrics.application_handle,
       actmetrics.activity_id,
       actmetrics.uow_id,
       varchar(actmetrics.stmt_text, 50) as stmt_text,
       actmetrics.total_act_time,
       actmetrics.total_act_wait_time,
       actmetrics.total_act_time,
       actmetrics.total_act_wait_time,
       CASE WHEN actmetrics.total_act_time > 0
       THEN DEC((
                FLOAT(actmetrics.total_act_wait_time) /
                 FLOAT(actmetrics.total_act_time)) * 100, 5, 2)
       ELSE NULL
       END AS PERCENTAGE_WAIT_TIME
FROM TABLE(MON_GET_ACTIVITY_DETAILS(37, 1 , 2, -2)) AS ACTDETAILS,
XMLTABLE (XMLNAMESPACES( DEFAULT 'http://www.ibm.com/xmlns/prod/db2/mon'),
  '$actmetrics/db2_activity_details'
  PASSING XMLPARSE(DOCUMENT ACTDETAILS.DETAILS) as "actmetrics"
  COLUMNS "APPLICATION_HANDLE" INTEGER PATH 'application_handle',
    "ACTIVITY_ID" INTEGER PATH 'activity_id',
    "UOW_ID" INTEGER PATH 'uow_id',
    "STMT_TEXT" VARCHAR(1024) PATH 'stmt_text',
    "TOTAL_ACT_TIME" INTEGER PATH 'activity_metrics/total_act_time',
    "TOTAL_ACT_WAIT_TIME" INTEGER PATH 'activity_metrics/total_act_wait_time'
  ) AS ACTMETRICS;

APPLICATION_HANDLE ACTIVITY_ID UOW_ID     STMT_TEXT
TOTAL_ACT_TIME TOTAL_ACT_WAIT_TIME PERCENTAGE_WAIT_TIME
------------------ ----------- -----------
-------------------------------------------------- ------------- -------------------
--------------------
            37           2          1 select * from sw.order_line
142045            121702              85.67
1 record(s) selected.
```

This shows that the query corresponding to the long-running UOW is spending 85% of its time in wait. MON_GET_ACTIVITY_DETAILS only gives information about activities that are currently in progress. To determine your hottest SQL statements over a period of time, you can query MON_GET_PKG_CACHE_STMT. It is similar to Dynamic SQL Snapshot but with an added advantage: you can monitor both Static and Dynamic SQLs!

MON_GET_ PKG_CACHE_STMT accepts four arguments:

- Section: Type ‘S’ for Static SQLs and ‘D’ for Dynamic SQLs.
- Executable ID: Specifies a unique identifier for section in package cache.
- Search_args: An optional input parameter of type CLOB(1K), that allows you to specify one or more optional search argument strings.
- Member: Node number.

What are my top 5 hottest SQL statements, sorted by Rows Read?:

If NULL is specified as Executable_ID, we get all rows. To filter rows, we select the Executable IDs of the top 10 SQLs (highlighted in blue) and use them as an argument to the MON_GET_PKG_CACHE_STMT

Unlike Dynamic SQL Snapshot, we get the Rows Returned that can quickly help us determine the SQL statements that have a high Rows Read/Rows Returned Ratio. See Example 4-22.

Example 4-22   What are my top 5 hottest SQL statements, sorted by Rows Read?

```
SELECT
NUM_EXECUTIONS, STMT_EXEC_TIME,
ROWS_READ, ROWS_RETURNED,
SUBSTR(STMT_TEXT,1, 200) AS STMT_TEXT
FROM
 (SELECT EXECUTABLE_ID
    FROM TABLE ( MON_GET_PKG_CACHE_STMT (NULL, NULL, NULL, -2) )
    ORDER BY ROWS_READ DESC FETCH FIRST 5 ROWS ONLY ) T1,
 TABLE (MON_GET_PKG_CACHE_STMT(NULL,T1.EXECUTABLE_ID ,NULL,-2));

                ROWS_READ            ROWS_RETURNED        STMT_TEXT
                --------------------------------------------------------------------------
                          70794652                  350 select nation, o_year, sum(amount) as s
                          57705617                  100 select s_name, count(*) as numwait from
                          56182450                   15 select n_name, sum(l_extendedprice *
                          49149656                  200 select c_name, c_custkey, o_orderkey,
                          48449096                    4 select l_shipmode, sum(case when i
5 record(s) selected.
```

All the queries are aggregate queries, doing huge table scans.

Database object monitoring

This provides reporting from the perspective of operations performed on particular database objects (table, index, buffer pool, table space and container). It provides a complementary perspective of database operation to the workload, allowing the user to pinpoint issues from a data object centric perspective rather than a workload centric perspective. See Figure 4-4.

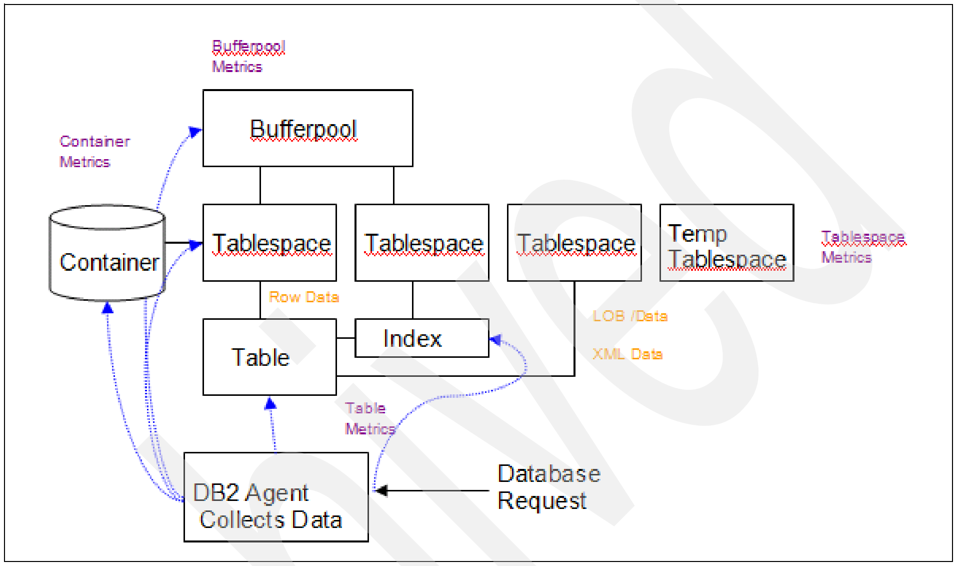


Figure 4-4 Database object monitoring

When agents perform work on the system and gather low-level monitoring metrics, relevant metrics are propagated to accumulation points in in-memory data objects as well as the aforementioned accumulation points in the workload framework.

Database object monitoring can be enabled only at the database level.

```
DB2 UPDATE DB CFG FOR <DBNAME> USING MON_OBJ_METRICS BASE
```

Database objects metrics are exposed through:

- MON_GET_TABLE
- MON_GET_INDEX
- MON_GET_BUFFERPOOL
- MON_GET_TABLESPACE
- MON_GET_CONTAINER

For example, you see a high Rows Read/Rows Selected ratio and want to determine the tables that are generating table scans? This is similar to **db2pd –tcbstats** but gives you the convenience of relational access to the data. See Example 4-23.

Example 4-23 Determine the tables that are generating table scan

```
select tabname, tabschema , table_scans
from table(mon_get_table('','',-2)) as t
order by table_scans desc fetch first 5 rows only"

TABNAME                  TABLE_SCANS          ROWS_READ
---------------------- -------------------  --------------
LAST_TRADE               30419                20789015
CUSTOMER                 16149                   63507
TRADE_REQUEST                1                   11523
```

You can then query the package cache to see which statements are executed against these tables and explore the possibility of creating indexes to eliminate the table scan.

4.6.2 New administrative views

Starting from DB2 9.7 Fixpack 1, there is a new set of administrative views created based on the new monitoring table functions. They can be used to access the most important metrics.

- MON_PKG_CACHE_SUMMARY
- MON_CURRENT_SQL
- MON_CURRENT_UOW
- MON_SERVICE_SUBCLASS_SUMMARY
- MON_WORKLOAD_SUMMARY
- MON_CONNECTION_SUMMARY
- MON_DB_SUMMARY
- MON_BP_UTILIZATION
- MON_TBSP_UTILIZATION
- MON_LOCKWAITS

The MONREPORT module provides a set of procedures for retrieving a variety of monitoring data and generating text reports. Details about the MONREPORT module is available at the following Web page:

http://publib.boulder.ibm.com/infocenter/db2luw/v9r7/index.jsp?topic=/com.ibm.db2.luw.sql.rtn.doc/doc/r0056557.html

Time-spent metrics is one of the most significant enhancements in monitoring. We already touched upon it in the MON_GET_ACTIVITY_DETAILS example, where we used ACT_WAIT_TIME to determine what percentage of total time SQL spent in wait. Let us take a closer look at it.

Time-spent metrics

Identifying the resource bottleneck is the first and the key step towards any performance investigation. Understanding the bottleneck can help you rule out a lot of possible problems early on. See Figure 4-5.

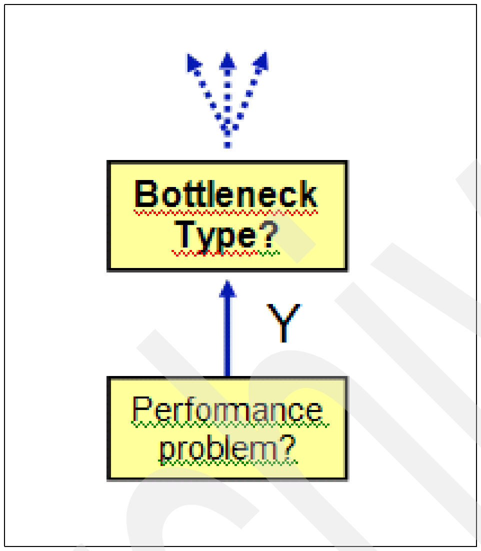


Figure 4-5 Identifying the resource bottleneck

DB2 9.7 introduces the concept of time-spent metrics, a set of metrics that provides a comprehensive breakdown of where and how time is being spent inside DB2. They give us valuable clues about the source of problems by telling where DB2 is spending most of its time. When used with the AIX Monitoring tools, they can drive the investigation in the right direction. There are three categories of time-spent metrics:

- Wait Time
- Component Time
- Component Processing Time

Wait Time

This category helps answer the following question: “What percentage of total time in DB2 is spent on waits, and what are the resources we are waiting on? Examples are:

- Buffer pool Read/Write Wait Time

  (POOL_READ_TIME + POOL_WRITE_TIME)
- Direct Read/Write Wait Time

  (DIRECT_READ_TIME + DIRECT_WRITE_TIME)
- Lock Wait Time

  (LOCK_WAIT_TIME)
- Log I/O Wait Time

  (LOG_DISK_WAIT_TIME + LOG_BUFFER_WAIT_TIME)
- TPC/IP (Send/Receive) Wait Time

  (TCPIP_SEND_WAIT_TIME + TCPIP_RECV_WAIT_TIME)
- FCM (Send/Receive) Wait Time

  (FCM_SEND_WAIT_TIME + FCM_RECV_WAIT_TIME)

Note: Note that the Buffer pool Read Time and Buffer pool Write time represent the synchronous read and write time.

Component Time

This category helps answer the following question: “Where am I spending time within DB2?”

Components in DB2 map to other stages of processing within DB2 (examples, Sort, Query Compilation, Transaction end processing (Rollback/Commit) and Utilities). Metrics in this category measure the elapsed time spent across major DB2 Components. Examples:

- Compile Time

  (TOTAL_COMPILE_TIME + TOTAL_IMPLICIT_COMPILE__TIME)
- Section Execution Time

  (TOTAL_SECTION_TIME)
- Sort Time

  (TOTAL_SECTION_SORT_TIME)

- Transaction End Processing Time

  (TOTAL_COMMIT_TIME + TOTAL_ROLLBACK_TIME)
- Load Processing time

  (TOTAL_LOAD_TIME)

Component Processing Time

This category, introduced in DB2 9.7 FixPack1, helps answer the following question: "How much time is spent doing actual work versus wait within a DB2 Component?" Examples:

- Compile Processing time

  (TOTAL_COMPILE_PROC_TIME +
  TOTAL_IMPLICIT_COMPILE_PROC_TIME)
- Section Processing time

  (TOTAL_SECTION_PROC_TIME)
- Sort Processing time

  (TOTAL_SECTION_SORT_PROC_TIME)

  Note that Section Processing time includes Sort Processing time
- Transaction End Processing Time

  (TOTAL_COMMIT_PROC_TIME + TOTAL_ROLLBACK_PROC_TIME)
- Load Processing time

  (TOTAL_LOAD_PROC_TIME)

Using the time spent metrics at other levels, we can get an insight into where we are spending time in DB2. See Figure 4-6 through Figure 4-8 on page 165.

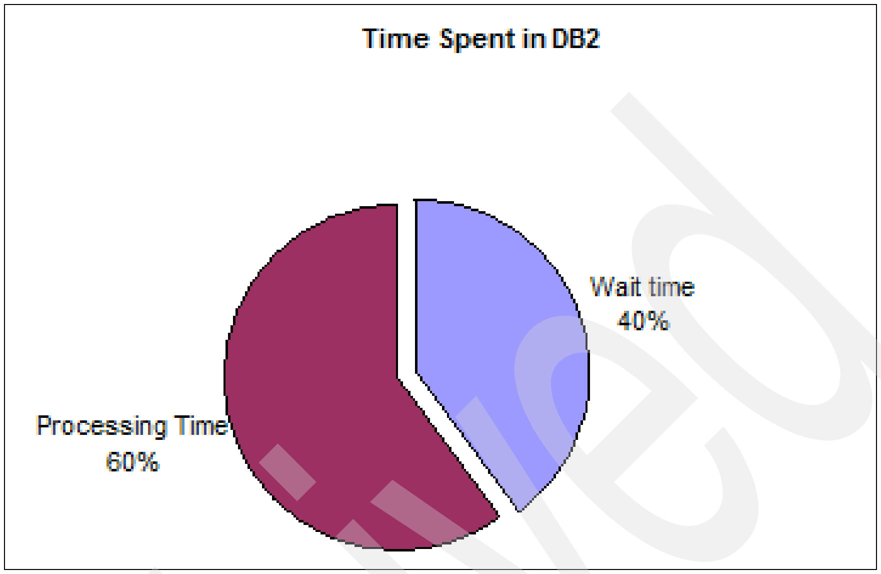


Figure 4-6 Wait time versus processing time with respect to overall request time

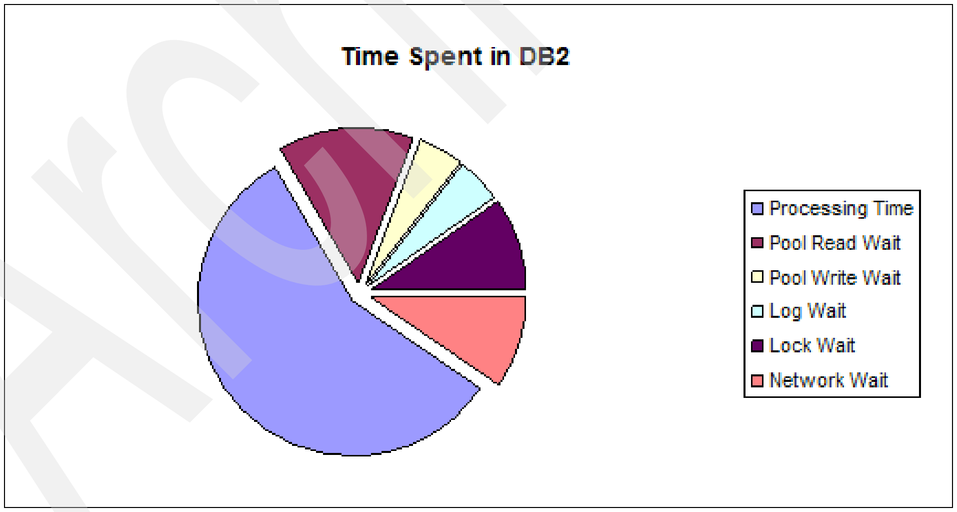


Figure 4-7 Breakup of Total Wait Time

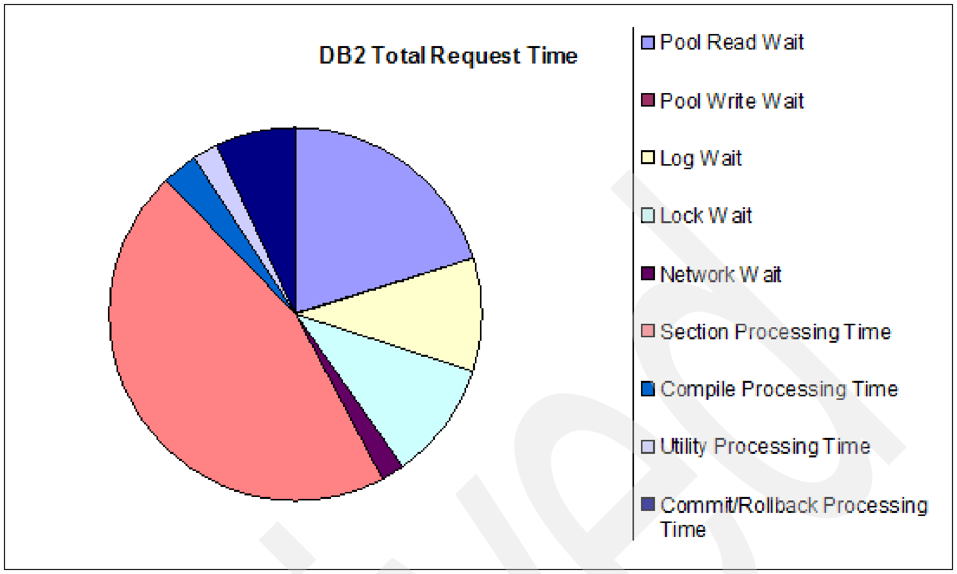


Figure 4-8 Breakup of processing time with component processing time metrics

There is another interesting dimension you can derive from the Time Spent metrics: you can determine how much time is being spent in the stages of DB2 processing using the Component Time metrics. See Figure 4-9.

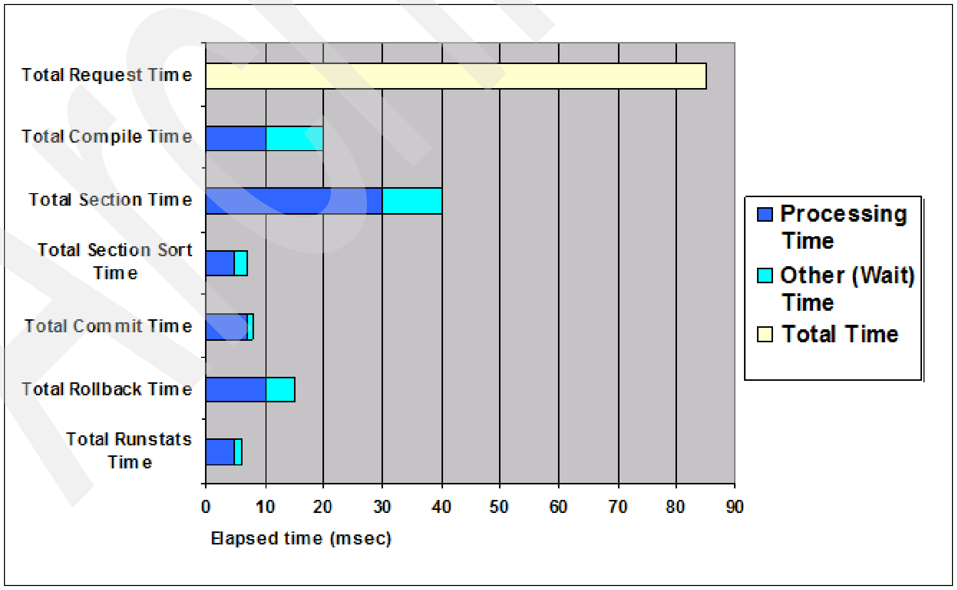


Figure 4-9 Other stages of DB2 processing

You can get these perspectives at the system level as well as at the activity level.

Few important points to keep in mind when looking at the time metrics are:

- All time metrics, except for CPU Time, is in milliseconds. CPU Time is in microseconds.
- All time metrics, including TOTAL_RQST_TIME, is the sum of all work done by agents working on a query and not the elapsed clock time.

  In other words, when we have parallelism between the agents, the values are greater than elapsed time. TOTAL_APP_RQST_TIME gives the actual execution time
- Member column correspond to DBPARTITIONNUM of snapshots.
- Sometimes you might find that the CPU Time is less than the processing time (Total Request time – Total Wait time). This is true in environments where you have many concurrent connections.

  Processing time represents the time an agent is eligible to run. However, it does not necessarily mean it is consuming CPU. It might be waiting on the run-queues for the CPU if there are a large number of threads on the system
- MON_FORMAT_XML_TIMES_BY_ROW is a convenient interface to access the time spent metrics. Visit the following Web page for more information:

  `http://publib.boulder.ibm.com/infocenter/db2luw/v9r7/index.jsp?topic=/com.ibm.db2.luw.sql.rtn.doc/doc/r0056557.html`

Let us see the time-spent metrics in action. A performance problem has been reported for an OLTP application with ~40 concurrent connections. The throughput is low at 10 transactions per second. We use the new administrative views in this example.

What percentage of overall request time in DB2 is spent on processing versus wait?

- Sum up TOTAL_RQST_TIME and TOTAL_WAIT_TIME using MON_GET_CONNECTION table function across all active connections.
- Total Processing Time is calculated as TOTAL_RQST_TIME - TOTAL_WAIT_TIME.
- CLIENT_IDLE_WAIT_TIME is not included in TOTAL_RQST_TIME. A relatively high client idle wait time might indicate that the problem is not in DB2.

The query converts the time spent metrics as a percentage of total request time, as shown in Example 4-24.

Example 4-24 Converting time spent metrics as percentage of total request time

```
WITH TimeSpent AS
( SELECT SUM(TOTAL_WAIT_TIME)AS WAIT_TIME,
SUM(TOTAL_RQST_TIME - TOTAL_WAIT_TIME)  AS PROC_TIME,
SUM(TOTAL_RQST_TIME)AS RQST_TIME,
SUM(CLIENT_IDLE_WAIT_TIME) as client_idle_wait_time
FROM TABLE(MON_GET_CONNECTION(NULL,NULL)) AS METRICS)

SELECT
RQST_TIME,
CASE WHEN RQST_TIME > 0
THEN DEC((FLOAT(WAIT_TIME))/FLOAT(RQST_TIME) * 100,5,2)
ELSE NULL END AS WAIT_PCT ,
CASE WHEN RQST_TIME > 0
THEN DEC((FLOAT(PROC_TIME))/FLOAT(RQST_TIME) * 100,5,2)
ELSE NULL END AS PROC_PCT ,
CLIENT_IDLE_WAIT_TIME
FROM TIMESPENT;
RQST_TIME            WAIT_PCT PROC_PCT CLIENT_IDLE_WAIT_TIME
-------------------- -------- -------- ---------------------
           113200332    40.19    59.80               6959603
```

It shows that 60% of the total request time is spent in processing while the remaining 40% is spent in wait.

What is the breakup of the wait times inside DB2? To find this, we query sysibmadm.mon_db_summary to get summary of the time spent metrics at the database level. See Example 4-25.

Example 4-25 What is the breakup of the wait times inside DB2

```
select RQST_WAIT_TIME_PERCENT, IO_WAIT_TIME_PERCENT,
LOCK_WAIT_TIME_PERCENT , NETWORK_WAIT_TIME_PERCENT from
sysibmadm.mon_db_summary
RQST_WAIT_TIME_PERCENT   IO_WAIT_TIME_PERCENT   LOCK_WAIT_TIME_PERCENT
NETWORK_WAIT_TIME_PERCENT
---------------------- --------------------     ----------------------
-----------------
   38.26              59.76                 18.50
5.01
```

This indicates that 60% of Wait time is spent in I/O Wait. Because OLTP is characterized by synchronous reads, it is normal to see the majority of wait time spent in Synchronous Buffer pool Data/Index reads. Is this high enough to be a concern? To answer this, we need to use AIX monitoring tools to place the system performance in the right perspective.

In this case, we are running with 40 client threads. Example 4-26 shows how average CPU use looks in **`vmstat`**.

Example 4-26 vmstat

```
kthr    memory              page              faults        cpu
----- ----------- ------------------------ ------------ -----------
 r  b   avm   fre  re  pi  po  fr   sr  cy  in   sy      cs   us sy id wa
 29 9 4544790 1291740   0   0   0   0    0   0 503 3040 3397 97 3 0 0
```

CPU is saturated at 100% - even if agents are blocked on I/O, we have 40 agents concurrently executing and they are driving the CPU hard. The Runnable queue is quite high. So it looks more like a CPU bottleneck. The next obvious question is where is DB2 spending its time in processing. See Example 4-27.

Example 4-27 Where is DB2 spending its time in processing

```
select (100 - RQST_WAIT_TIME_PERCENT) PROC_TIME_PERCENT,
SECTION_PROC_TIME_PERCENT, SECTION_SORT_PROC_TIME_PERCENT,
COMPILE_PROC_TIME_PERCENT, TRANSACT_END_PROC_TIME_PERCENT from
sysibmadm.mon_db_summaryl
PROC_TIME_PERCENT SECTION_PROC_TIME_PERCENT
SECTION_SORT_PROC_TIME_PERCENT COMPILE_PROC_TIME_PERCENT
TRANSACT_END_PROC_TIME_PERCENT
----------------- -------------------------
------------------------------ -------------------------
------------------------------
            62.37                      2.61
0.00                     49.83                           0.52

  1 record(s) selected.
```

This shows that 50% of our processing time is spent in compiling rather than executing queries. We are constantly re-compiling queries either because the package cache is small or possibly because we are using the same SQL with other literals. We can further check the package cache hit ratio and monitor the SQLs to see if there is a possibility to use parameter markers, instead of literals.

The time-spent metrics, along with vmstat, helped us identify the source of the performance issue.

The time-spent metrics are available at both the system level and activity level. They are also available in the new event monitors introduced in DB2 9.7. More details about the Time Spent Metrics and other monitoring enhancements are available in DB2 9.7 Information Center:

http://publib.boulder.ibm.com/infocenter/db2luw/v9r7/index.jsp?topic=/com.ibm.db2.luw.wn.doc/doc/c0056333.html

4.7 Monitoring tools for AIX

AIX provides a lot of utilities that can help analyze or identify a performance bottleneck when a DB2 application is deployed and while serving several workloads. This section describes these utilities along with useful tools that can be used to identify the bottlenecks. The following performance thresholds were observed in these tools or common AIX utilities. This can be used as reference point to identify performance bottlenecks.

- CPU bound

  When %user + %sys is greater than 80% (Tool used is **`vmstat`**).
- Disk I/O bound

  When %iowait is greater than 40% (Common utility used is **`vmstat`**).
- Application disk bound

  When %tm_act is greater than 70% (common utility used is **`iostat`**).
- Paging space low

  When % paging space used is greater than 70% active (common utility is **`lsps -a`**).
- Paging bound

  When paging logical volumes %tm_act is greater than 30% of the I/O (common utility is iostat) and paging activity is greater than 10 times the number of CPUs (common utility is **`vmstat`**).
- Thrashing

  Rising page outs, CPU wait, and run queue high (common utility is **`vmstat`**).

Tip: Although most of the utilities can be used in various ways to identify performance bottlenecks, nmon can be used as a single or general tool that can provide all of most of the data on a single platform to achieve the same objective.

Details on installing and using nmon can be found in the following Web page:

`http://www.ibm.com/developerworks/aix/library/au-analyze_aix/`

We have a separate section dedicated to nmon to monitor AIX (see “nmon” on page 179) and how it maps to the existing AIX utilities that is used traditionally by performance engineers.

In this section, we focus on using these utilities and tools to help identify the bottlenecks at various Db2 workloads while monitoring DB2.

Virtual memory management statistics, VMSTAT

The **`vmstat`** command is used to report statistics about kernel threads in the run and wait queues, memory, paging, disks, interrupts, system calls, context switches, and CPU activity. If the **`vmstat`** command is used without any options or only with the interval and optionally, the count parameter, such as vmstat 2, then the first line of numbers is an average since system reboot.

Syntax

```
vmstat interval count
```

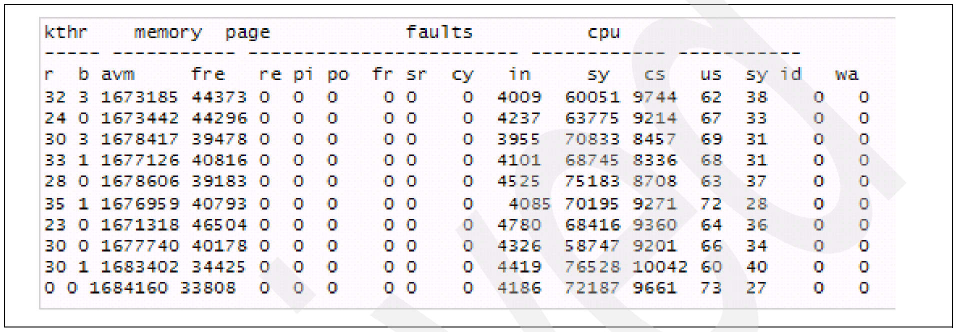
Reporting

The column headings in the report are:

- r: Number of processes on run queue per second
- b: Number of processes awaiting paging in per second
- avm: Active virtual memory pages in paging space
- fre: Real memory pages on the free list
- re: Page reclaims; free, but claimed before being reused
- pi: Paged in (per second)
- po: Paged out (per second)
- fr: Pages freed (page replacement per second)
- sr: Pages per second scanned for replacement
- cy: Complete scans of page table
- in: Device interrupts per second
- sy: System calls per second
- cs: CPU context switches per second
- us: User CPU time percentage
- sys: System CPU time percentage
- id: CPU idle percentage (nothing to do)
- wa: CPU waiting for pending local Disk I/O

The important columns in VMSTAT output to look for are id, sys, wa, and us while monitoring CPU of the AIX system.

Figure 4-10 shows a case of a high CPU issue being identified using the VMSTAT utility. The system is hitting an average of 65% user CPU usage (us column) and 35% system (sy column) CPU usage. Pi and Po column values are equal to 0, so there are no paging issues. The wa column shows there does not seem to be any I/O issues. See Figure 4-10.



```
kthr     memory   page               faults          cpu
----- ----------- ------------------------ ------------ -----------
r  b avm      fre     re pi  po  fr sr   cy    in       sy    cs    us   sy  id   wa
32 3 1673185 44373 0  0   0    0 0    0   4009   60051 9744   62   38     0    0
24 0 1673442 44296 0  0   0    0 0    0   4237   63775 9214   67   33     0    0
30 3 1678417 39478 0  0   0    0 0    0   3955   70833 8457   69   31     0    0
33 1 1677126 40816 0  0   0    0 0    0   4101   68745 8336   68   31     0    0
28 0 1678606 39183 0  0   0    0 0    0   4525   75183 8708   63   37     0    0
35 1 1676959 40793 0  0   0    0 0    0    4085  70195 9271   72   28     0    0
23 0 1671318 46504 0  0   0    0 0    0   4780   68416 9360   64   36     0    0
30 0 1677740 40178 0  0   0    0 0    0   4326   58747 9201   66   34     0    0
30 1 1683402 34425  0  0   0    0 0    0   4419   76528 10042 60   40     0    0
0  0 1684160 33808  0  0   0    0 0    0   4186   72187 9661   73   27     0    0
```

Figure 4-10 vmstat output

The way to interpret VMSTAT for CPU usage in a virtualization environment is a bit different from the traditional way. The CPU usage in a LPAR is dependent on the type of partitioning used and type of mode. For details on this, see the following Web page:

http://www.ibm.com/developerworks/wikis/display/WikiPtype/Virtualization+Concepts

The VMSTAT snapshot in Example 4-28 on page 172 shows the system monitoring activity on an uncapped micro portioning environment with initial CPU entitlement as 0.4, highlighted as “ENT” in the output. With shared pools and LPAR micro partitioning being enabled, note two additional columns as PC and EC shown in VMSTAT output.

PC represents the physical consumed processors and EC represents percentage of entitlement consumed. Example 4-28 on page 172 shows the initial entitlement is 0.4 CPU with physical consumed up to 2.57 CPU being used. In other words, the user and system cpu usage is relative to physical consumed and initial entitlement values.

EC represents entitled capacity. EC represents entitled capacity. This value is related to the initial CPU entitlement. Depending on the initial CPU entitlement, the entitled capacity shows how much % of the initial CPU entitlement is being used.

In Example 4-28, with uncapped partition, the EC value is more than 100 as the initial CPU entitlement is 0.4 CPU and when the physical consumed, PC is more than 0.4 CPU, EC goes more than 100.

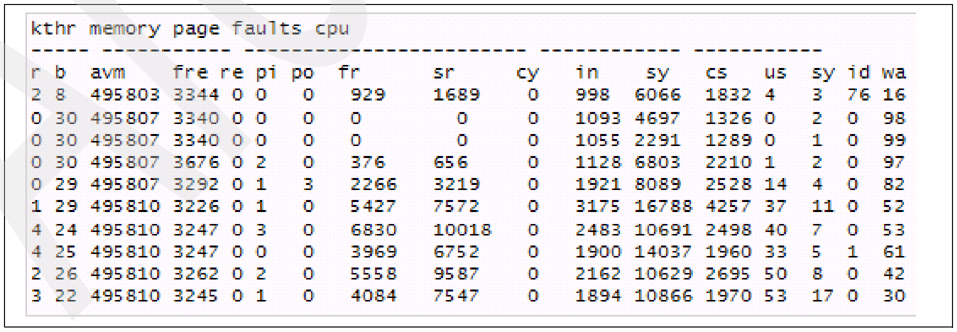
Example 4-28 VMSTAT snapshot

```
System configuration: lcpu=8 mem=16384MB ent=0.40

kthr    memory              page              faults              cpu
----- ----------- ------------------------ ------------ -----------------------
 r  b   avm   fre  re  pi  po  fr   sr  cy  in   sy  cs us sy id wa    pc    ec
 5  0 2493379 965450  0   0   0   0    0   0 426 20400 5727 86 12  2  0  0.66 165.8
 3  0 2493383 965328  0   0   0   0    0   0 541 25948 7791 96  4  1  0  2.57 643.1
 5  0 2493414 965219  0   0   0   0    0   0 486 24601 7551 95  4  1  0  2.17 542.9
 3  0 2493410 965166  0   0   0   0    0   0 410 21398 6209 92  7  1  0  1.25 313.0
 3  0 2493421 965115  0   0   0   0    0   0 376 19853 6593 95  4  1  0  1.79 446.5
 3  0 2493452 964993  0   0   0   0    0   0 547 25833 7323 94  5  1  0  1.91 476.5
 4  0 2493454 964918  0   0   0   0    0   0 466 26966 7848 95  4  1  0  2.12 530.2
 6  0 2493472 964809  0   0   0   0    0   0 554 30049 8407 94  5  1  0  2.22 553.9
```

Tip: VMSTAT can be used to identify memory bottlenecks. If the pi and po values are not zero, it is most likely a memory issue. Occasional values of non zero are OK. For details refer to the following Web page:

http://publib.boulder.ibm.com/infocenter/pseries/v5r3/topic/com.ibm.aix.prftungd/doc/prftungd/mem_perf.htm

The snapshot from VMSTAT in Figure 4-11 shows the wa (waiting on I/O) column to be unusually high. This indicates there might be I/O bottlenecks on the system, which in turn, causes the CPU usage to be inefficient.



```
kthr memory page faults cpu
----- ----------- ------------------------ ------------ -----------
r b  avm    fre re pi po  fr    sr    cy  in    sy    cs   us  sy id wa
2 8  495803 3344 0 0  0   929   1689  0   998   6066  1832 4   3  76 16
0 30 495807 3340 0 0  0   0     0     0   1093  4697  1326 0   2  0  98
0 30 495807 3340 0 0  0   0     0     0   1055  2291  1289 0   1  0  99
0 30 495807 3676 0 2  0   376   656   0   1128  6803  2210 1   2  0  97
0 29 495807 3292 0 1  3   2266  3219  0   1921  8089  2528 14  4  0  82
1 29 495810 3226 0 1  0   5427  7572  0   3175  16788 4257 37  11 0  52
4 24 495810 3247 0 3  0   6830  10018 0   2483  10691 2498 40  7  0  53
4 25 495810 3247 0 0  0   3969  6752  0   1900  14037 1960 33  5  1  61
2 26 495810 3262 0 2  0   5558  9587  0   2162  10629 2695 50  8  0  42
3 22 495810 3245 0 1  0   4084  7547  0   1894  10866 1970 53  17 0  30
```

Figure 4-11 VMSTAT snapshot

Disk IO statistics: IOSTAT

The **`iostat`** command is used to report CPU and I/O statistics for TTY devices, disks, and CD-ROMs. It is used to generate reports that can be used to change the system configuration to better balance the input/output load between physical disks.

Syntax

`iostat interval count`

Reporting columns

Reporting columns are:

- %tm_act

  Reports back the percentage of time that the physical disk was active or the total time of disk requests.

- Kbps

  Reports back the amount of data transferred to the drive in kilobytes.

- tps

  Reports back the number of transfers-per-second issued to the physical disk.

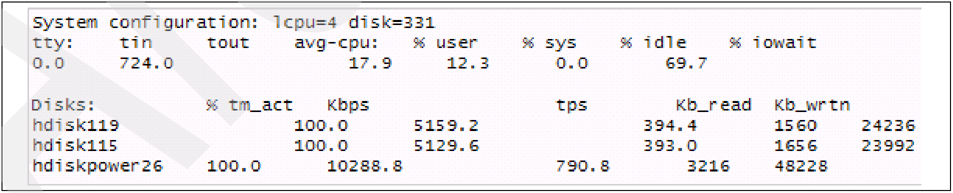
- Kb_read

  Reports back the total data (kilobytes) from your measured interval that is read from the physical volumes.

- Kb_wrtn

  Reports back the amount of data (kilobytes) from your measured interval that is written to the physical volumes.

See Figure 4-12.



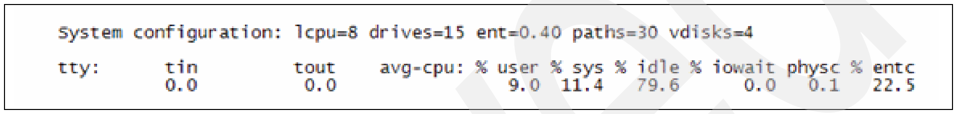
```
System configuration: lcpu=4 disk=331
tty:     tin      tout     avg-cpu:    % user     % sys     % idle     % iowait
0.0      724.0                 17.9      12.3       0.0        69.7

Disks:              % tm_act    Kbps                   tps         Kb_read   Kb_wrtn
hdisk119                    100.0       5159.2                 394.4         1560      24236
hdisk115                    100.0       5129.6                 393.0         1656      23992
hdiskpower26        100.0       10288.8                790.8          3216      48228
```

Figure 4-12 Snaphot from IOSTAT showing greater value than 40% to report a IO bottleneck

To check if there is resource contention, focus on the %tm_act value from the output. An increase in this value, especially more than 40%, implies that processes are waiting for I/O to complete, and there is an I/O issue. For more details on IOSTAT, visit the AIX Information Center:

http://publib.boulder.ibm.com/infocenter/pseries/v5r3/index.jsp?topic=/com.ibm.aix.prftungd/doc/prftungd/iostat_command.htm

In case of an AIX partitioning environment, the IOSTAT header changes. The portion of IOSTAT header output in Figure 4-13 shows an uncapped partitioning LPAR with initial CPU entitlement of 0.40. Remember that physc % is physical consumed and is similar to the VMSTAT PC column discussed in the VMSTAT section.



```
System configuration: lcpu=8 drives=15 ent=0.40 paths=30 vdisks=4

tty:       tin          tout    avg-cpu: % user % sys % idle % iowait physc % entc
           0.0           0.0                9.0  11.4   79.6       0.0   0.1   22.5
```

Figure 4-13 IOSTAT header output

List paging space: lsps

The **lsps** command displays the characteristics of paging spaces, such as the paging space name, physical volume name, volume group name, size, percentage of the paging space used, and whether the space is active or inactive. See Example 4-29.

Example 4-29 List paging space

```
lsps -a
Page Space      Physical Volume   Volume Group    Size %Used Active
Auto  Type
paging01        hdisk0            rootvg        1024MB     3    yes
yes     lv
paging00        hdisk0            rootvg        1024MB     3    yes
yes     lv
hd6             hdisk0            rootvg         512MB     0     no
yes     lv
```

Logical volume manager statistic: lvmstat

The **lvmstat** command helps you identify and remedy hot-spot logical partitions (LPs) within your logical volumes. The gathering of statistics has to be enabled first with the **lvmstat** command for either a logical volume or an entire volume group. When identified, use **migratelp** to move those hot-spot LPs elsewhere.

The following commonly used flags are for the **lvmstat** command:

- –e

  Enables the gathering of statistics about the logical volume.
- -d

  Disables the gathering of statistics
- -l

  Identifies the target logical volume
- -v

  Specifies the target volume group
- -F

  Specifies colon separated output

See the following Web page for details on using LVMSTAT:

http://publib.boulder.ibm.com/infocenter/aix/v6r1/index.jsp?topic=/com.ibm.aix.cmds/doc/aixcmds3/lvmstat.htm

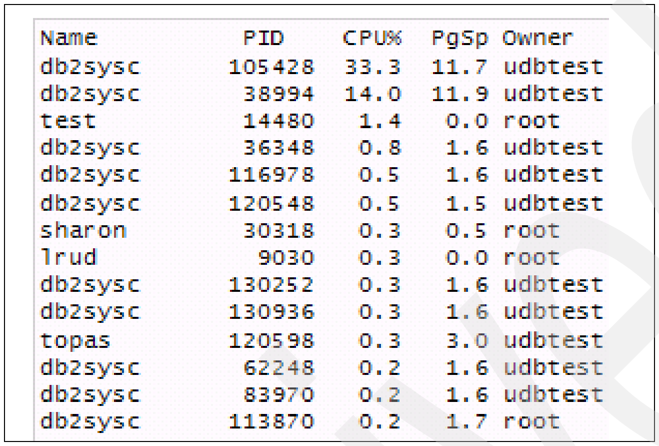
Process State PS

This utility is used to display the status of currently active processes. Apart from using this utility to understand the CPU of a active process, this utility can be used to diagnose or identify performance issues (such as memory leak and hang situations). For details on PS utility usage, see the following Information Center Web page:

http://publib.boulder.ibm.com/infocenter/pseries/v5r3/index.jsp?topic=/com.ibm.aix.prftungd/doc/prftungd/ps_command.htm

TOPAS

This utility is used to get the CPU usage of a certain process in the AIX system. It is used to identify which the processes are consuming most CPU in the AIX system when VMSTAT command whows that there is a CPU bottleneck in the system. See Figure 4-14.



```
Name              PID    CPU%  PgSp Owner
db2sysc        105428    33.3  11.7 udbtest
db2sysc         38994    14.0  11.9 udbtest
test            14480     1.4   0.0 root
db2sysc         36348     0.8   1.6 udbtest
db2sysc        116978     0.5   1.6 udbtest
db2sysc        120548     0.5   1.5 udbtest
sharon          30318     0.3   0.5 root
lrud             9030     0.3   0.0 root
db2sysc        130252     0.3   1.6 udbtest
db2sysc        130936     0.3   1.6 udbtest
topas          120598     0.3   3.0 udbtest
db2sysc         62248     0.2   1.6 udbtest
db2sysc         83970     0.2   1.6 udbtest
db2sysc        113870     0.2   1.7 root
```

Figure 4-14 Snapshot from topas is used to establish that the DB2 engine is contributing to the CPU bottleneck

topas can be used with several options. One common option used in virtual environment is **topas –C**. This command collects a set of metrics from AIX partitions running on the same hardware platform. For details on using TOPAS, see the following Web page:

http://publib.boulder.ibm.com/infocenter/aix/v6r1/topic/com.ibm.aix.cmds/doc/aixcmds5/topas.htm

Netstat

The **netstat** command is used to show network status. It can be used to determine the amount of traffic in the network to ascertain whether performance problems are due to network congestion. See the following Web page for details on its usage:

http://publib.boulder.ibm.com/infocenter/pseries/v5r3/topic/com.ibm.aix.prftungd/doc/prftungd/using_netstat.htm

System Virtual Memory Monitor: SVMON

The **svmon** command is used to capture and analyze a snapshot of virtual memory.

Report

The following are the column headings of the report:

- size

  In pages (4096)
- inuse

  In use
- free

  Not in use (including rmss pages)
- pin

  Pinned (locked by application)
- work

  Pages in working segments
- pers

  Pages in persistent segments
- clnt

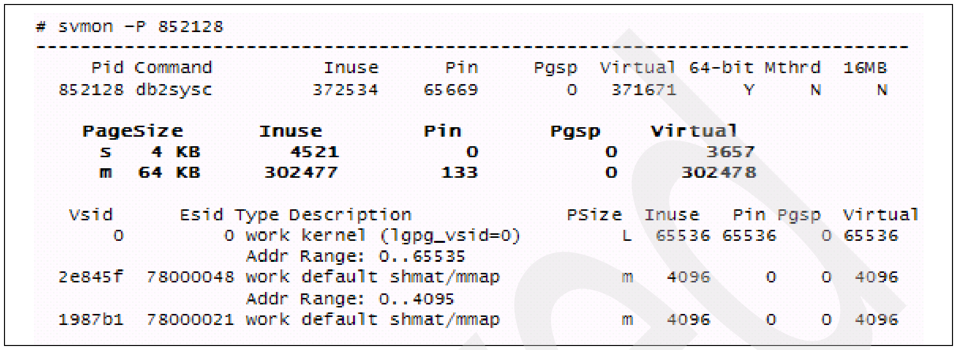
  Pages in client segments
- pg

  space Paging space

`SVMON –G` is a common utility that is used to present global statistics and can help in getting the available memory in system. This is quite effective in finding real memory estimate used by DB2 instance.

Tip: The **`VMSTAT –v`** command can be used to get the similar function.

SVMON can be used to monitor the number of 4-kilobyte and 64-kilobyte page frames on a AIX Power system. For example, to display DB2 process statistics about each page size, use the -P flag with the DB2 process ID (PID) with the svmon command. Figure 4-15 shows the same.



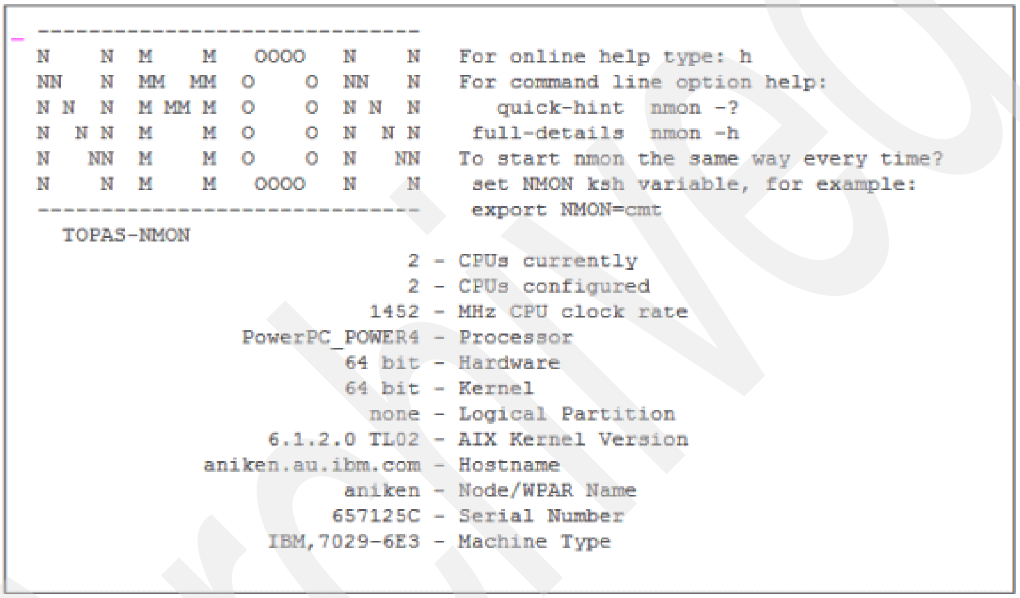
```
# svmon -P 852128
-------------------------------------------------------------------------------
     Pid Command          Inuse      Pin     Pgsp  Virtual 64-bit Mthrd  16MB
  852128 db2sysc         372534    65669        0   371671      Y     N     N

    PageSize        Inuse        Pin       Pgsp    Virtual
      s    4 KB      4521          0          0       3657
      m   64 KB    302477        133          0     302478

   Vsid      Esid Type Description              PSize  Inuse   Pin Pgsp Virtual
       0         0 work kernel (lgpg_vsid=0)         L  65536 65536    0 65536
                   Addr Range: 0..65535
  2e845f  78000048 work default shmat/mmap           m   4096     0    0  4096
                   Addr Range: 0..4095
  1987b1  78000021 work default shmat/mmap           m   4096     0    0  4096
```

Figure 4-15 SVMON

nmon

nmon is an AIX monitoring tool that can used to capture and monitor most of the system related data. The beauty of using **nmon** is by using only a single utility, several system monitoring parameters relevant from DB2 point of view can be analyzed. **nmon** makes the analysis with more ease by generating a spreadsheet automatically, with the help of **nmon** analyzer.

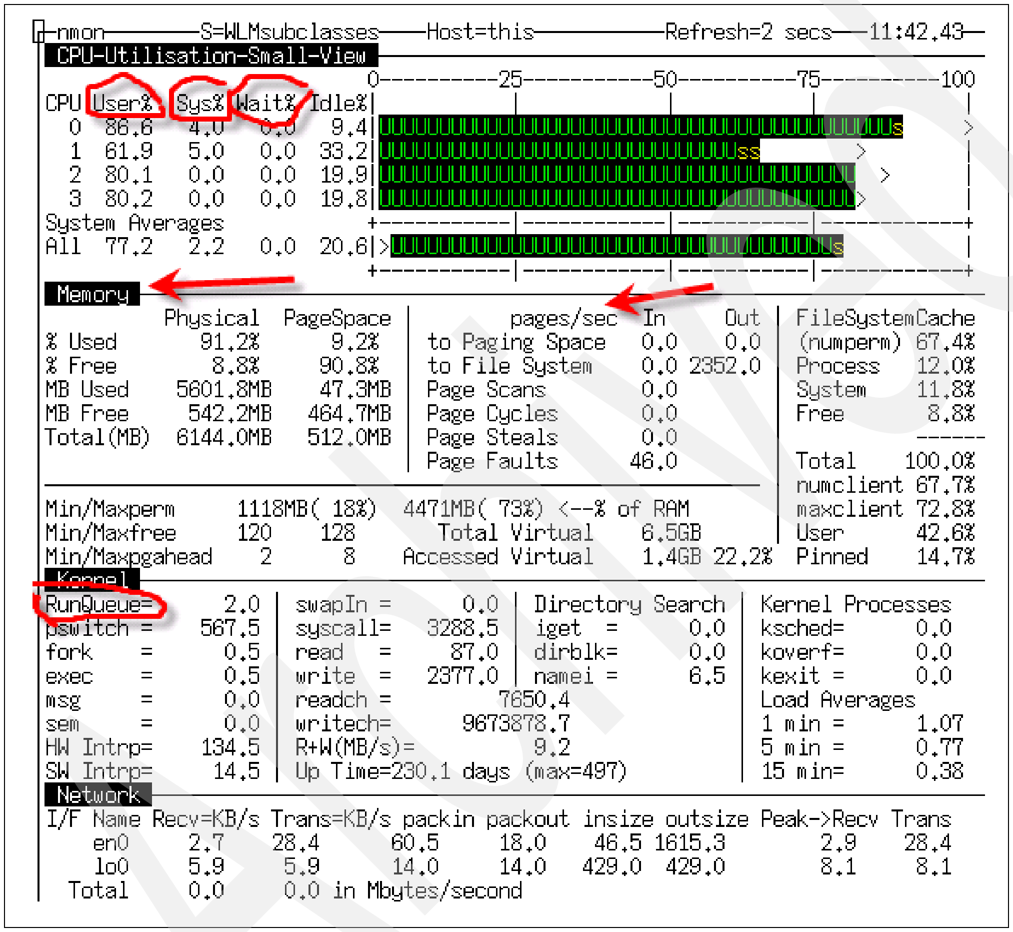
Figure 4-16 shows the first panel seen when **nmon** is used. The first panel shows the AIX level and Processor type details. Use **nmon –h** for full details of usage help.



```
 ------------------------------
 N    N  M    M   OOOO   N    N   For online help type: h
 NN   N  MM  MM  O    O  NN   N   For command line option help:
 N N  N  M MM M  O    O  N N  N      quick-hint  nmon -?
 N  N N  M    M  O    O  N  N N    full-details  nmon -h
 N   NN  M    M  O    O  N   NN   To start nmon the same way every time?
 N    N  M    M   OOOO   N    N    set NMON ksh variable, for example:
 ------------------------------    export NMON=cmt
   TOPAS-NMON
                               2 - CPUs currently
                               2 - CPUs configured
                            1452 - MHz CPU clock rate
                 PowerPC_POWER4 - Processor
                          64 bit - Hardware
                          64 bit - Kernel
                            none - Logical Partition
                    6.1.2.0 TL02 - AIX Kernel Version
                aniken.au.ibm.com - Hostname
                          aniken - Node/WPAR Name
                         657125C - Serial Number
                    IBM,7029-6E3 - Machine Type
```

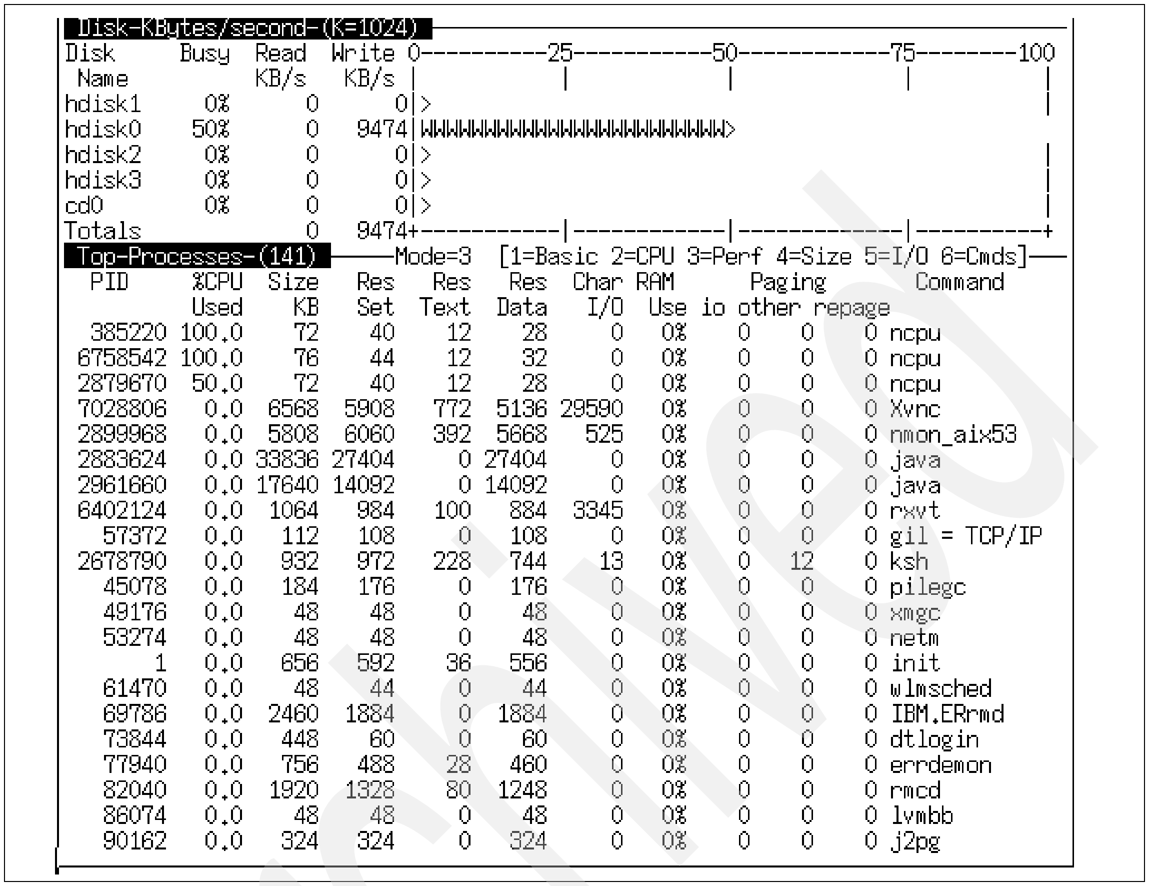
Figure 4-16 AIX and processor details

The **nmon** can be used to display all the AIX system monitoring data on one panel and it can get updated periodically. When running, type h for further help on the options or type q to quit. Figure 4-17 and Figure 4-18 on page 181 show most of the similar system parameters such as CPU, Memory, Paging, Disk, Network and so forth, which we monitored by using several AIX utilities relevant to monitor DB2 application.



```
┌-nmon---------S=WLMsubclasses---Host=this-----------Refresh=2 secs---11:42.43--
 CPU-Utilisation-Small-View
                                0----------25-----------50----------75----------100
 CPU User%  Sys% Wait% Idle%|           |            |           |            |
  0   86.6   4.0   0.0   9.4|UUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUs        >
  1   61.9   5.0   0.0  33.2|UUUUUUUUUUUUUUUUUUUUUUUUUUUUUUss         >           |
  2   80.1   0.0   0.0  19.9|UUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUU  >          |
  3   80.2   0.0   0.0  19.8|UUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUU>           |
 System Averages            +-----------|------------|-----------|------------+
 All  77.2   2.2   0.0  20.6|>UUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUs            |
                            +-----------|------------|-----------|------------+
 Memory
          Physical  PageSpace |        pages/sec  In     Out | FileSystemCache
 % Used      91.2%      9.2%  | to Paging Space   0.0    0.0 | (numperm) 67.4%
 % Free       8.8%     90.8%  | to File System    0.0 2352.0 | Process   12.0%
 MB Used   5601.8MB    47.3MB | Page Scans        0.0        | System    11.8%
 MB Free    542.2MB   464.7MB | Page Cycles       0.0        | Free       8.8%
 Total(MB) 6144.0MB   512.0MB | Page Steals       0.0        |           ------
                              | Page Faults      46.0        | Total    100.0%
 ------------------------------------------------------------ | numclient 67.7%
 Min/Maxperm     1118MB( 18%)  4471MB( 73%) <--% of RAM       | maxclient 72.8%
 Min/Maxfree     120     128      Total Virtual    6.5GB      | User      42.6%
 Min/Maxpgahead    2       8   Accessed Virtual    1.4GB 22.2%  Pinned    14.7%
 Kernel
 RunQueue=       2.0 | swapIn =       0.0 | Directory Search | Kernel Processes
 pswitch =     567.5 | syscall=    3288.5 | iget   =      0.0 | ksched=       0.0
 fork    =       0.5 | read   =      87.0 | dirblk=       0.0 | koverf=       0.0
 exec    =       0.5 | write  =    2377.0 | namei =       6.5 | kexit =       0.0
 msg     =       0.0 | readch =          7650.4                 Load Averages
 sem     =       0.0 | writech=       9673878.7                 1 min =      1.07
 HW Intrp=     134.5 | R+W(MB/s)=           9.2                 5 min =      0.77
 SW Intrp=      14.5 | Up Time=230.1 days (max=497)             15 min=      0.38
 Network
 I/F Name Recv=KB/s Trans=KB/s packin packout insize outsize Peak->Recv Trans
     en0       2.7       28.4    60.5    18.0   46.5 1615.3       2.9   28.4
     lo0       5.9        5.9    14.0    14.0  429.0  429.0       8.1    8.1
   Total       0.0        0.0 in Mbytes/second
```

Figure 4-17 nmon output



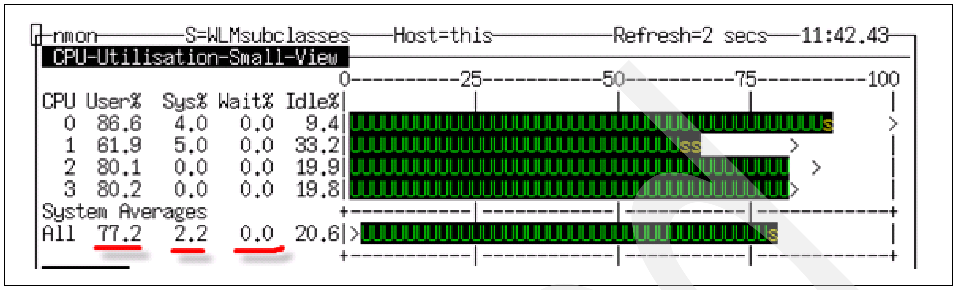
```
 Disk-KBytes/second-(K=1024) ────────────────────────────────────────────────────────
 Disk      Busy  Read  Write 0----------25-----------50------------75--------100
  Name           KB/s   KB/s |          |           |            |          |
 hdisk1     0%      0      0|>                                              |
 hdisk0    50%      0   9474|WWWWWWWWWWWWWWWWWWWWWWWWW>
 hdisk2     0%      0      0|>                                              |
 hdisk3     0%      0      0|>                                              |
 cd0        0%      0      0|>                                              |
 Totals             0   9474+-----------|------------|-------------|----------+
 Top-Processes-(141) ──────Mode=3  [1=Basic 2=CPU 3=Perf 4=Size 5=I/O 6=Cmds]──
   PID     %CPU  Size   Res   Res   Res  Char RAM      Paging       Command
           Used    KB   Set  Text  Data   I/O  Use io other repage
   385220 100.0    72    40    12    28     0   0%    0     0     0 ncpu
  6758542 100.0    76    44    12    32     0   0%    0     0     0 ncpu
  2879670  50.0    72    40    12    28     0   0%    0     0     0 ncpu
  7028806   0.0  6568  5908   772  5136 29590   0%    0     0     0 Xvnc
  2899968   0.0  5808  6060   392  5668   525   0%    0     0     0 nmon_aix53
  2883624   0.0 33836 27404     0 27404     0   0%    0     0     0 java
  2961660   0.0 17640 14092     0 14092     0   0%    0     0     0 java
  6402124   0.0  1064   984   100   884  3345   0%    0     0     0 rxvt
    57372   0.0   112   108     0   108     0   0%    0     0     0 gil = TCP/IP
  2678790   0.0   932   972   228   744    13   0%    0    12     0 ksh
    45078   0.0   184   176     0   176     0   0%    0     0     0 pilegc
    49176   0.0    48    48     0    48     0   0%    0     0     0 xmgc
    53274   0.0    48    48     0    48     0   0%    0     0     0 netm
        1   0.0   656   592    36   556     0   0%    0     0     0 init
    61470   0.0    48    44     0    44     0   0%    0     0     0 wlmsched
    69786   0.0  2460  1884     0  1884     0   0%    0     0     0 IBM.ERrmd
    73844   0.0   448    60     0    60     0   0%    0     0     0 dtlogin
    77940   0.0   756   488    28   460     0   0%    0     0     0 errdemon
    82040   0.0  1920  1328    80  1248     0   0%    0     0     0 rmcd
    86074   0.0    48    48     0    48     0   0%    0     0     0 lvmbb
    90162   0.0   324   324     0   324     0   0%    0     0     0 j2pg
```

Figure 4-18 nmon output

There are separate sections in one panel that represent separate system monitoring parameter. Each of these can be mapped to its corresponding AIX monitoring utility as we discussed in the earlier sections in this chapter. For example, the **Topas** command equivalent can be seen in “Top Processes” section of nmon. Similarly, VMSTAT o/p equivalent can be seen in “CPU-Utilization-Small-View” section of the nmon panel.

The performance threshold also applies to this and can be used as a reference to identify using nmon if a performance bottleneck is encountered or not. For example, if there is a CPU bound situation, check the "CPU-Utilization-Small-View“ section of the nmon output as shown in Figure 4-19 on page 182. If the user% + sys% goes beyond 80%, the system is CPU bound.

Figure 4-19 shows in this case, the user% is 77% and sys% is 2.2%, which is about 80%. The Wait% is zero, signifying that there is CPU waiting for other activity (such as IO) to be completed.



```
-nmon-----------S=WLMsubclasses----Host=this--------------Refresh=2 secs---11:42.43-
 CPU-Utilisation-Small-View
                             0----------25-----------50----------75----------100
CPU User%  Sys% Wait% Idle%|           |           |           |           |
  0  86.6   4.0   0.0   9.4|UUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUs      >
  1  61.9   5.0   0.0  33.2|UUUUUUUUUUUUUUUUUUUUUUUUUUUUUUss        >        |
  2  80.1   0.0   0.0  19.9|UUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUU  >      |
  3  80.2   0.0   0.0  19.8|UUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUU>        |
System Averages            +-----------|-----------|-----------|-----------+
All  77.2   2.2   0.0  20.6|>UUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUUs         |
                           +-----------|-----------|-----------|-----------+
```

Figure 4-19 nmon output

nmon is much simpler to use compared to remembering so many utilities for monitoring AIX. On top of this, nmon analyzer helps to produce a spread sheet automatically, which makes it easier to see the system vitals in the form of charts and graphs. Refer to the following Web page learn more about nmon analyzer:

`http://www.ibm.com/developerworks/aix/library/au-nmon_analyser/`

Figure 4-20 provides a sample CPU usage pattern captured over a period of time by nmon and generated from nmon analyzer. This is equivalent for VMSTAT output shown in Figure 4-19.

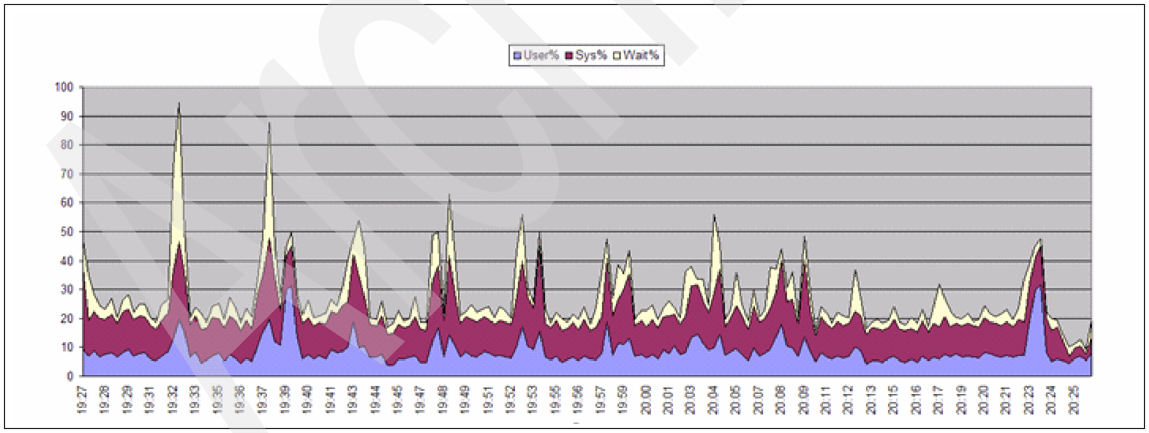


Figure 4-20 Sample CPU usage pattern

These snapshots show that there is good amount of system% activity with % wait time by the CPU's. Figure 4-21 shows the status of the active processes.

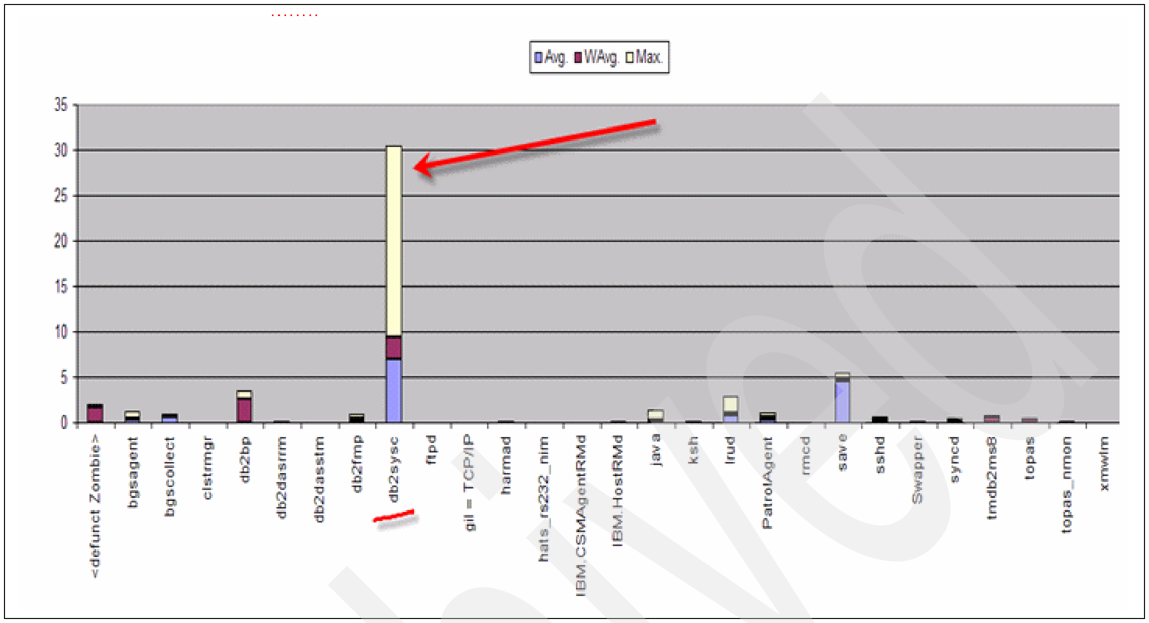


Figure 4-21 Status of the active processes

Figure 4-22 shows a network usage chart captured by nmon and shown by the nmon analyzer. The figure shows that there is a network issue that was getting worse under high load situations. The pattern highlighted in Figure 4-22 shows that there is a network issue due to which there is no network IO activity in a few occasions. This is similar to “netstat” and “tcpdump” analysis data that is required to monitor the network traffic.

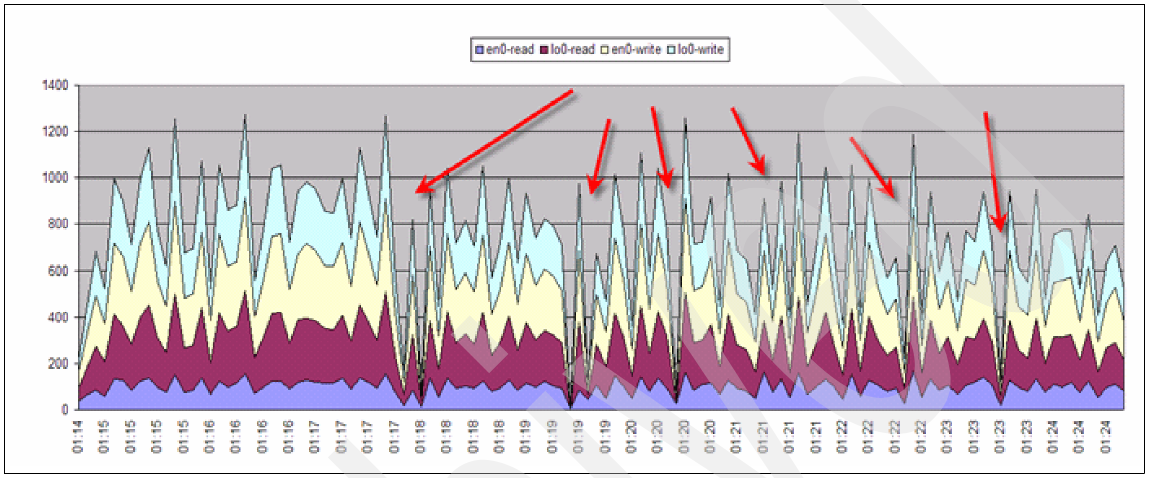


Figure 4-22 Network usage chart

In a virtualization environment, the nmon output is a bit different to the traditional nmon output. Figure 4-23 shows LPAR-related information as shown by nmon, which highlights the initial CPU entitlement value as compared to the virtual CPUs.

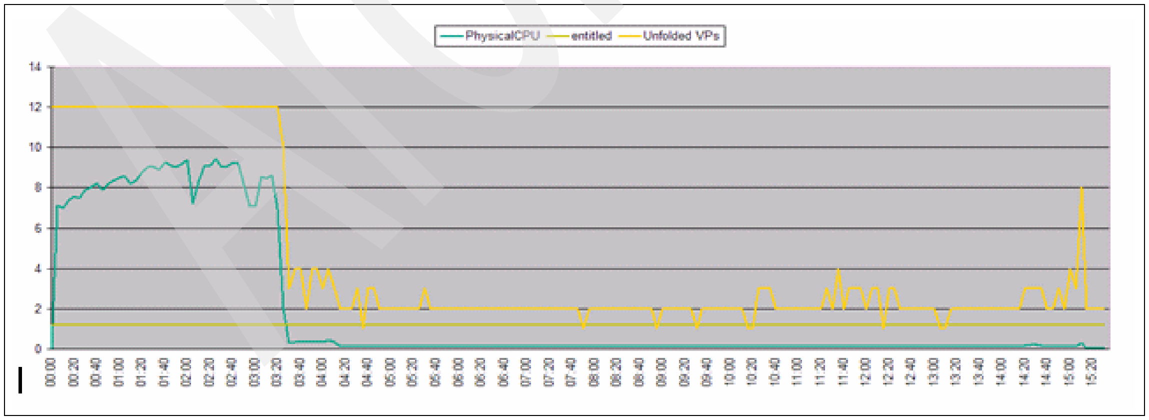


Figure 4-23 nmon graph

The nmon graph in Figure 4-24 shows the CPU behavior of an uncapped partition LPAR with an initial entitlement of 1.2 CPU's. Thus, CPU behavior is again relative to the entitled capacity and the physical processor's that are being consumed.

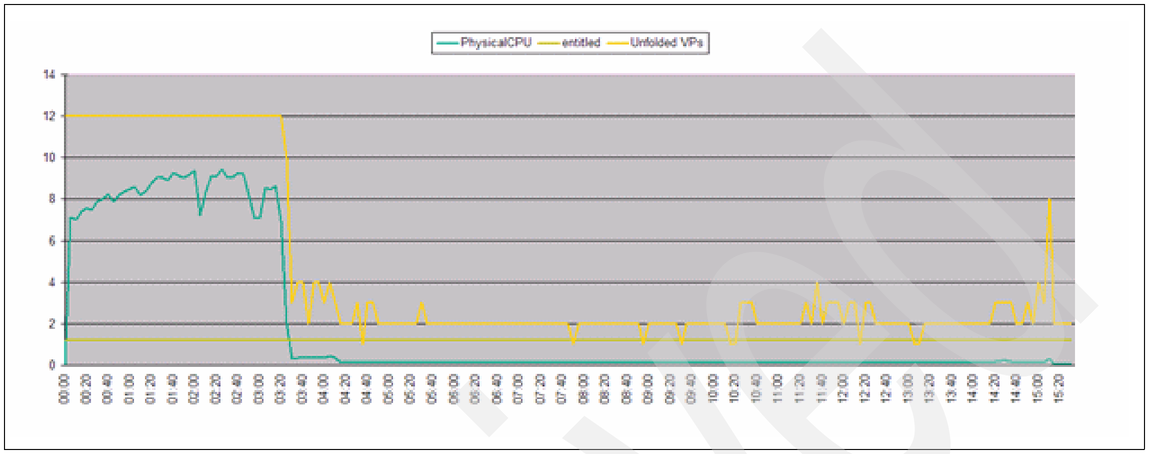


Figure 4-24 nmon graph

4.8 Monitoring scenarios

In this section we cover several real life monitoring scenarios.

4.8.1 Case 1: Alter Tablespace no file system cache

In this scenario, we have applications connecting to DB2 through WebSphere Application Server. These applications, which perform inserts and updates, are experiencing a severe performance degradation. The applications have been ported onto the current system from another where the data was identical to the original environment in which the application performed as expected. Given this situation, one can employ a top-down approach to determine the cause of this degradation.

Let us assume that it has been determined that the problem does not lie in WebSphere Application Server. At this point we start looking into the DB2 and AIX monitoring information. See Example 4-30.

Example 4-30   vmstat output

```
vmstat
kthr    memory              page              faults        cpu
----- ----------- ------------------------ ------------ -----------
 r  b   avm   fre  re  pi  po  fr   sr  cy  in   sy  cs us sy id wa
 2  2 5187657 10998   0   0   0 863  963   0 1693 79115 7175  3 12 67
18
 4  1 5187621 10914   0   0   0 167  202   0 571 4751 4892  0 26 54 19
 1  1 5187559 10532   0   0   0   0    0   0 1433 19293 8553  2  9 73
16
20  0 5187614 10230   0   0   0   0    0   0 656 6603 4921  1 14 63 22
 5  0 5187657 10058   0   0   0 128  146   0 873 6912 5784  1 34 53 12
 8  0 5187656 10107   0   0   0 138  145   0 230 3707 3630  1 25 63 11
 0  0 5187669 10008   0   0   0   1    0   0 403 10491 4611  1 25 60 14
 8  0 5187685  9972   0   0   0  59   73   0 573 7408 5146  1 23 62 14
11  2 5187833  9915   0   0   0 483  540   0 274 38571 3721  3 46 36 15
 7  2 5187709 10598   0   0   0 420  484   0 222 3827 3691  1 34 50 15
```

Starting with the vmstat output it is not apparent as to where exactly the bottleneck can be, even though we see plenty of kernel threads on the run queue and likewise with time spent in wait. The focus now turns to DB2 data, which is collected when the performance degradation was observed. Again, we employ a top-down approach starting with the dbm snapshot and working down to individual queries if needed. This is where the quality of information becomes so important. Based on the quality of this information, one can narrow down and hypothesize about where the problem can lie. In our case a 3x degradation is experienced by insert and update statements for all the applications. Knowing this, we can look into factors that affect more than a query, as opposed to say an access plan for any given query. See Example 4-31.

Example 4-31   Initial Database Manager Snapshot

```
Database Manager Snapshot

Node type                                      = Enterprise Server
Edition with local and remote clients
Instance name                                  = DB2INST1
Number of database partitions in DB2 instance  = 1
Database manager status                        = Active

Private Sort heap allocated                    = 0
Private Sort heap high water mark              = 7050
Post threshold sorts                           = 0
Piped sorts requested                          = 3644324
Piped sorts accepted                           = 3644324

Start Database Manager timestamp               = 11/03/2009
14:29:58.578177
Last reset timestamp                           =
Snapshot timestamp                             = 11/06/2009
20:59:51.493364

Remote connections to db manager               = 63
Remote connections executing in db manager     = 5
Local connections                              = 1
Local connections executing in db manager      = 0
Active local databases                         = 1

High water mark for agents registered          = 93
High water mark for agents waiting for a token = 0
Agents registered                              = 93
Agents waiting for a token                     = 0
Idle agents                                    = 23

Committed private Memory (Bytes)               = 1916928

Switch list for db partition number 0
Buffer Pool Activity Information  (BUFFERPOOL) = ON  11/03/2009
14:29:58.578177
Lock Information                        (LOCK) = ON  11/03/2009
14:29:58.578177
Sorting Information                     (SORT) = ON  11/03/2009
14:29:58.578177
SQL Statement Information          (STATEMENT) = ON  11/03/2009
14:29:58.578177
Table Activity Information             (TABLE) = ON  11/03/2009
14:29:58.578177
Take Timestamp Information         (TIMESTAMP) = ON  11/03/2009
14:29:58.578177
Unit of Work Information                 (UOW) = ON  11/03/2009
14:29:58.578177

Agents assigned from pool                      = 5408
Agents created from empty pool                 = 115
Agents stolen from another application         = 0
High water mark for coordinating agents        = 93
```

```
Max agents overflow                           = 0
Hash joins after heap threshold exceeded      = 0

Total number of gateway connections           = 0
Current number of gateway connections         = 0
Gateway connections waiting for host reply    = 0
Gateway connections waiting for client request = 0
Gateway connection pool agents stolen         = 0
```

Again, we cannot deduce where the problem lies so we proceed with the next set of snapshots, the database snapshots, which are taken 70 seconds apart. Example 4-32 shows the first snapshot.

Example 4-32 Database Snapshot taken at 11/06/2009 20:59:51.556288

```
Database Snapshot

Database name                               = SAMPLE
Database path                               =
/home/db2inst1/db2inst1/NODE0000/SQL00001/
Input database alias                        = SAMPLE
Database status                             = Active
Catalog database partition number           = 0
Catalog network node name                   =
Operating system running at database server= AIX
Location of the database                    = Local
First database connect timestamp            = 11/03/2009 14:31:55.028855
Last reset timestamp                        =
Last backup timestamp                       = 11/06/2009 04:26:58.000000
Snapshot timestamp                          = 11/06/2009 20:59:51.556288

Number of automatic storage paths           = 0

High water mark for connections             = 91
Application connects                        = 3383
Secondary connects total                    = 0
Applications connected currently            = 64
Appls. executing in db manager currently    = 7
Agents associated with applications         = 64
Maximum agents associated with applications= 91
Maximum coordinating agents                 = 91

Locks held currently                        = 202
Lock waits                                  = 5343
Time database waited on locks (ms)          = 2019035
```

```
Lock list memory in use (Bytes)          = 62960
Deadlocks detected                       = 10
Lock escalations                         = 198
Exclusive lock escalations               = 177
Agents currently waiting on locks        = 0
Lock Timeouts                            = 52
Number of indoubt transactions           = 0

Total Private Sort heap allocated        = 0
Total Shared Sort heap allocated         = 0
Shared Sort heap high water mark         = 0
Total sorts                              = 3644404
Total sort time (ms)                     = 3000518
Sort overflows                           = 2089
Active sorts                             = 0

Buffer pool data logical reads           = 573598824
Buffer pool data physical reads          = 100130713
Buffer pool temporary data logical reads   = 268442132
Buffer pool temporary data physical reads  = 6240450
Asynchronous pool data page reads        = 91774524
Buffer pool data writes                  = 15181373
Asynchronous pool data page writes       = 14919461
Buffer pool index logical reads          = 913437418
Buffer pool index physical reads         = 9632933
Buffer pool temporary index logical reads  = 0
Buffer pool temporary index physical reads = 0
Asynchronous pool index page reads       = 1966562
Buffer pool index writes                 = 4859808
Asynchronous pool index page writes      = 4838753
Total buffer pool read time (milliseconds) = 80594078
Total buffer pool write time (milliseconds)= 16800361
Total elapsed asynchronous read time     = 20606257
Total elapsed asynchronous write time    = 15088127
Asynchronous data read requests          = 4176336
Asynchronous index read requests         = 133529
No victim buffers available              = 1817877
LSN Gap cleaner triggers                 = 0
Dirty page steal cleaner triggers        = 0
Dirty page threshold cleaner triggers    = 0
Time waited for prefetch (ms)            = 12712476
Unread prefetch pages                    = 99918
Direct reads                             = 2952905490
Direct writes                            = 20900113
Direct read requests                     = 37386554
Direct write requests                      = 1342561
Direct reads elapsed time (ms)             = 96245210
Direct write elapsed time (ms)             = 28184965
Database files closed                      = 0
Data pages copied to extended storage      = 0
Index pages copied to extended storage     = 0
Data pages copied from extended storage    = 0
Index pages copied from extended storage   = 0

Host execution elapsed time                = 2.346626

Commit statements attempted                = 8665942
Rollback statements attempted              = 645683
Dynamic statements attempted               = 71548695
Static statements attempted                = 5686790
Failed statement operations                = 176729
Select SQL statements executed             = 20914657
Update/Insert/Delete statements executed   = 13160495
DDL statements executed                    = 110
Inactive stmt history memory usage (bytes) = 0

Internal automatic rebinds                 = 0
Internal rows deleted                      = 8588236
Internal rows inserted                     = 1254435
Internal rows updated                      = 0
Internal commits                           = 3383
Internal rollbacks                         = 123
Internal rollbacks due to deadlock         = 71

Rows deleted                               = 30132067
Rows inserted                              = 30161713
Rows updated                               = 2627441
Rows selected                              = 61559544
Rows read                                  = 9500034756
Binds/precompiles attempted                = 0

Log space available to the database (Bytes)= 9928132139
Log space used by the database (Bytes)     = 98875861
Maximum secondary log space used (Bytes)   = 0
Maximum total log space used (Bytes)       = 2927224730
Secondary logs allocated currently         = 0
Log pages read                             = 461948
Log read time (sec.ns)                     = 22.000000004
Log pages written                          = 9057242
Log write time (sec.ns)                    = 8867.000000004
```

```
Number write log IOs                        = 3779013
Number read log IOs                         = 0
Number partial page log IOs                 = 1817223
Number log buffer full                      = 420
Log data found in buffer                    = 8221005
Appl id holding the oldest transaction      = 1268
Log to be redone for recovery (Bytes)       = 45237972
Log accounted for by dirty pages (Bytes)    = 45237972

File number of first active log             = 77623
File number of last active log              = 77657
File number of current active log           = 77623
File number of log being archived           = Not applicable

Package cache lookups                       = 25999275
Package cache inserts                       = 683862
Package cache overflows                     = 0
Package cache high water mark (Bytes)       = 4449189
Application section lookups                 = 76454994
Application section inserts                 = 822653

Catalog cache lookups                       = 5079469
Catalog cache inserts                       = 7738
Catalog cache overflows                     = 0
Catalog cache high water mark               = 0

Workspace Information

 Shared high water mark                     = 0
 Corresponding shared overflows             = 0
 Total shared section inserts               = 0
 Total shared section lookups               = 0
 Private high water mark                    = 5332000
 Corresponding private overflows            = 0
 Total private section inserts              = 822653
 Total private section lookups              = 23453369

Number of hash joins                        = 31086
Number of hash loops                        = 22
Number of hash join overflows               = 47
Number of small hash join overflows         = 1
```

Example 4-33 shows the second snapshot, which was taken 70 seconds after the first snapshot.

Example 4-33 Subsequent database snapshot (11/06/2009 21:01:00.612305)

```
Database Snapshot

Database name                              = SAMPLE
Database path                              =
/db2inst1/db2inst1/NODE0000/SQL00001/
Input database alias                       = SAMPLE
Database status                            = Active
Catalog database partition number          = 0
Catalog network node name                  =
Operating system running at database server= AIX
Location of the database                   = Local
First database connect timestamp           = 11/03/2009 14:31:55.028855
Last reset timestamp                       =
Last backup timestamp                      = 11/06/2009 04:26:58.000000
Snapshot timestamp                         = 11/06/2009 21:01:00.612305

Number of automatic storage paths          = 0

High water mark for connections            = 91
Application connects                       = 3384
Secondary connects total                   = 0
Applications connected currently           = 64
Appls. executing in db manager currently   = 3
Agents associated with applications        = 64
Maximum agents associated with applications= 91
Maximum coordinating agents                = 91

Locks held currently                       = 205
Lock waits                                 = 5345
Time database waited on locks (ms)         = 2021212
Lock list memory in use (Bytes)            = 64960
Deadlocks detected                         = 10
Lock escalations                           = 198
Exclusive lock escalations                 = 177
Agents currently waiting on locks          = 0
Lock Timeouts                              = 52
Number of indoubt transactions             = 7

Total Private Sort heap allocated          = 0
Total Shared Sort heap allocated           = 0
```

```
Shared Sort heap high water mark          = 0
Total sorts                               = 3645714
Total sort time (ms)                      = 3000542
Sort overflows                            = 2089
Active sorts                              = 0

Buffer pool data logical reads            = 573845323
Buffer pool data physical reads           = 100132787
Buffer pool temporary data logical reads  = 268442849
Buffer pool temporary data physical reads = 6240450
Asynchronous pool data page reads         = 91774540
Buffer pool data writes                   = 15181663
Asynchronous pool data page writes        = 14919751
Buffer pool index logical reads           = 913523930
Buffer pool index physical reads          = 9636215
Buffer pool temporary index logical reads  = 0
Buffer pool temporary index physical reads = 0
Asynchronous pool index page reads        = 1966562
Buffer pool index writes                  = 4861015
Asynchronous pool index page writes       = 4839960
Total buffer pool read time (milliseconds) = 80751657
Total buffer pool write time (milliseconds)= 16805469
Total elapsed asynchronous read time      = 20607017
Total elapsed asynchronous write time     = 15093235
Asynchronous data read requests           = 4176352
Asynchronous index read requests          = 133529
No victim buffers available               = 1817877
LSN Gap cleaner triggers                  = 0
Dirty page steal cleaner triggers         = 0
Dirty page threshold cleaner triggers     = 0
Time waited for prefetch (ms)             = 12713235
Unread prefetch pages                     = 99918
Direct reads                              = 2952931034
Direct writes                             = 20904061
Direct read requests                      = 37395065
Direct write requests                     = 1342873
Direct reads elapsed time (ms)            = 96329593
Direct write elapsed time (ms)            = 28385885
Database files closed                     = 0
Data pages copied to extended storage     = 0
Index pages copied to extended storage    = 0
Data pages copied from extended storage   = 0
Index pages copied from extended storage  = 0

Host execution elapsed time               = 2.344887
```

```
Commit statements attempted                 = 8668913
Rollback statements attempted               = 645683
Dynamic statements attempted                = 71567629
Static statements attempted                 = 5687093
Failed statement operations                 = 176797
Select SQL statements executed              = 20919560
Update/Insert/Delete statements executed    = 13162309
DDL statements executed                     = 110
Inactive stmt history memory usage (bytes) = 0

Internal automatic rebinds                  = 0
Internal rows deleted                       = 8588250
Internal rows inserted                      = 1254436
Internal rows updated                       = 0
Internal commits                            = 3384
Internal rollbacks                          = 123
Internal rollbacks due to deadlock          = 71

Rows deleted                                = 30132088
Rows inserted                               = 30163209
Rows updated                                = 2627637
Rows selected                               = 61563275
Rows read                                   = 9502045889
Binds/precompiles attempted                 = 0

Log space available to the database (Bytes)= 9926403251
Log space used by the database (Bytes)      = 100604749
Maximum secondary log space used (Bytes)    = 0
Maximum total log space used (Bytes)        = 2927224730
Secondary logs allocated currently          = 0
Log pages read                              = 461948
Log read time (sec.ns)                      = 22.000000004
Log pages written                           = 9058007
Log write time (sec.ns)                     = 8886.000000004
Number write log IOs                        = 3779594
Number read log IOs                         = 0
Number partial page log IOs                 = 1817491
Number log buffer full                      = 420
Log data found in buffer                    = 8221005
Log to be redone for recovery (Bytes)       = 46550039
Log accounted for by dirty pages (Bytes)    = 46550039

File number of first active log             = 77623
File number of last active log              = 77657
File number of current active log          = 77623
File number of log being archived          = Not applicable

Package cache lookups                      = 26005109
Package cache inserts                      = 683897
Package cache overflows                    = 0
Package cache high water mark (Bytes)      = 4449189
Application section lookups                = 76474230
Application section inserts                = 822754

Catalog cache lookups                      = 5079477
Catalog cache inserts                      = 7739
Catalog cache overflows                    = 0
Catalog cache high water mark              = 0

Workspace Information

 Shared high water mark                    = 0
 Corresponding shared overflows            = 0
 Total shared section inserts              = 0
 Total shared section lookups              = 0
 Private high water mark                   = 5332000
 Corresponding private overflows           = 0
 Total private section inserts             = 822754
 Total private section lookups             = 23459051

Number of hash joins                       = 31131
Number of hash loops                       = 22
Number of hash join overflows              = 47
Number of small hash join overflows        = 1
```

Observations based on these snapshots

Here are our observations:

- Locking does not seem to be an issue as the Time database waited on locks (ms) = 2 seconds.
- There does not appear to be a problem with the sorting either as the difference in 'Total sort time (ms)' is negligible.
- The data page hit ratio is 82%. This is somewhat borderline however, it cannot cause the 3x performance degradation.
- The Index page hit ratio is 98% which is good.

- The amount of time spent in the directly reading and writing pages is quite high though, as shown:

```
Direct reads elapsed time (ms) = 96245210
Direct write elapsed time (ms) = 28184965
Direct reads elapsed time (ms) = 96329593
Direct write elapsed time (ms) = 28385885
Direct read = 84383msec
84 seconds
Direct write = 200920msec
200 seconds
```

This leads us to the suspicion that the bottleneck might be in I/O. Based on this we need to determine which table spaces this I/O is performed against. To find this information we observe the table space snapshot and look for a table space that shows high direct reads and writes as per the previous findings. This reveals that LONG_TS is the table space against which the vast majority of direct reads and writes are spending the most time. See Example 4-34.

Example 4-34 Tablespace snapshot

```
Tablespace name                            = TBSP1
  Tablespace ID                            = 1
  Tablespace Type                          = Database managed space
  Tablespace Content Type                  = Long data only
  Tablespace Page size (bytes)             = 4096
  Tablespace Extent size (pages)           = 32
  Automatic Prefetch size enabled          = No
  Tablespace Prefetch size (pages)         = 32
  Buffer pool ID currently in use          = 1
  Buffer pool ID next startup              = 1
  Using automatic storage                  = No
  Auto-resize enabled                      = No
  File system caching                      = Yes
  Tablespace State                         = 0x'00000000'
   Detailed explanation: Normal
  Total number of pages                    = 27848630
  Number of usable pages                   = 27848576
  Number of used pages                     = 23445856
  Number of pending free pages             = 0
  Number of free pages                     = 4402720
  High water mark (pages)                  = 27407776
  Current tablespace size (bytes)          = 2398838784
  Rebalancer Mode                          = No Rebalancing
  Minimum Recovery Time                    = 06/07/2003 03:58:05.000000
  Number of quiescers                      = 0
  Number of containers                    = 1

  Container Name                          = /db2/SAMPLE/TBSP1
    Container ID                          = 0
    Container Type                        = File (extent sized tag)
    Total Pages in Container              = 27848630
    Usable Pages in Container             = 27848576
    Stripe Set                            = 0
    Container is accessible               = Yes

  Table space map:

   Range  Stripe Stripe  Max         Max  Start  End    Adj.
Containers
   Number Set    Offset  Extent      Page Stripe Stripe
   [   0] [   0]      0  870267  27848575      0 870267   0    1 (0)

  Buffer pool data logical reads          = 17032043
  Buffer pool data physical reads         = 14343
  Buffer pool temporary data logical reads   = 0
  Buffer pool temporary data physical reads  = 0
  Asynchronous pool data page reads       = 0
  Buffer pool data writes                 = 10082
  Asynchronous pool data page writes      = 10053
  Buffer pool index logical reads         = 0
  Buffer pool index physical reads        = 0
  Buffer pool temporary index logical reads  = 0
  Buffer pool temporary index physical reads = 0
  Asynchronous pool index page reads      = 0
  Buffer pool index writes                = 0
  Asynchronous pool index page writes     = 0
  Total buffer pool read time (millisec)  = 300846
  Total buffer pool write time (millisec) = 14893
  Total elapsed asynchronous read time    = 0
  Total elapsed asynchronous write time   = 13715
  Asynchronous data read requests         = 0
  Asynchronous index read requests        = 0
  No victim buffers available             = 34
  Direct reads                            = 636260731
  Direct writes                           = 10830778
  Direct read requests                    = 1669858
  Direct write requests                   = 1219477
  Direct reads elapsed time (ms)          = 49280622
  Direct write elapsed time (ms)          = 26318311
  Number of files closed                  = 0
  Data pages copied to extended storage    = 0
  Index pages copied to extended storage   = 0
  Data pages copied from extended storage  = 0
  Index pages copied from extended storage = 0
```

So, what is contained in this table space? Looking at the Tablespace Content Type we can see that it is Long Data Only. This gives us a good piece of information that the problem has something to do with the reading and writing of long data (for example, clobs/lobs) against this table space. However, consider the output of other monitoring data before reaching any conclusion.

At this point, rather than proceeding with looking into the buffer pools/tables and various other snapshot data, let us focus our attention to tracing from a DB2 and AIX point of view. Using the perfcount option of a DB2 trace, it can be seen in Example 4-35 that DB2 is spending a significant amount of time in the following DB2 functions:

Example 4-35 Perfcount option of a DB2 trace

```
EXECUTIONS  SECONDS     C3
----------- ----------- -----------------------
112         263         sqldCompleteAsyncWrite
743         254         sqldxWriteLobData
757         254         sqldx_diskwrite
743         254         sqldxReplaceLob
721         253         sqldxCreateLob
721         253         sqldxLobCreate
```

DB2 trace: The **db2trc** command controls the trace facility of a DB2 instance or the DB2 Administration Server (DAS). The trace facility records information about operations and puts this information into a readable format.

Even without deep knowledge of DB2, you can estimate that functions sqldxWriteLobData, sqldxReplaceLob, sqldxCreateLob, and sqldxLobCreate appear to be functions that manipulates lob data on disk. This is consistent with the observation regarding direct writes and reads against a long table space.

At this point we know that DB2 is spending a considerable amount of time manipulating lobs against this table space. Is this abnormal or expected if there is a high amount of activity that performs this sort of work? Again, the quality of data comes into play. It is determined that the application does process a significant amount of lob data.

We know there is something suspicious about the I/O against this table space. Now let us go back to the definition of this table space to see if there is something that perhaps is changed or has been overlooked. From the table space snapshot we can see that this table space uses file system caching.

```
File system caching                      = Yes
```

Even though I/O operations against a DMS Long Data table spaces are direct I/O, we might not be taking full advantage of concurrent I/O operations against this table space. At this point it was decided to alter the table space to disallow file system caching, which permits concurrent I/O against the table space containers.

```
db2 alter tablespace TBSP1 NO FILE SYSTEM CACHING
```

Having made this change, the application throughput increased back to the expected level, that is, the 3x performance degradation was alleviated.

4.8.2 Case 2: Alleviating bottlenecks during index creation

This scenario demonstrates what happens as a result of not configuring the various DB2 configuration parameters and table space attributes correctly. This leads to excessive I/O and, coupled with the serialization of the tasks involved, drastically affects the impact of the SQL statement. The scenario is the creation of a simple index. However, as it is shown, by taking simple precautions and monitoring exactly what is happening, we can increase the speed of an index creation by up to four times. Let us use the table space definitions in Example 4-36 and Example 4-37 on page 200 as our example.

Example 4-36 Tablespace definitions

```
[663] db2ins29@eva88 /home2/db2ins29/pmr/b> cat crttbsp.sql

create large tablespace datadms
pagesize 8192
managed by database
using ( file '/pmr/amar/dataconta' 40000 ,
        file '/pmr/amar/datacontb' 40000 ,
        file '/home2/db2ins29/pmr/b/datacontc' 40000 ,
        file '/home2/db2ins29/pmr/b/datacontd' 40000 )
EXTENTSIZE 32
PREFETCHSIZE 172
FILE SYSTEM CACHING
;
create large tablespace indexdms
pagesize 8192
```

```
managed by database
using ( file '/pmr/amar/indexconta' 20000 ,
        file '/pmr/amar/indexcontb' 20000 ,
        file '/home2/db2ins29/pmr/b/indexcontc' 20000 ,
        file '/home2/db2ins29/pmr/b/indexcontd' 20000 )
EXTENTSIZE 32
PREFETCHSIZE 128
FILE SYSTEM CACHING
;
```

Example 4-37 Table spaces are defined on the following physical/logical volumes

```
hdisk0:
LV NAME               LPs   PPs   DISTRIBUTION          MOUNT POINT
pmrlv                 160   160   00..112..48..00..00   /pmr
paging00              40    40    00..00..40..00..00    N/A
db2lv                 20    20    20..00..00..00..00    /db2
data2vgjfslog         1     1     00..00..00..00..01    N/A

hdisk2:
LV NAME               LPs   PPs   DISTRIBUTION          MOUNT POINT
data1vg_jfslogl       1     1     01..00..00..00..00    N/A
homelv                420   420   10..112..111..111..76 /home2
data1vg_raw_lv5       20    20    20..00..00..00..00    N/A
data1vg_raw_lv2       20    20    20..00..00..00..00    N/A
data1vg_raw_lv1       20    20    20..00..00..00..00    N/A
data1vg_raw_lv4       20    20    20..00..00..00..00    N/A
data1vg_raw_lv3       20    20    20..00..00..00..00    N/A
```

During the creation of the index, table data pages on which the index is to be created are fetched into the buffer pool if needed. After this, the columns (keys) on which the index is to be defined are extracted and processed within the sortheap, if possible. If this is a relatively large index, this process of sorting can spill into a temporary table space resulting in I/O. Let us create an index and observe this happening,

Slow index create

The relevant dbm and db configuration parameters that were set for this test are as shown in Example 4-38.

Example 4-38 dbm and db configuration parameters

```
Enable intra-partition parallelism     (INTRA_PARALLEL) = NO
Sort heap thres for shared sorts (4KB) (SHEAPTHRES_SHR) = 5000
Sort list heap (4KB)                         (SORTHEAP) = 256
Number of I/O servers                    (NUM_IOSERVERS) = 1
Number of asynchronous page cleaners   (NUM_IOCLEANERS) = 1
```

The index is created. It takes approximately 16 minutes, as shown in Example 4-39.

Example 4-39 Index created

```
time db2 "create index inxall on mystaff ( ID, NAME, DEPT, JOB, YEARS,
SALARY, COMM )"
DB20000I  The SQL command completed successfully.
real    16m15.86s
user    0m0.02s
sys     0m0.01s
```

Let us observe what happened with regards to the index creation. This scenario is based on DB2 version 9.5, in which the parallelism for the CREATE INDEX statement is enabled regardless of the setting of the intra_parallel configuration parameter. From the **db2pd -edus** output we can see this happening even though the dbm cfg parameters INTRA_PARALLEL is not enabled. This is because there are DB2 sub-agents (db2agntdp) performing work on behalf of the coordinator agent with EDU ID = 1801. See Example 4-40.

Example 4-40 db2pd -edus output

```
EDU ID   TID          Kernel TID     EDU Name
USR          SYS
527     6527         4001859       db2agntdp (SAMPLE  ) 0
0.015435     0.006243
6270     6270         5152897       db2agntdp (SAMPLE  ) 0
29.983732     1.296822
6013     6013         5128343       db2agntdp (SAMPLE  ) 0
35.754063     1.534027
5756     5756         2662549       db2agntdp (SAMPLE  ) 0
36.241472     1.785477
5499     5499         4968605       db2agntdp (SAMPLE  ) 0
42.883862     2.024181
```

```
5242      5242          2338831       db2agntdp (SAMPLE  ) 0
34.613903     1.418642
4985      4985          2830441       db2agntdp (SAMPLE  ) 0
35.269299     1.597505
4728      4728          4718843       db2agent (idle) 0
0.132892     0.180506
4471      4471          3825687       db2evmgi (DB2DETAILDEADLOCK) 0
0.015022     0.033653
4214      4214          2891985       db2wlmd (SAMPLE) 0
0.014463     0.032677
3957      3957          4952109       db2taskd (SAMPLE) 0
0.001486     0.001270
3700      3700          5029949       db2stmm (SAMPLE) 0
0.003971     0.003357
3443      3443          4407437       db2pfchr (SAMPLE) 0
1.159142     4.693933
3186      3186          2797797       db2pclnr (SAMPLE) 0
0.229199     0.199029
2929      2929          3244285       db2dlock (SAMPLE) 0
0.002543     0.001243
2672      2672          2568379       db2lfr (SAMPLE) 0
0.000031     0.000005
2415      2415          3952751       db2loggw (SAMPLE) 0
0.030844     0.053785
2158      2158          4190425       db2loggr (SAMPLE) 0
0.185102     0.159664
1801      1801          3473637       db2agent (SAMPLE) 0
5.549438     0.737307
1543      1543          4280447       db2resync 0
0.000210     0.000091
1286      1286          4165793       db2ipccm 0
0.019325     0.012484
1029      1029          3788911       db2licc 0
0.000110     0.000158
772       772           1343733       db2thcln 0
0.000733     0.000326
515       515           917711        db2aiothr 0
0.034643     0.061765
2         2             5075121       db2alarm 0
0.005181     0.006202
258       258           667855        db2sysc 0
0.106086     0.067665
```

Let us observe the I/O on the disks. We know that hdisk2 contains /home2, which also contains the temporary table space. Initially, we see that hdisk2 is quite busy with regards to the write, as there is not enough room in the sortheap / shared sort area for the index key sorting. Thereafter, when the sorting operation finishes, considerable writes are displayed against hdisk0 and hdisk2, which is a result of writing out the index pages from the temporary table space onto the disk. Example 4-41 is a formatted output of running iostat while the index creation was in progress.

After the sort has spilled into the temporary table space in hdisk2, we can see pages being written to the temporary table space.

Example 4-41 Disk use

```
Disks:        % tm_act     Kbps      tps    Kb_read   Kb_wrtn
hdisk0           0.0       0.0       0.0          0         0
hdisk2           0.0       0.0       0.0          0         0
hdisk0           1.5      23.2       2.9          0        48
hdisk2          65.4    1166.1     100.2        440      1968
hdisk0           1.0       5.9       1.5          0        12
hdisk2          98.8    6848.7     276.7          0     13860
hdisk0           0.0       0.0       0.0          0         0
hdisk2         100.0    5076.0     277.0          0     10152
hdisk0           0.0       0.0       0.0          0         0
hdisk2         100.0    6840.0     259.5          0     13680
hdisk0           0.0       0.0       0.0          0         0
hdisk2         100.0    4354.0     283.5          0      8708
```

This continues for approximately 10 minutes into the index creation, after which we can see that the sorting has finished and the index itself is being written out to the index table space on disk, as shown in Example 4-42.

Example 4-42 Index itself is being written out to the index table space

```
Disks:        % tm_act     Kbps      tps    Kb_read   Kb_wrtn
hdisk0           0.0       0.0       0.0          0         0
hdisk2          99.9    3264.0     363.3          0      6532
hdisk0           0.0       0.0       0.0          0         0
hdisk2          98.8    3349.0     392.8          0      6472
hdisk0          32.6     210.4      37.1          0       420
hdisk2         100.2    1811.4     184.3          0      3616
hdisk0          60.5     400.0      68.5          0       800
hdisk2          59.0     324.0      55.5          0       648
hdisk0          64.5     400.0      68.5          0       800
hdisk2          73.0     440.0      75.0          0       880
```

At the same time, using **db2pd** with the table space output, we can see this happening until the entire index is created. It consists of 23584 pages as shown in Example 4-43.

Example 4-43 db2pd -table spaces output during index creation

```
Tablespace Configuration:
Address            Id    Type Content PageSz ExtentSz Auto Prefetch BufID BufIDDisk
FSC NumCntrs MaxStripe  LastConsecPg Name
0x070000003040CDE0 0     DMS  Regular 8192   4        Yes  4        1     1
Off 1        0          3            SYSCATSPACE
0x070000003040D660 1     SMS  SysTmp  8192   32       Yes  32       1     1
On  1        0          31           TEMPSPACE1
0x0700000031F8A120 2     DMS  Large   8192   32       Yes  32       1     1
Off 1        0          31           USERSPACE1
0x0700000031F8A940 3     DMS  Large   8192   32       Yes  32       1     1
Off 1        0          31           IBMDB2SAMPLEREL
0x0700000031F8B160 4     DMS  Large   8192   32       No   172      1     1
On  4        0          31           DATADMS
0x0700000031F8BDA0 5     DMS  Large   8192   32       No   128      1     1
On  4        0          31           INDEXDMS

Tablespace Statistics:
Address            Id    TotalPgs   UsablePgs  UsedPgs    PndFreePgs FreePgs    HWM
State      MinRecTime NQuiescers
0x0700000031F8BDA0 5     80000      79872      160        0          79712      160
0x00000000 0          0
0x0700000031F8BDA0 5     80000      79872      160        0          79712      160
0x00000000 0          0
0x0700000031F8BDA0 5     80000      79872      192        0          79680      192
0x00000000 0          0
0x0700000031F8BDA0 5     80000      79872      1120       0          78752      1120
0x00000000 0          0
0x0700000031F8BDA0 5     80000      79872      1824       0          78048      1824
0x00000000 0          0
0x0700000031F8BDA0 5     80000      79872      2336       0          77536      2336
0x00000000 0          0
0x0700000031F8BDA0 5     80000      79872      22240      0          57632      22240
0x00000000 0          0
0x0700000031F8BDA0 5     80000      79872      22848      0          57024      22848
0x00000000 0          0
0x0700000031F8BDA0 5     80000      79872      23456      0          56416      23456
0x00000000 0          0
0x0700000031F8BDA0 5     80000      79872      23584      0          56288      23584
0x00000000 0          0
```

Correlating to these events we can also see the shared sort usage using **db2mtrk**. The same can be noticed using **db2pd** with the mempools output. Notice that in Example 4-44 the shared sort increases to 6.8 M. This is roughly the number of sub-agents multiplied by the sortheap allocated by each of them plus a little bit extra for the overflow.

Example 4-44 Shared sort usage using either db2mtrk

```
bph (S16K)  bph (S8K)   bph (S4K)   shsorth     lockh       dbh
   576.0K      448.0K      384.0K      0           640.0K      34.1M
   576.0K      448.0K      384.0K      0           640.0K      34.1M
   576.0K      448.0K      384.0K      0           640.0K      34.1M
   576.0K      448.0K      384.0K      6.3M        640.0K      34.1M
   576.0K      448.0K      384.0K      6.8M        640.0K      34.1M
   576.0K      448.0K      384.0K      6.8M        640.0K      34.1M
   576.0K      448.0K      384.0K      6.8M        640.0K      34.1M
```

Faster index create

For this case we have made the following changes to the scenario. The script used to make the changes is shown in Example 4-45.

Example 4-45 Script

```
db2 update db cfg for sample using SHEAPTHRES_SHR 300000
db2 update db cfg for sample using SORTHEAP 40000
db2 update db cfg for sample using NUM_IOSERVERS AUTOMATIC
db2 update db cfg for sample using NUM_IOCLEANERS AUTOMATIC
db2 alter tablespace datadms PREFETCHSIZE automatic
db2 alter tablespace datadms NO FILE SYSTEM CACHING
```

Let us see what happens when attempting to create the same index. The index is created and it takes approximately four minutes. This is an improvement by a factor of four over the previous index creation. See Example 4-46.

Example 4-46 Create the same index

```
[599] db2ins29@eva88 /home2/db2ins29/pmr/b> time db2 "create index
inxall on mystaff ( ID, NAME, DEPT, JOB, YEARS, SALARY, COMM )"
DB20000I  The SQL command completed successfully.

real    4m22.45s
user    0m0.01s
sys     0m0.01s
```

As per the previous case, let us observe what happened with regards to the fast index creation. The **db2pd -edus** output looks similar to the output. The difference is that there are more prefetchers. This can result in better reads, especially when the data is spread across one or more file systems, as data can be read in parallel. Again, we see the sub-agents doing the majority of the work in terms of USR CPU usage. Likewise the converse applies for writing the data using page cleaners. See Example 4-47.

Example 4-47 db2pd -edus output

```
EDU ID    TID          Kernel TID     EDU Name                               USR          SYS

8740      8740         4718603        db2agntdp (SAMPLE  ) 0                 0.002413     0.000842
8483      8483         3932375        db2agnta (SAMPLE) 0                   31.873649     0.912463
8226      8226         4223029        db2agnta (SAMPLE) 0                   31.151058     0.882912
7969      7969         2977915        db2agnta (SAMPLE) 0                   33.142749     0.924740
7712      7712         3465319        db2agnta (SAMPLE) 0                   29.103719     0.903962
7455      7455         4280451        db2agnta (SAMPLE) 0                   32.160521     0.891415
7198      7198         3473641        db2agnta (SAMPLE) 0                   29.682801     0.876035
6941      6941         4165797        db2agent (idle) 0                      0.040437     0.057003
6684      6684         5128345        db2evmgi (DB2DETAILDEADLOCK) 0         0.005699     0.011232
6427      6427         2662551        db2wlmd (SAMPLE) 0                     0.005559     0.011234
6170      6170         4968607        db2taskd (SAMPLE) 0                    0.001373     0.001133
5913      5913         2338835        db2stmm (SAMPLE) 0                     0.002828     0.002174
5656      5656         1990779        db2pfchr (SAMPLE) 0                    0.057185     0.404891
5399      5399         1986759        db2pfchr (SAMPLE) 0                    0.082485     0.435514
5142      5142         4034585        db2pfchr (SAMPLE) 0                    0.315293     0.909635
4885      4885         3825693        db2pfchr (SAMPLE) 0                    0.408819     1.102220
4628      4628         4190429        db2pclnr (SAMPLE) 0                    0.027033     0.022178
4371      4371         4419759        db2pclnr (SAMPLE) 0                    0.027504     0.022568
4114      4114         4952113        db2pclnr (SAMPLE) 0                    0.025872     0.021212
3857      3857         3788917        db2pclnr (SAMPLE) 0                    0.025435     0.020776
3600      3600         1343739        db2pclnr (SAMPLE) 0                    0.025373     0.020968
3343      3343         5075127        db2pclnr (SAMPLE) 0                    0.025684     0.021113
3086      3086         917717         db2pclnr (SAMPLE) 0                    0.026580     0.021514
2829      2829         5161055        db2dlock (SAMPLE) 0                    0.000968     0.000484
2572      2572         3952759        db2lfr (SAMPLE) 0                      0.000074     0.000026
2315      2315         3629081        db2loggw (SAMPLE) 0                    0.011805     0.026484
2058      2058         4554979        db2loggr (SAMPLE) 0                    0.057497     0.053045
1801      1801         5173331        db2agent (SAMPLE) 0                    5.522430     1.000394
1543      1543         667889         db2resync 0                            0.000154     0.000044
1286      1286         5029953        db2ipccm 0                             0.005281     0.003560
1029      1029         4407441        db2licc 0                              0.000115     0.000173
772       772          2797801        db2thcln 0                             0.000073     0.000028
515       515          3244033        db2aiothr 0                            0.022980     0.044339
2         2            2568383        db2alarm 0                             0.001259     0.001606
258       258          2289899        db2sysc 0                              0.093105     0.022703
```

Let us observe the I/O on the disks. In the previous case there is considerable write activity against hdisk2, which is spilling into the temporary table space. In this case however, there is considerable read activity against both hdisk0 and hdisk2, which is where the table data reside. See Example 4-48 on page 207.

Example 4-48 Observing the I/O on the disks

```
Disks:        % tm_act     Kbps      tps    Kb_read     Kb_wrtn
hdisk0           0.0       0.0       0.0          0           0
hdisk2           0.0       0.0       0.0          0           0
hdisk0          38.9   11732.0     310.3      23552           0
hdisk2          62.3   11477.0     302.4      23040           0
hdisk0          46.0   11520.0     302.5      23040           0
hdisk2          51.5   11584.0     294.5      23168           0
hdisk0          43.0   11123.0     297.2      22232           0
hdisk2          51.5   11090.9     289.7      22168           0
```

Unlike the previous case in which this continued for 10 minutes, in this case this activity only last approximately 3 minutes after which sorting has finished and the index itself is being written out to the index table space on disk. Notice how the disk use is significantly higher and likewise with the throughput of the number of bytes being written. This can be attributed to the greater number of prefetchers. In the previous example this action took 6 minutes as even though the sorting had been performed, the pages were required to be read from the temporary table space into DB2's memory and then written out to disk. In this example, the operation was much quicker because of the greater number of page cleaners and the fact that the pages to be written out are already in memory. See Example 4-49.

Example 4-49 Observing the I/O on the disks

```
Disks:        % tm_act     Kbps      tps    Kb_read   Kb_wrtn
hdisk0          89.0    1208.0     151.0          0      2416
hdisk2          94.5    1452.0     181.5          0      2904
hdisk0          90.0    1204.0     150.5          0      2408
hdisk2          84.5    1412.0     163.0          0      2824
hdisk0          88.1    1248.8     156.1          0      2496
hdisk2          99.1    1450.9     184.1          0      2900
hdisk0          99.4    1336.5     167.1          0      2688
hdisk2          91.0    1330.5     170.5          0      2676
```

At the same time, using **db2pd** with the table space output, we can see this happening until the entire index is created. It consists of 23584 pages. See Example 4-50.

Example 4-50   db2pd -table spaces output during the index creation

```
Tablespace Configuration:
Address            Id    Type Content PageSz ExtentSz Auto Prefetch BufID BufIDDisk
FSC NumCntrs MaxStripe  LastConsecPg Name
0x0700000032A9F700 0     DMS  Regular 8192   4        Yes  4        1     1
Off 1        0          3            SYSCATSPACE
0x0700000031F8A0A0 1     SMS  SysTmp  8192   32       Yes  32       1     1
On  1        0          31           TEMPSPACE1
0x0700000031F8E920 2     DMS  Large   8192   32       Yes  32       1     1
Off 1        0          31           USERSPACE1
0x0700000031F8F140 3     DMS  Large   8192   32       Yes  32       1     1
Off 1        0          31           IBMDB2SAMPLEREL
0x0700000031F78100 4     DMS  Large   8192   32       Yes  128      1     1
Off 4        0          31           DATADMS
0x0700000031F787A0 5     DMS  Large   8192   32       No   128      1     1
On  4        0          31           INDEXDMS

Tablespace Statistics:
Address            Id    TotalPgs   UsablePgs  UsedPgs    PndFreePgs FreePgs    HWM
State      MinRecTime NQuiescers
0x0700000031F787A0 5     80000      79872      160        0          79712      160
0x00000000 0          0
0x0700000031F787A0 5     80000      79872      160        0          79712      160
0x00000000 0          0
0x0700000031F787A0 5     80000      79872      448        0          79424      448
0x00000000 0          0
0x0700000031F787A0 5     80000      79872      3520       0          76352      3520
0x00000000 0          0
0x0700000031F787A0 5     80000      79872      6112       0          73760      6112
0x00000000 0          0
0x0700000031F787A0 5     80000      79872      17792      0          62080      17792
0x00000000 0          0
0x0700000031F787A0 5     80000      79872      20800      0          59072      20800
0x00000000 0          0
0x0700000031F787A0 5     80000      79872      23584      0          56288      23584
0x00000000 0          0
0x0700000031F787A0 5     80000      79872      23584      0          56288      23584
0x00000000 0          0
```

Correlating to the events we can also see the shared sort usage using either **db2mtrk** or **db2pd** with the mempools option. The same can be noticed using **db2pd** with the mempools output. Notice the shared sort increases to approximately 900 MB. This is a little bit less than the number if sub-agents multiplied by the sortheap allocated for each one, that is, 6x 40000 x 4096 = 983040000.

Example 4-51 db2pd with the mempools output

```
   bph (S32K)  bph (S16K)  bph (S8K)   bph (S4K)   shsorth     lockh
   832.0K      576.0K      448.0K      384.0K      0           640.0K
   832.0K      576.0K      448.0K      384.0K      141.4M      640.0K
   832.0K      576.0K      448.0K      384.0K      361.7M      640.0K
   832.0K      576.0K      448.0K      384.0K      910.9M      640.0K
   832.0K      576.0K      448.0K      384.0K      910.9M      640.0K
   832.0K      576.0K      448.0K      384.0K      508.1M      640.0K
   832.0K      576.0K      448.0K      384.0K      508.1M      640.0K
   832.0K      576.0K      448.0K      384.0K      508.1M      640.0K
   832.0K      576.0K      448.0K      384.0K      508.1M      640.0K
   832.0K      576.0K      448.0K      384.0K      508.1M      640.0K
   832.0K      576.0K      448.0K      384.0K      910.9M      640.0K
   832.0K      576.0K      448.0K      384.0K      508.1M      640.0K
   832.0K      576.0K      448.0K      384.0K      2.1M        640.0K
   832.0K      576.0K      448.0K      384.0K      2.1M        640.0K
   832.0K      576.0K      448.0K      384.0K      2.1M        640.0K
```

We can see that changing a few parameters at the database configuration and the table space level can lead to a significant improvement in the amount of time it takes to create an index. By optimizing the database to take full advantage of the multi-threaded architecture and exploiting the memory available, it can alleviate the various bottlenecks that otherwise are left unchanged.

4.8.3 Case 3: Alleviating I/O bottleneck

An OLTP application was tested on the test machine and deployed on the production server. Production server is an 8-way Power 6 processor with 128 GB RAM, while the test server runs on a 4-way Power3 processor with 8 GB RAM. The production server is attached to DS3400 Storage Server with 96 spindles.

Despite production servers having a relatively high configuration compared to the test server, the throughput was almost the same on both production and test server.

Analysis

nmon and DB2 Snapshot were collected during the workload run. Looking at the CPU use seen in Example 4-52 and Figure 4-25, it seems we are hitting an I/O bottleneck. The average I/O wait (measured by wa) is 50%. Despite 100 connections, the run queue is low, at 3 threads.

Example 4-52 CPU use

```
kthr    memory              page              faults              cpu
===============================================================================
 r  b  p   avm    fre    fi  fo  pi  po  fr   sr  in   sy  cs      us sy id wa
 3  0  0 4966174 21500  129  0   4   0   0   0    0 4480 106056     17  7 27 48
```

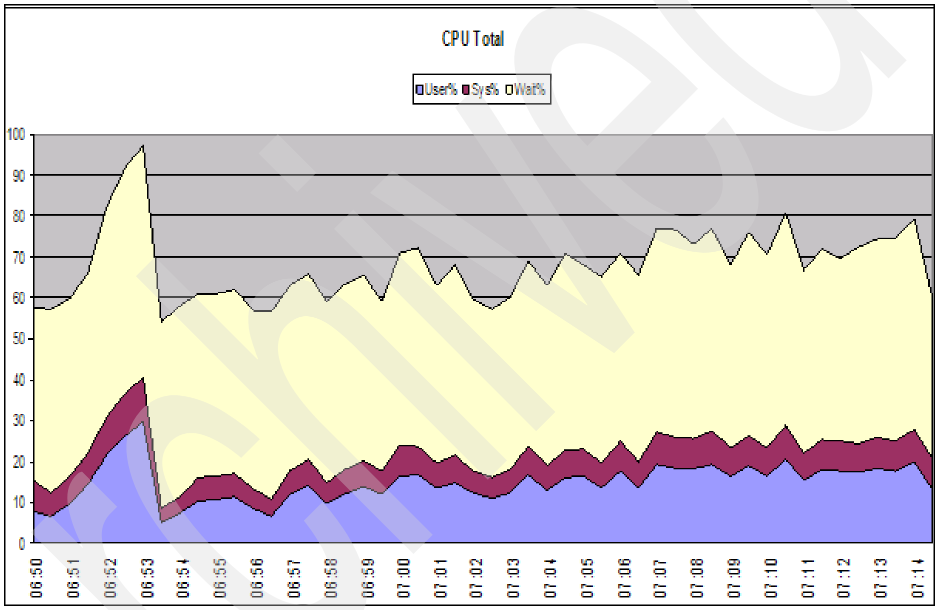


Figure 4-25 CPU total

Let us take a look at statistics related to disk activity in nmon. There are eight hdisks that are fairly active: hdisk23, hdisk24, hdisk25, hdisk26, hdisk27, hdisk28, hdisk29 and hdisk30. See Figure 4-26.

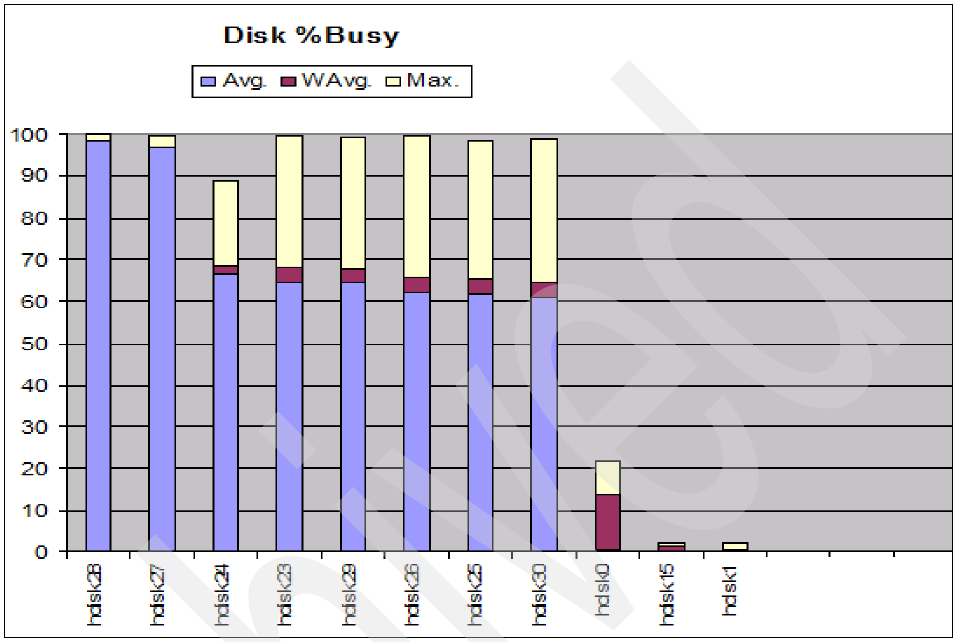


Figure 4-26 Disk%busy

All disks, except for hdisk24, are characterized by read activity, as shown in Figure 4-27.

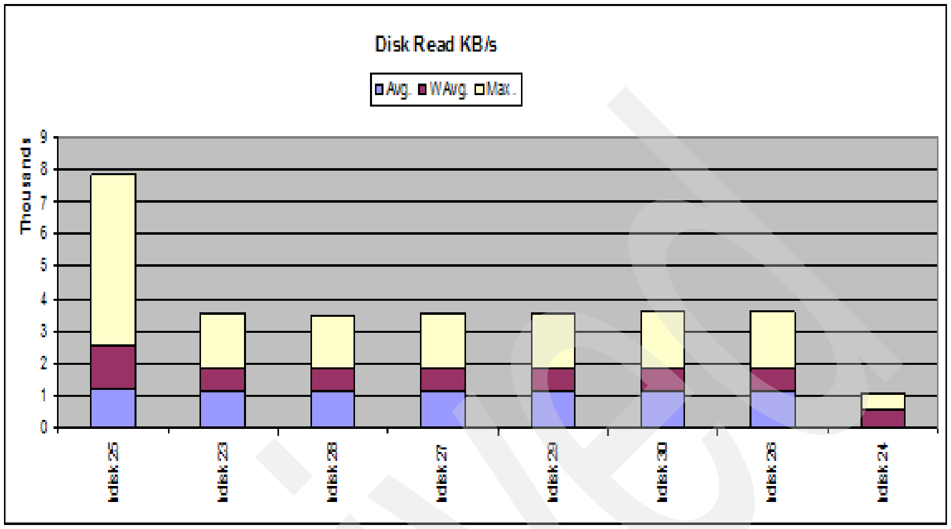


Figure 4-27 Disk Read KB/s

hdisk24 has relatively high write activity, as seen in Figure 4-28.

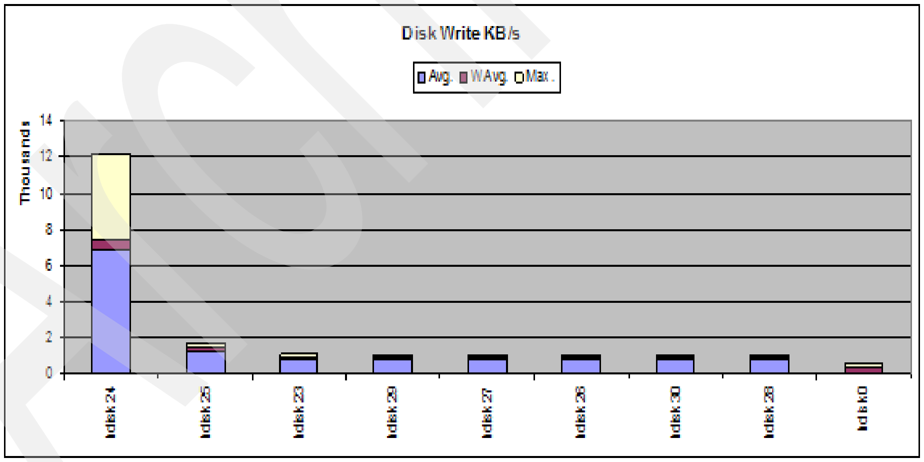


Figure 4-28 Disk write KB/s

The I/O activity looks balanced across the adapters, as shown in Figure 4-29.

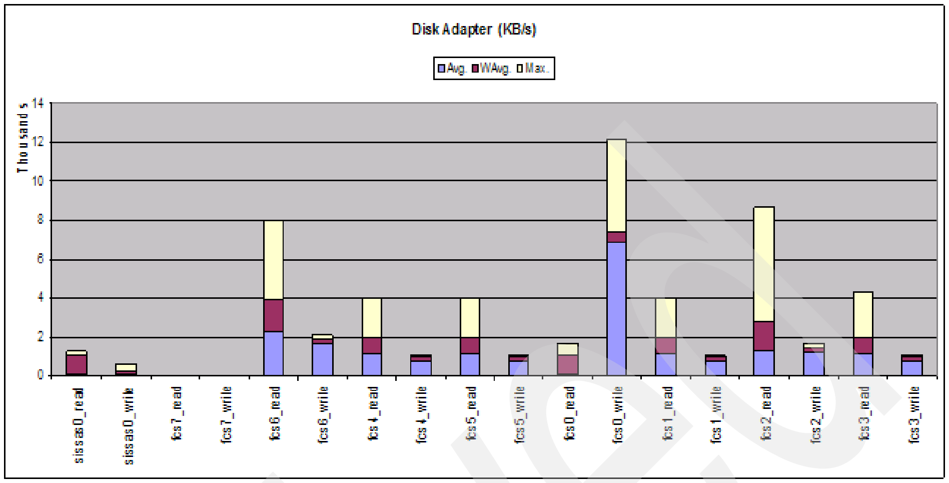


Figure 4-29 The I/O activity looks balanced across the adapters

So far we have not been able to determine any obvious issue with I/O. Let us see what the DB2 Key Performance Indicators suggest. Is it that DB2 is driving many physical I/Os? Let us map the LUNs to file systems. See Example 4-53.

Example 4-53 LUNs to file systems

```
/usr/sbin/lspv -l hdisk23
hdisk23:
LV NAME               LPs     PPs     DISTRIBUTION          MOUNT POINT
DATALV7               900     900     00..819..81..00..00         /DATA7

hdisk24 maps to /LGSHR
hdisk25 maps to /DATA1
hdisk26 maps to /DATA2
hdisk27 maps to /DATA3
hdisk28 maps to /DATA4
hdisk29 maps to /DATA5
hdisk30 maps to /DATA6
```

Next, let us map the file systems to DB2 table space containers and logs, as shown in Example 4-54.

Example 4-54 DB2 table space containers and log

```
SELECT TABLESPACE_ID, TABLESPACE_NAME,CONTAINER_NAME
FROM TABLE(SNAPSHOT_CONTAINER('PRDDB',-1)) AS T

TABLESPACE_ID         TABLESPACE_NAME        CONTAINER_NAME
-------------------------------------------  --------------------------
6                      TS_TXM                    /DATA1/PRDDB1
6                      TS_TXM                    /DATA2/PRDDB1
6                      TS_TXM                    /DATA3/PRDDB1
6                      TS_TXM                    /DATA4/PRDDB1
6                      TS_TXM                    /DATA5/PRDDB1
6                      TS_TXM                    /DATA6/PRDDB1
6                      TS_TXM                    /DATA7/PRDDB1

db2 get db cfg for prddb1  | grep –i ““Path to log files”
Path to log files         = /LGSHR
```

TS_TXM table space is using the DATA[1-7] file systems as containers, while /LGSHR is where the transaction logs of the database are stored. This explains why we saw relatively higher percentage of writes on hdisk24.

Let us see what tables are active in TS_TXM table space. See Example 4-55. The number of rows read is 0. There are no tablescans occurring against the tables.

Example 4-55 Tables that are active in TS_TXM table space

```
SELECT TABLE_NAME, ROWS_READ FROM TABLE(SNAPSHOT_TABLE('TPCC',-1))
WHERE TABLE_NAME IN (SELECT TABNAME FROM SYSCAT.TABLES WHERE
TBSPACEID=6)

TABLE_NAME              ROWS_READ
------------------
-----------------------------------------------------------------------
--
CUSTOMER                  0
ORDERS                    0
WAREHOUSE                 0
DISTRICT                  0
NEW_ORDER                 0
STOCK                     0
HISTORY                   0
ORDER_LINE                0
```

Example 4-56 shows that the bufferpool hit ratio is quite good.

Example 4-56 Buffer pool Hit Ratio

```
SELECT BP_NAME, TOTAL_HIT_RATIO_PERCENT, DATA_HIT_RATIO_PERCENT,
              INDEX_HIT_RATIO_PERCENT
              FROM SYSIBMADM.BP_HITRATIO

BP_NAME
TOTAL_HIT_RATIO_PERCENT DATA_HIT_RATIO_PERCENT INDEX_HIT_RATIO_PERCENT
-------------------------------------------------------------------------------
------------------------------------------------- -----------------------
---------------------- -----------------------
IBMDEFAULTBP
99.39                  97.90                  99.90
```

Let us check the Read Response time in the Snapshot. See Example 4-57. Synchronous read time is important because it blocks the agents and prevents them from doing useful work. This shows up as more idle time in **vmstat**.

Example 4-57 Average read time of 72 ms points to poor I/O response time

```
SELECT BP_NAME, AVERAGE_SYNC_READ_TIME_MS FROM SYSIBMADM.BP_READ_IO

BP_NAME                AVERAGE_SYNC_READ_TIME_MS
-----------------------------------------------------------------------------
IBMDEFAULTBP           72
```

To get details about I/O response time, let us monitor **iostat –D** at regular intervals. We are constantly hitting sqfull condition for all LUNs except hdisk24. sqfull indicates the number of times the service queue becomes full per second. That is, the disk is not accepting any more service requests. On an average, every I/O transfer is spending 14 milliseconds in the wait queue. There might be two possible reasons: Either we have exhausted on the bandwidth or queue_depth of the device is not set properly. Example 4-58 shows hdisk30 use.

Example 4-58 hdisk30 use

```
hdisk30          xfer:  %tm_act      bps      tps      bread      bwrtn
                          90.1      3.5M   854.1       3.5M        0.0
                 read:       rps  avgserv  minserv  maxserv   timeouts      fails
                         854.1      3.0      0.1     28.3          0          0
                write:       wps  avgserv  minserv  maxserv   timeouts      fails
                           0.0      0.0      0.0      0.0          0          0
                queue:  avgtime  mintime  maxtime  avgwqsz    avgsqsz     sqfull
                          14.0      0.0    234.6      0.8        0.2      806.4
```

Next, let us check the queue depth of the devices:

```
lsattr -E -l hdisk30 -a queue_depth
queue_depth 3 Queue DEPTH True
```

They have been left at default of 3. The Storage Administrator informs us that there are 12 physical disks (spindles) under each LUN. The rule of thumb is that at least three outstanding I/Os can wait on each physical disk device to keep it near 100% busy. Because seek optimization is employed in current storage subsystems, having more than three outstanding I/O can improve throughput/response time for I/Os. The queue_depth of the LUNs was set to five times the number of spindles. The num_cmd_elems was also adjusted for the adapters, so that it is at least equal to the sum of queue_depth of all LUNs attached to it.

```
chdev –l hdisk23 -a queue_depth=60
chdev –l fcs0 –a num_cmd_elems=120
```

sqfull was monitored and there were no sqfull observed for the devices. The Average Buffer pool Synchronous Read time reduced to 11.5 milliseconds.

This helped improve the throughput of the OLTP workload by 3x.

4.8.4 Case 4: Memory consumed by the AIX file system cache

The production server hosts Sales Data Warehouse was a well performing system until recently. Performance of the warehouse has degraded with long turnaround time for any workload, impacting business. At times, the server has become unresponsive and had to be rebooted.

Analysis

High level of paging activity is noticed in **`vmstat`** when the response time drops. Take note of the pi and po columns. See Example 4-59.

Example 4-59 vmstat output

```
System configuration: lcpu=4 mem=16384MB ent=2.00
kthr    memory              page              faults              cpu
----- ----------- ------------------------ ------------ -----------------------
 r  b   avm   fre  re  pi  po  fr   sr  cy  in   sy  cs us sy id wa    pc    ec
 0 28 3010156  1552   0  38 1140 1152 1152   0 543  604 11393  7 16 14 62  0.25  24.6
 8 14 3011227  1552   0  70 605 512  512   0 469  444 5595  3 18 29 50  0.22  22.3
 1 10 3012633  1552   0  35 740 768  768   0 476  512 4564  4 11 38 47  0.16  16.0
 0 33 3013989  1552   0  50 724 640  640   0 433  316 11996  5 19 28 48  0.25  24.8
 0 15 3015311  1552   0  46 707 704  704   0 508  559 6221  4 13 36 47  0.19  18.6
 0 14 3016184  1552   0  26 463 448  448   0 394  292 5644  3 10 46 40  0.15  14.8
 0 10 3017089  1552   0  24 475 448  448   0 431  360 4900  3  9 56 33  0.12  12.4
```

Let us use **lsps** to determine the paging space use details. We do have adequate paging space around. However, the "%Used" in **lsps** is increasing over time. See Example 4-60.

Example 4-60 lsps -a

```
Page Space      Physical Volume   Volume Group    Size %Used Active  Auto  Type
paging01        hdisk9            work2vg      32768MB    20   yes   yes    lv
paging03        hdisk8            home2vg      24576MB    20   yes   yes    lv
paging02        hdisk7            home1vg      24576MB    20   yes   yes    lv
paging00        hdisk6            systemvg     16896MB    20   yes   yes    lv
hd6             hdisk0            rootvg        4096MB    20   yes   yes    lv
```

It is important to understand the effect of AIX tunables, minperm and maxperm, with respect to paging. As mentioned in Chapter 2, "AIX configuration" on page 35, protecting computational memory is important for DB2. In other words, we need to favor computational pages over file system cache pages (client and persistent pages).

- Minperm

  If percent of real memory occupied by file pages falls below this level, VMM steals both file and computational pages. Hence, a low value (3%) is recommended for DB2.

- Maxperm

  If percent of real memory occupied by file pages increases beyond this level, VMM steals only file pages. The recommended value for this is 90%.

Let us get more insights the memory usage. The -v option of **vmstat** displays the percentage of real memory being used by other categories of pages

vmstat -v was collected and monitored at regular intervals. The minperm and maxperm are set, as per recommendations. However, numperm and numclient is high at 70%. This suggests that the file system cache is occupying 70% of the real memory. See Example 4-61.

Example 4-61 vmstat -v

```
/usr/bin/vmstat -v
    16318448 memory pages
    15690369 lruable pages
      144213 free pages
           2 memory pools
     1060435 pinned pages
        80.0 maxpin percentage
         3.0 minperm percentage <<- system's minperm setting
        90.0 maxperm percentage <<- system's maxperm setting
        70.9 numperm percentage <<- % of memory having
non-computational pages
    11138217 file pages    <<- No of non-computational pages
         0.0 compressed percentage
           0 compressed pages
        70.5 numclient percentage <<- % of memory containing non-comp
client pages
        90.0 maxclient percentage <<- system 's maxclient setting
    11074290 client pages         <<- No of client pages
           0 remote pageouts scheduled
          36 pending disk I/Os blocked with no pbuf
        3230 paging space I/Os blocked with no psbuf
       21557 filesystem I/Os blocked with no fsbuf
           1 client filesystem I/Os blocked with no fsbuf
           0 external pager filesystem I/Os blocked with no fsbuf
           0 Virtualized Partition Memory Page Faults
        0.00 Time resolving virtualized partition memory page faults
```

nmon also displays the memory statistics. The MEMNEW tab in the nmon spreadsheet displays the details about usage of memory. See Figure 4-30.

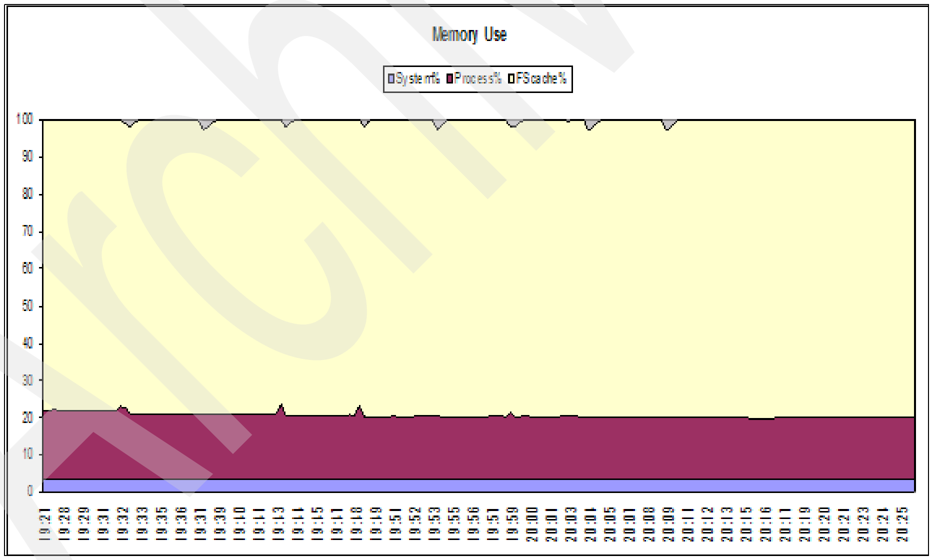


Figure 4-30 nmon output

Next, let us check which process is consuming the file system cache. We use **svmon** to list the top 10 consumers of file system cache.

- -c option of svmon provides details about client memory
- -f about persistent
- -w about computational memory

Example 4-62 svmon output

```
$ svmon -c -Pt 10
-------------------------------------------------------------------------------
--------
     Pid Command          Inuse      Pin     Pgsp  Virtual 64-bit Mthrd
16MB
  160752 db2sysc          8859432        0        0        0      Y
Y     N   =

     PageSize                Inuse        Pin       Pgsp    Virtual
     s    4 KB               10887          0          0          0
     m   64 KB                   0          0          0          0

    Vsid      Esid Type Description              PSize  Inuse     Pin
Pgsp Virtual
  585cb2         - clnt /dev/data_lv8:4098        s   1476572   0    -
-
  779aec         - clnt /dev/data_lv7:33          s   1476400   0    -
-
  e815d3         - clnt /dev/data_lv3:33          s   1476882   0    -
-
  822107         - clnt /dev/data_lv6:33          s   1476360   0    -
-
  21a040         - clnt /dev/data_lv4:33          s   1476770   0    -
-
  8edd1f         - clnt /dev/data_lv5:33          s   1476172   0    -
-
  780ef3        10 clnt text data BSS heap,          s     34      0
-      -
                        /dev/homeDSlv:542222
  3c907b  9fffffff clnt USLA text,/dev/hd2:94283     s     14     0
-      -
   6900e         - clnt /dev/flat_lv:12343       s      6     0    -
-
  1e2c3e         - clnt /dev/flat_lv:12345       s      6     0    -
-
  d2efa7         - clnt /dev/flat_lv:12684       s      3     0    -
-
```

```
  d4f3ab        - clnt /dev/hd9var:620            s     3     0
-       -
  381072        - clnt /dev/hd2:78707             s     2     0
-       -
  cb0195       - clnt /dev/flat_lv:12461      s     2     0     -
-
  db9db4       - clnt /dev/flat_lv:12652      s     1     0     -
-
```

db2sysc is at the top of the list. The InUse column displays the total number of client pages DB2 is using. It alone accounts for 80% of numclients (8859432 pages out of 11074290 client pages).

If you want to check how much space db2sysc is occupying in the file system cache, you can pass db2sysc's pid as an argument (**`svmon -c –P <db2sysc-pid>`**)

Why is DB2 using so much of file system cache? Either it is the DB2 logs (less likely, as this is a data warehouse workload and logging activity is minimal) or file system caching is ON for DB2 Tablespace. File System Caching is not recommended for DB2 containers, except for cases where your table space has LOB data. As DB2 already buffers recently used data in buffer pool, having file system caching ON unnecessarily creates two levels of buffering.

In case you are curious, this is how we can determine files db2sysc is using from the **`svmon`** output. The Description column points to the location and inode of the file. Let us map /dev/data_lv8:4098 to file. See Example 4-63.

Example 4-63 lsfs output

```
lsfs /dev/data_lv8
Name           Nodename   Mount Pt              VFS   Size    Options
Auto Accounting
/dev/data_lv8 --         /data08           jfs2  65536000 rw
no   no

# find /data08 -inum 4098
/data08/ts_work/con08.dat
```

Next, check if file system caching is ON for any DB2 table space using table space snapshot.

```
db2 get snapshot for tablespaces on <dbname>
```

Example 4-64 Tablespace snapshot

```
Tablespace name                              = DATA_TS
  Tablespace ID                              = 0
  Tablespace Type                            = Database managed space
  Tablespace Content Type                    = All permanent data.
Regular table space.
  Tablespace Page size (bytes)               = 4096
  Tablespace Extent size (pages)             = 4
  Automatic Prefetch size enabled            = Yes
  Buffer pool ID currently in use            = 1
  Buffer pool ID next startup                = 1
  Using automatic storage                    = Yes
  Auto-resize enabled                        = Yes
  File system caching                        = Yes
```

File system caching was ON for one table space. After disabling file system cache for the table space, there was no paging observed and that data warehouse performance was back to normal.

In this scenario, we found that DB2 is the culprit. In case you find non-DB2 processes are consuming file system cache, you'd need to determine the cause and rectify it.

4.8.5 Case 5: Disk performance bottleneck

In this scenario, a performance bottleneck related to DB2 on AIX using RAID array disk is encountered when running various workload stress tests. This section focuses on the approach taken to fix this issue and provides a few hints on tuning the related parameters. As the OLTP workload is increased, there is throughput and a 70% increase in response time of the transactions compared to the baseline numbers captured with half the load.

Using the following nmon graph on CPU shows consistent CPU wait%, which shows that the processing is on wait for I/O or disk activity. See Figure 4-31.

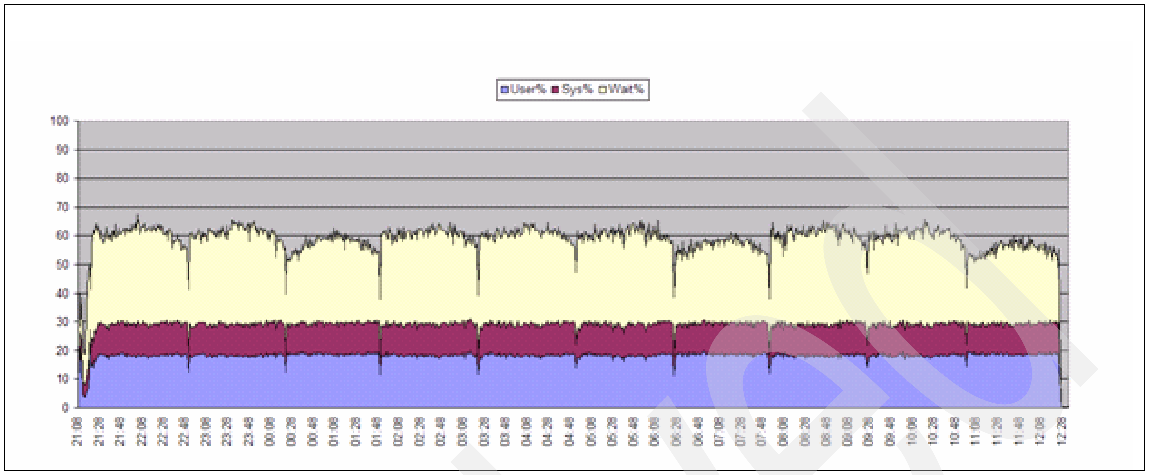


Figure 4-31 nmon output

One thing to check for using the DB2 monitor or snapshots is the buffer pool size usage and hit ratio. Remember, a smaller buffer pool size and lower buffer pool hit ratio can cause high disk I/O usage, which causes the CPU waits to go higher. In this case, the setting used for buffer pool was auto tuning and the DB2 monitoring shows that the DB2 buffer pool hits ratio is 99% which ruled out this being a cause. See Figure 4-32.

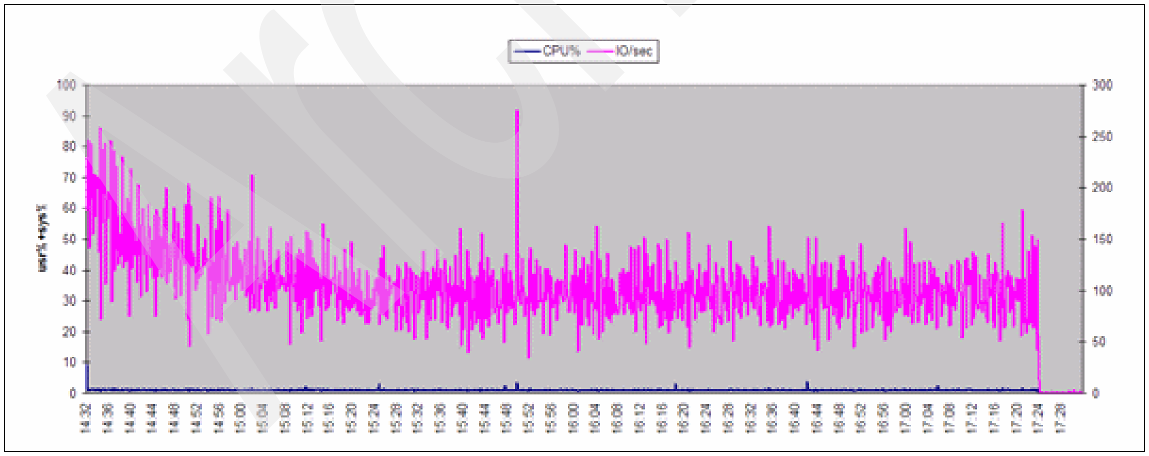


Figure 4-32 nmon chart

From the nmon chart, we can see that an increasing load results in longer disk I/O time (and waits). DB2 does both sequential and random reads. But random reads are done in page sizes, so if you do a read ahead on a random read, the disk actually takes longer to read the page as it transfers more data. Sequential read ahead is normally a good thing for sequential reads, because even if you read one page, the disk reads the rest of it, anticipating your next read request, and although it slows down the first read request, the speed of the subsequent reads improves significantly.

DB2 is normally smart, and when it does sequential reads, it does not do them in small units, but instead reads multiple pages at a time (large I/O). In this case, these large sequential reads see no benefit from read ahead because the requested I/O size is probably similar to the disk stripe size used by read ahead. The OLTP workload in this case, was suspected to be causing such an issue of large waits due the read requests being random I/O as opposed to sequential.

Based on this, the sequential read ahead AIX setting was tuned and experimentation lead to a maximum throughput and reduced the response time such that the negative performance was alleviated. The following AIX settings are under investigation to fix this issue.

- ioo -o minpageahead=Number
- ioo -o maxpageahead=Number

> **Note:** Prior to AIX5.3 version, use, minpgahead and maxpgahead values can be used with options -r and -R respectively in the **`vmtune`** AIX command. For details on using these options, see *AIX 5L Performance Tools Handbook*, SG24-6039.

Using higher values of maxpgahead can be applied in environments where the sequential performance of striped logical volumes is of paramount importance. The initial settings for this scenario were minpageahead as 4 and maxpageahead as 16. However, in this scenario, this value had a negative negative effect which merely added to the IO waits.

To fix this issue, the minpgahead value was varied from 0–4 with maxpgahead varied from 8–16. Subsequent tests were carried out for each of these. The minpgahead being 0 showed maximum improvement in performance for this issue. Remember, a value of 0 for minpgahead can be useful for situation’s of random I/O or workload that has predominantly doing random I/O, as it is totally opposite to the mechanism of sequential read ahead. As a result, the response time difference came down to 10% from 60–70% due to an increase in workload and it resulted in better throughput values.

5

LPAR considerations

As seen in1.5, “Introduction to PowerVM virtualization” on page 21, logical partition (LPAR) is a fundamental building block for the PowerVM virtualization environment. For effective DB2 deployment in the virtualized environment, it is imperative to understand various LPAR features and how to apply them. The chosen LPAR features directly influence functional and non-functional requirements of a datacenter.

In this chapter we discuss at a length about best practices for using the appropriate LPAR type, applying uncapped micro-partitions, multiple shared CPU pools, and how to manage resources and prioritize resource allocation among multiple LPARs hosted on a system.

We cover the following topics in this chapter:

5.1 LPAR planning

Logical partition (LPAR) is an isolated computing domain. Each LPAR needs its own processing capacity, memory, network connection, storage, and operating system. POWER Hypervisor™ encapsulates the system hardware resources to provides levels of abstraction to the LPARs configured on a system. The LPARs running on a system are completely independent from each other. LPARs can have differing amounts of resources and in fact can run any supported operating system (such as AIX, Linux on POWER, and i5/OS®).

The system shown in Figure 5-1 is configured with four LPARs. The LPAR with AIX 6.1 is dedicated processor LPAR. The rest of the LPARs are shared processor LPARs. Each LPAR is running variety of operating systems with differing amount of resources.

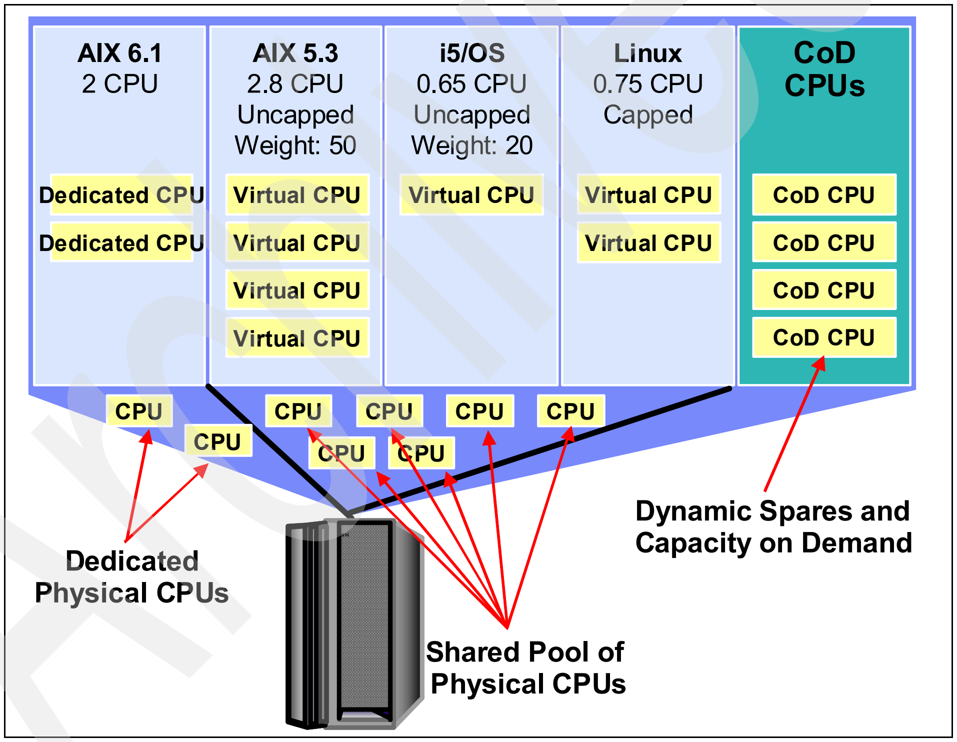


Figure 5-1 LPAR configurations

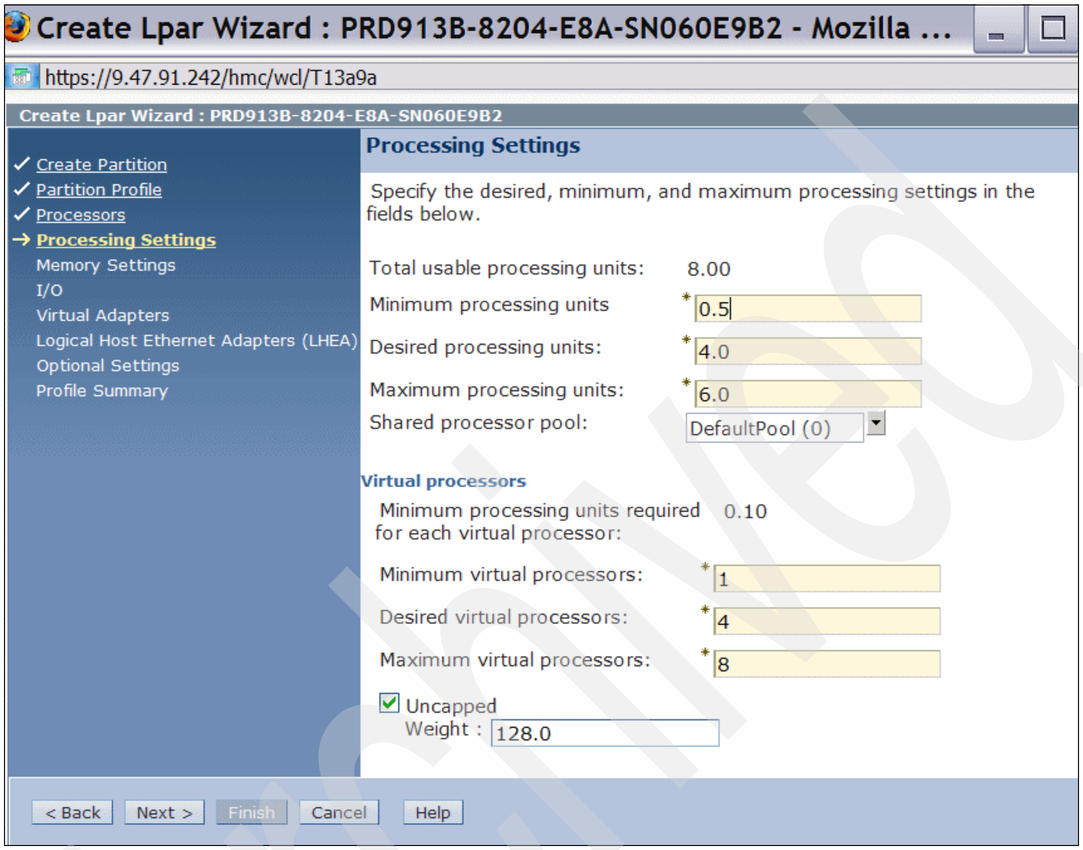
When setting up the system, carefully consider the LPAR requirements, especially the number of planned LPARs with respect to system physical resources, including the number of configured physical processors in a system and number of I/O slots. Depending on model type, size, and installed options such as PowerVM, a POWER6 system can run up to 255 LPARs.For example, if we configure more LPARs than the number of physical processors in a system, a few of the LPARs are shared processor LPARs. Similarly, a limited number of I/O adapters require use of Virtual I/O Servers (VIOSs), because each LPAR needs storage for installing an operating system, such as rootvg for AIX. More information about VIOS, and how to apply it in DB2 environment is available in Chapter 6, "Virtual I/O" on page 241.

After the LPAR is created and the AIX operating system is installed, the user experience is similar to working on a regular host running AIX. However, there are new concepts and setup options related to the virtualization that we discuss in the remainder of the chapter.

5.2 LPAR profile

When a system administrator provisions an LPAR, each LPAR has a set of attributes associated with it. In general, the attributes establish the amount of resources (such as processor, memory, I/O) an LPAR can consume. LPAR resources can be altered even at runtime without requiring a restart of AIX 6L or DB2, using the dynamic logical partition (DLPAR) facility. Any DLPAR operation must honor the specific resource limit set in the LPAR profile. For example, if an LPAR profile dictates that a minimum amount of entitled capacity is 0.50, a DLPAR operation requesting any lower entitled capacity is rejected with an error.

Figure 5-2 shows a LPAR properties information dialog box for processor configuration.


Create Lpar Wizard : PRD913B-8204-E8A-SN060E9B2 - Mozilla ...
https://9.47.91.242/hmc/wcl/T13a9a
Create Lpar Wizard : PRD913B-8204-E8A-SN060E9B2
✓ Create Partition
✓ Partition Profile
✓ Processors
→ Processing Settings
Memory Settings
I/O
Virtual Adapters
Logical Host Ethernet Adapters (LHEA)
Optional Settings
Profile Summary
< Back
Next >
Finish
Cancel
Help

Processing Settings

Specify the desired, minimum, and maximum processing settings in the fields below.

Total usable processing units:	8.00
Minimum processing units	* 0.5
Desired processing units:	* 4.0
Maximum processing units:	* 6.0
Shared processor pool:	DefaultPool (0)

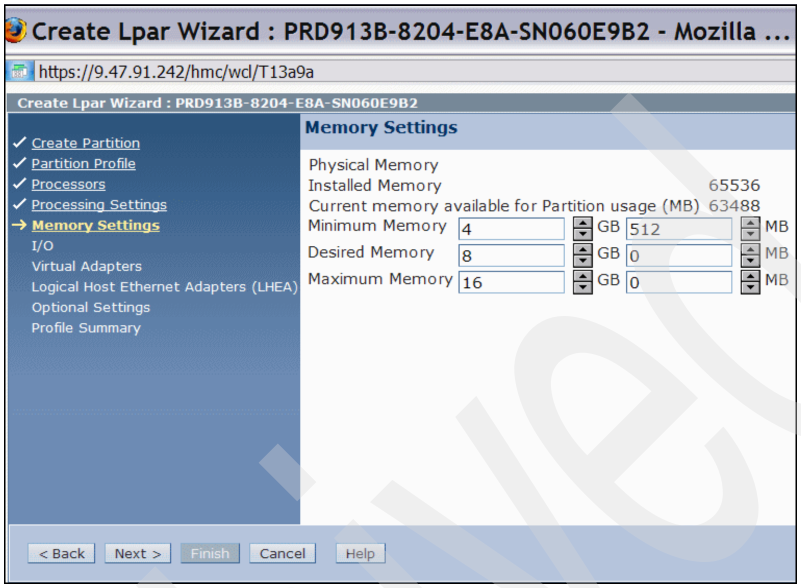
Virtual processors

Minimum processing units required for each virtual processor:	0.10
Minimum virtual processors:	* 1
Desired virtual processors:	* 4
Maximum virtual processors:	* 8

✓ Uncapped
Weight : 128.0

Figure 5-2 LPAR processor minimum, desired, maximum attributes

Figure 5-3 shows LPAR properties information dialog box for memory configuration.


Create Lpar Wizard : PRD913B-8204-E8A-SN060E9B2 - Mozilla ...

https://9.47.91.242/hmc/wcl/T13a9a

Create Lpar Wizard : PRD913B-8204-E8A-SN060E9B2

Memory Settings

Physical Memory

Installed Memory			65536	
Current memory available for Partition usage (MB)			63488	
Minimum Memory	4	GB	512	MB
Desired Memory	8	GB	0	MB
Maximum Memory	16	GB	0	MB

Figure 5-3 LPAR memory minimum, desired, and maximum attributes

Processor and memory resources have minimum, desired, and maximum values in an LPAR profile. The desired value is used when LPAR is started. If the desired amount cannot be fulfilled when starting an LPAR, the POWER Hypervisor attempts to start an LPAR using any amount between the minimum and desired value. If the minimum amount cannot be fulfilled, LPAR fails to start.

5.3 Dynamic logical partitioning

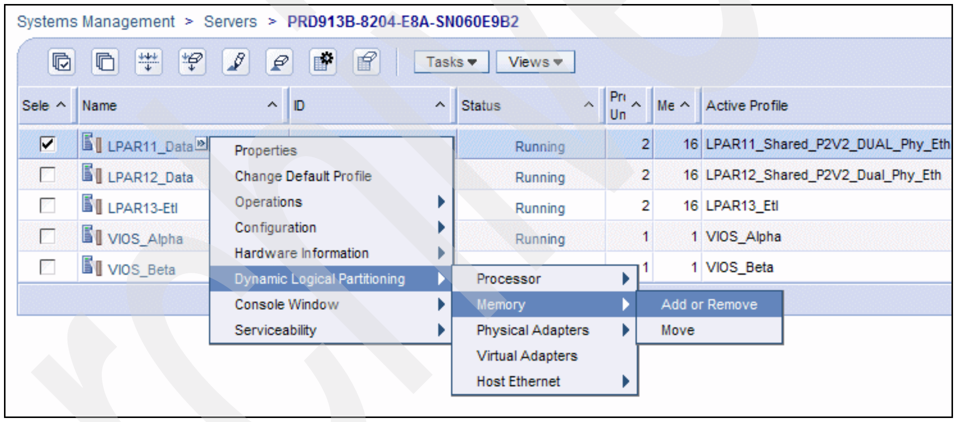
Dynamic logical partitioning (DLPAR) allows altering partition's resources while an LPAR is running, without the need to reboot the system. These partition resources can be added or removed from the LPAR without requiring a restart of the LPAR or DB2. The following resources can be dynamically altered:

- ► Physical processors (for dedicated processor LPARs)
- ► Virtual processors or entitled capacity (for micro-partitions)
- ► Memory
- ► Physical I/O adapters
- ► Virtual I/O adapters

It is an effective tool for dynamic workload management. The traditional workload management strategy is to size and allocate resources for peak workload. In most cases, however, it results in under-used resources when workload demand is not at the peak.

Rather than provisioning resources for peak usage, use the current workload resources requirement to allocate LPAR resource. When the demand changes, use the DLPAR facility to add or remove the required resources. For example, if the current processor consumption decreases, use DLPAR to remove processor resources from an LPAR. Similarly, when AIX 6.1 or DB2 needs more memory (in event of an AIX 6.1 paging activity or low DB2 buffer pool hit ratio, for example), more memory can be added dynamically. Similarly, the resources can be moved from one LPAR to another.

The DLPAR operation is easily accessible using the Hardware Management Console graphical interface. Figure 5-4 shows the menu options to perform the DLPAR operation.



Sele	Name	ID	Status	Pr Un	Me	Active Profile
✓	LPAR11_Data		Running	2	16	LPAR11_Shared_P2V2_DUAL_Phy_Eth
	LPAR12_Data		Running	2	16	LPAR12_Shared_P2V2_Dual_Phy_Eth
	LPAR13-Etl		Running	2	16	LPAR13_Etl
	VIOS_Alpha		Running	1	1	VIOS_Alpha
	VIOS_Beta			1	1	VIOS_Beta

Figure 5-4 Dynamic logical partitioning (DLPAR) menu

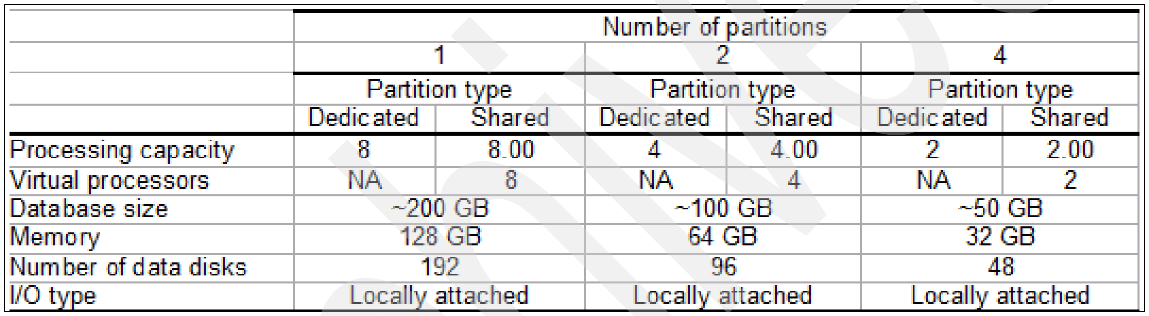
Note: If you are using the DLPAR facility to add or remove memory, we recommend the use of DB2's self tuning memory management (STMM) feature for most benefits. The STMM, with the *INSTANCE_MEMORY* database manager configuration parameter, and the *DATABASE_MEMORY* database configuration parameter (both set to **`AUTOMATIC`**) ensures that memory change in an LPAR is registered with DB2 for optimized memory use. Additionally, it takes the burden from the database administrator to retune DB2 after a memory change in an LPAR.

5.4 Dedicated LPAR versus micro-partition

There are two types of LPAR. Most of the time, questions arise regarding the correct partition type for the DB2 workload and environment. Besides performance, there are other factors that contribute to deciding the partition type.

A test was conducted with one, two, and four partitions running simultaneously for each partition type. All tests used direct attached I/O with properly scaled DB2 data. In essence, the test contained a set of six performance runs: two runs for a given number of partitions, each time changing only the partition type between dedicated and shared-processor partitions. The partition type change is a nondestructive change (no data loss). The only requirement is that the partition be restarted.

Figure 5-5 shows the results of these tests.



	Number of partitions					
	1		2		4	
	Partition type		Partition type		Partition type	
	Dedicated	Shared	Dedicated	Shared	Dedicated	Shared
Processing capacity	8	8.00	4	4.00	2	2.00
Virtual processors	NA	8	NA	4	NA	2
Database size	~200 GB		~100 GB		~50 GB	
Memory	128 GB		64 GB		32 GB	
Number of data disks	192		96		48	
I/O type	Locally attached		Locally attached		Locally attached	

Figure 5-5 Result of the tests

5.4.1 LPAR throughput

As shown in Figure 5-6, there is no difference between the types of partitions.

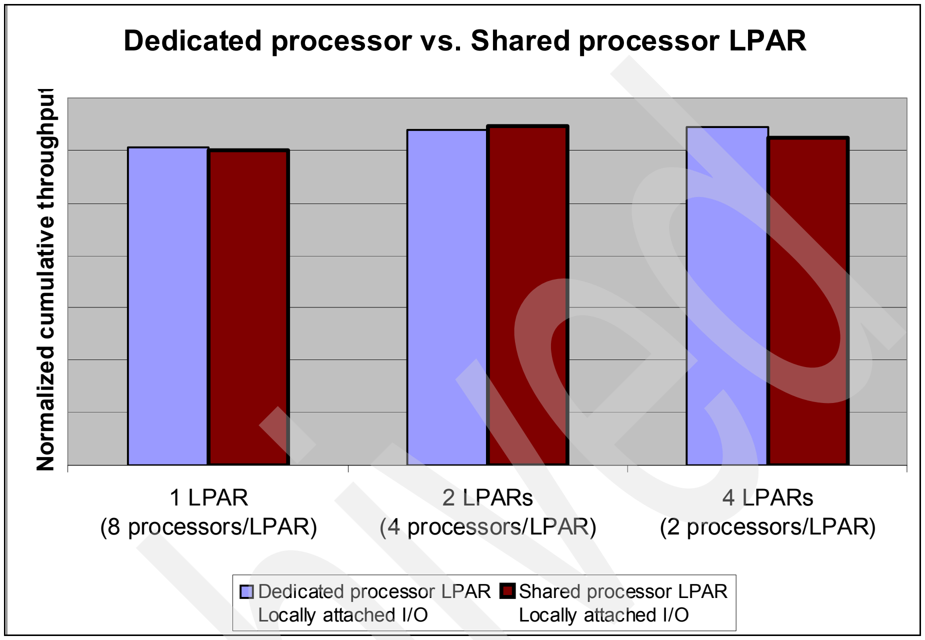


Figure 5-6 Compare dedicated and shared-processor partitions

Performance of the shared processor partition (for example the cumulative throughput) is identical to the dedicated partition, for one, two, and four partitions. This behavior is validation of the fact that POWER Hypervisor uses an efficient scheduling algorithm for the shared processor partitions with reasonably large processor entitlement. For such shared processor partitions, POWER Hypervisor attempts to schedule on the same physical cores each time, similar to dedicated processor partition.

5.4.2 LPAR scalability

Another interesting observation from the same data is depicted in Figure 5-7 on page 233. There is no partitioning scalability overhead for either type of partition. That is, the amount of work done by an eight processor dedicated processor partitions (or shared processor partitions with 8.00 entitled capacity with eight virtual processors) is twice that of the four processor partition, which, in turn, is twice that of the two processor partition.

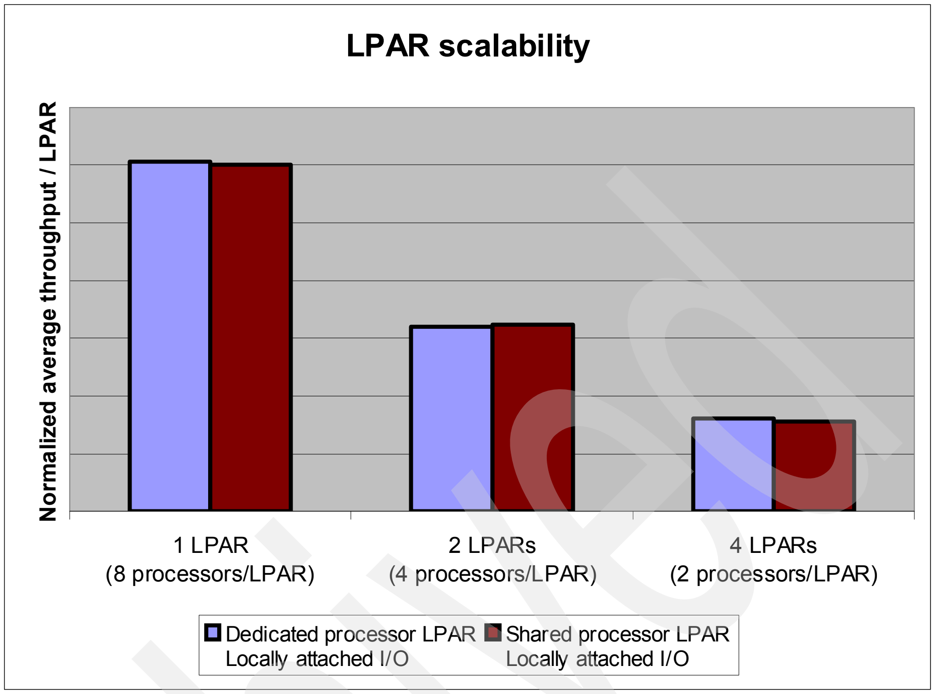


Figure 5-7 Dedicated processor and shared processor LPAR scalability

It is more cost-effective to reduce unused processing capacity within a partition by appropriately sizing a partition, which results in improved use at a system level. A general rule of thumb is not to waste processing capacity within a partition. Thus, the recommendation is to use a shared processor partition for DB2 deployments. The shared processor partition improves processor use for the whole system and offers flexible scaling using the uncapped feature when the workload demand exceeds the currently assigned processor entitlement.

5.4.3 Discussion

Beginning with POWER6 processor architecture-based servers, there is an option to donate unused dedicated processor partition processing capacity. This feature eliminates unused processing cycles in a dedicated processor partition, further solidifying PowerVM virtualization offering.

Strictly speaking from a performance point of view, theoretically, because of processor and memory affinity, a dedicated partition must offer optimal performance. However, it has been found that this is not the case for DB2 workloads. That is, there is no performance difference between dedicated and shared-processor partitions for the system under test.

With the same performance characteristics for both dedicated- and shared-processor partitions, consider other aspects and differences between the partition types. For a dedicated partition, processor assignment is in increments of one whole processor. Therefore, idle processor capacity often occurs within a dedicated partition. For example, a dedicated partition with three processors that runs DB2 and consumes the equivalent to 2.10 processors (that is, vmstat shows 70 percent partition use) results in 0.90 processor wasted capacity unless shared dedicated capacity for this LPAR is turned on.

The dedicated partition allows adding or removing processor resources at run time, without restarting AIX or DB2. The only way to alter dedicated partition processor allocation is by using a dynamic LPAR operation. Dynamic LPAR is relatively slower (with a latency of a few seconds, depending on the workload) than the near-instant entitled capacity change that is offered by an uncapped shared-processor partition. A shared-processor partition offers many advantages, including suitability for dynamic, unpredictable workloads, effective resource use, and easy workload management.

Tip: Whenever possible use a shared processor LPAR for running DB2. The shared processor LPAR (micro-partition) offers the same level of performance as a dedicated processor LPAR for most kinds of DB2 workloads. Besides, the shared processor LPAR offers many other workload management features and improves overall server use.

5.5 Uncapped micro-partition

This is one of the primary distinguishing features between dedicated and shared-processor partitions. The dedicated-processor partition is capped by definition. That is, a dedicated-processor partition cannot use more than its allocated processors, even if idle processing capacity is available on the system. When applied appropriately, uncapped shared-processor partition is effective in dealing with highly dynamic or unpredictable workloads. The uncapped weight is used to arbitrate free-pool capacity allocation among competing uncapped shared-processor partitions.

As an example, consider a scenario where a multitier infrastructure consists of a DB2 server as a back-end tier and IBM WebSphere as an application server. Each is hosted in an individual, uncapped shared-processor partition of a System p server. Assume that the DB2 partition is a high-priority partition. Therefore, the partition's uncapped weight is set higher than the WebSphere partition. At peak demand, both DB2 and WebSphere exceed the entitled capacity and are allowed to use free-pool processing capacity (if available). However, the DB2 partition gets more free-pool processing cycles than WebSphere.

5.6 Micro-partition: Virtual processors

The use of micro-partitions requires setting the number of virtual processors to an appropriate value. Under-configuring the number of virtual processors can lead to wasted processing capacity or might not allow the use of free-pool processing cycles (the uncapped partition's maximum entitled capacity is limited to the number of virtual processors). Over-configuring of virtual processors is not a problem, because, by default, AIX does not schedule unused virtual processors. (refer to AIX **schedo** command configuration parameter vpm_xvcpus)

A test was performed to measure virtual-processor folding effectiveness by varying the number of virtual processors for one of the shared-processor partitions from the four-partition scenario listed in Figure 5-5 on page 231. As expected, a relatively higher number of virtual processors did not have any negative performance impact. The partition had an entitled-processor capacity of 1.85. Three runs were performed. The first test involved the number of virtual processors being set to round up the entitled capacity (1.85 to 2) Subsequent tests configured a much higher number of virtual processors.

Based on the recorded results on the setup, a recommendation for the DB2 environment, for capped partitions, is to set the number of virtual processors equal to the rounded-up entitled capacity.

For an uncapped partition, we suggest the number of virtual processor to be twice the round up value of the current entitlement, but not to exceed the total number of physical processors configured in a server. A 16-way server, hosting an uncapped shared processor LPAR with an entitled capacity of 1.85 must have four virtual processors, whereas another uncapped shared processor LPAR (on the same system) with an entitled capacity of 8.50 must have 16 virtual processors.

This configuration for uncapped partitions ensures that there are sufficiently configured virtual processors to allow the use of the free-pool processing capacity. Carefully tune uncapped weight to assist fair distribution of the free pool capacity.

Tip: For DB2 environment use the following general guideline to configure virtual processors for shared processor LPAR.

For capped shared processor LPAR, set the number of virtual processor to be equal to the rounded-up number of the entitled capacity.

For uncapped shared processor LPAR, set the number of virtual processor to be equal to the round-up of minimum (2 X entitled capacity, number of pyhsical processors in a system)

5.7 Multiple shared-processor pools

POWER6 processor-based systems support Multiple Shared-Processor Pools (MSPPs). Unlike the previous mechanism, where all micro-partitions belong to system wide default pool (called a *physical shared processor pool*), this new capability allows a systems administrator to create a set of micro-partitions in a separate, user-defined pool.

Note: For more information about multiple shared-processor pools see the IBM Redbooks publication *PowerVM Virtualization on IBM System p: Introduction and Configuration Fourth Edition*, SG24-7940.

MSPPs use processor capacity from the physical shared processor pool. There can only be one physical shared processor pool in the system. Each shared-processor pool has an associated entitled pool capacity, which is consumed by the set of micro-partitions belonging in that shared-processor pool.

The micro-partitions within a shared-processor pool are guaranteed to receive their entitled capacity. In addition, unused processor cycles within their shared-processor pool are harvested and are redistributed to eligible micro-partitions within the same shared-processor pool.

The source of additional processor cycles can be the reserved pool capacity, which is processor capacity that is specifically reserved for this particular shared-processor pool, but not assigned to any of the micro-partitions in the shared-processor pool.

When the set of micro-partitions in a shared-processor pool are heavily loaded, they can consume additional processor capacity (assuming they are uncapped) from outside their shared-processor pool up to a defined maximum (the maximum pool capacity). Processor capacity distributed in this way has been ceded by underused sets of micro-partitions in their shared-processor pool.

There is a default shared-processor pool (shared-processor pool$_0$) that is used for all micro-partitions if the system administrator has not created other shared-processor pools. The shared-processor pool$_0$ has properties that ensure operational consistency with POWER6 processor-based micro-partitioning and the physical shared processor pool.

Figure 5-8 shows an architectural overview of a POWER6-based server with two shared-processor pools defined.

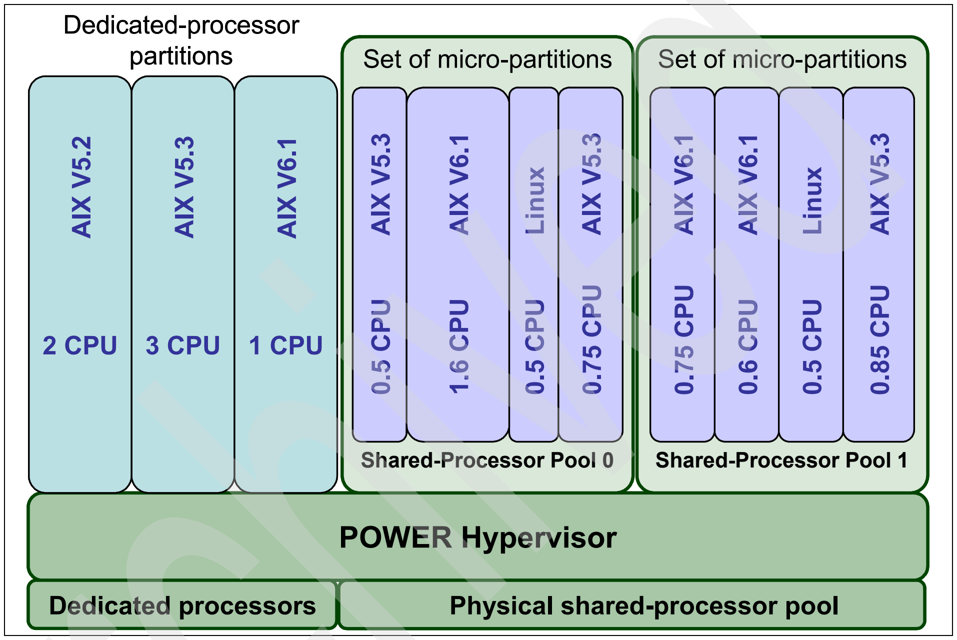


Figure 5-8 POWER6-based server with MSPPs defined

There can be up to 64 MSPPs in total. With the default MSPP, there can be an additional 63 MSPPs that are user-defined.

Workload management using the uncapped feature of micro-partition becomes challenging if a system has an excessive number of uncapped micro-partitions. The uncapped weight offers limited tuning options for distributing free pool capacity. Additionally, managing uncapped weight across dozens of micro-partitions is challenging, too. To address this, a shared processor pool can be defined with a relatively small number of micro-partitions belonging to it. This method offers a more granular processor resource management.

If a system is hosting a variety of applications and workloads, we suggest a separate shared-processor pool for application stack containing DB2 workloads. For example, in the case of a multi-tier application stack, all related stack components are part of the same shared processor pool for granular processor resource management. Also, if the VIOS is an uncapped micro-partition, and is predominantly servicing DB2 I/O requests, the VIOS is recommended to be part of the same shared processor pool as well.

5.8 LPAR monitoring

The standard AIX monitoring tools and interfaces are enhanced for the virtualization environment. Additionally, there are a few new monitoring tools to get the status and monitor an LPAR activity in an AIX virtualization environment

A new AIX command, **lparstat**, is available to get information about the LPAR profile and current use. Example 5-1 shows the output of the **lparstat -i** command, which prints exhaustive LPAR information, including LPAR type, virtual processors, entitled capacity, and memory.

Example 5-1 LPAR information using lparstat command

```
# lparstat -i
Node Name                                  : el9-93-11
Partition Name                             : MOBILE-LPAR
Partition Number                           : 3
Type                                       : Shared-SMT
Mode                                       : Capped
Entitled Capacity                          : 1.00
Partition Group-ID                         : 32771
Shared Pool ID                             : 0
Online Virtual CPUs                        : 2
Maximum Virtual CPUs                       : 16
Minimum Virtual CPUs                       : 1
Online Memory                              : 40960 MB
Maximum Memory                             : 56320 MB
Minimum Memory                             : 1024 MB
Variable Capacity Weight                   : 0
Minimum Capacity                           : 0.10
Maximum Capacity                           : 8.00
Capacity Increment                         : 0.01
Maximum Physical CPUs in system            : 8
Active Physical CPUs in system             : 8
Active CPUs in Pool                        : 8
Shared Physical CPUs in system             : 8
Maximum Capacity of Pool                   : 800
Entitled Capacity of Pool                  : 180
Unallocated Capacity                       : 0.00
```

```
Physical CPU Percentage                        : 50.00%
Unallocated Weight                             : 0
Memory Mode                                    : Dedicated
Total I/O Memory Entitlement                   : -
Variable Memory Capacity Weight                : -
Memory Pool ID                                 : -
Physical Memory in the Pool                    : -
Hypervisor Page Size                           : -
Unallocated Variable Memory Capacity Weight: -
Unallocated I/O Memory entitlement             : -
Memory Group ID of LPAR                        : -
Desired Virtual CPUs                           : 2
Desired Memory                                 : 40960 MB
Desired Variable Capacity Weight               : 0
Desired Capacity                               : 1.00
Target Memory Expansion Factor                 : -
Target Memory Expansion Size                   : -
```

Another use of the **lparstat** command is similar to **vmstat**. **lparstat** can be used to get current rolling LPAR use and other performance counters. Additionally, **lparstat** offers many command options to monitor various performance counters. See Example 5-2, Refer to the *AIX Commands Reference* at the following Web page for more information:

http://publib.boulder.ibm.com/infocenter/pseries/v5r3/index.jsp?topic=/com.ibm.aix.doc/doc/base/commandsreference.htm

Example 5-2 lparstat command

```
# lparstat 1

System configuration: type=Shared mode=Capped smt=On lcpu=4 mem=40960MB
psize=8 ent=1.00

%user  %sys  %wait  %idle physc %entc  lbusy  vcsw phint  %nsp
----- ----- ------ ------ ----- ----- ------ ----- ----- -----
  0.1   0.3    0.0   99.6  0.01   0.8    0.0   472     0    99
  0.0   0.2    0.0   99.8  0.01   0.6    0.0   471     0    99
  0.0   0.7    0.0   99.3  0.01   1.1    0.0   468     0    99
  0.0   0.2    0.0   99.8  0.01   0.6    0.0   469     0    99
  0.0   0.2    0.0   99.8  0.01   0.6    0.0   469     0    99
  0.2   0.5    0.0   99.3  0.01   1.1    0.0   474     0    99
```

The AIX **vmstat** command shows additional information for the shared processor LPAR. As shown in Example 5-3, the last two columns are displayed only for shared processor partition. **pc** is number of physical processors consumed, and **ec** is the percentage of entitled capacity consumed. It can be more than 100 if the partition is uncapped, and it is currently using more than its entitlement.

Example 5-3 vmstat for shared processor LPAR

```
# vmstat 1

System configuration: lcpu=4 mem=40960MB ent=1.00

kthr    memory            page            faults          cpu
----- ----------- ------------------------ ------------
-----------------------
 r  b   avm   fre  re  pi  po  fr   sr  cy  in   sy  cs us sy id wa
pc    ec
 0  0 1062519 7866593   0   0   0   0    0   0   9 7652 358  3  3 94  0
0.06   5.9
 0  0 1062520 7866592   0   0   0   0    0   0   2   62 297  0  1 99  0
0.01   1.2
 0  0 1062520 7866592   0   0   0   0    0   0   3   58 277  0  0 99  0
0.01   0.7
 0  0 1062520 7866592   0   0   0   0    0   0   3  918 284  0  1 99  0
0.01   1.2
 0  0 1062520 7866592   0   0   0   0    0   0   5   58 264  0  0 99  0
0.01   0.7
 0  0 1062520 7866592   0   0   0   0    0   0   3   59 279  0  0 99  0
0.01   0.6
 0  0 1062520 7866592   0   0   0   0    0   0   3   75 294  0  0 99  0
0.01   0.6
 0  0 1062520 7866592   0   0   0   0    0   0   4   81 285  0  0 99  0
0.01   0.6
 0  0 1062520 7866592   0   0   0   0    0   0   1   62 289  0  0 99  0
0.01   0.6
 0  0 1062520 7866592   0   0   0   0    0   0   2   64 281  0  0 99  0
0.01   0.6
 0  0 1062520 7866592   0   0   0   0    0   0  11 7602 359  4  4 92  0
0.09   8.8
 0  0 1062520 7866592   0   0   0   0    0   0   2   60 296  0  0 99  0
0.01   0.7
```

6

Virtual I/O

DB2 performance is dependent on the I/O subsystem performance. To attain the best possible I/O throughput, the data layouts of database tables demand special attention from database administrators and system administrators. The chosen I/O type has great impact on the manageability and extensibility of the DB2 storage requirements. Thus, it is critical to consider workload priorities and to examine trade-offs between disk I/O types. You can choose between locally attached I/O or virtual IO, or both within a partition.

In this chapter we discuss the advantages of using virtual I/O, VIOS (which is the backbone for enabling virtual I/O), high availability for virtual I/O and best practices for configuring for virtual I/O for a DB2 workload.

This chapter has the following sections:

Note: Some of the information in this chapter has been consolidated from the following materials. For more information, you can refer to these sources:

- IBM Systems Hardware Information Center:

  http://publib.boulder.ibm.com/infocenter/powersys/v3r1m5/index.jsp?topic=/iphb1/iphb1_vios_virtualioserveroverview.htm

- IBM System p Advanced POWER Virtualization Best Practices:

  http://www.redbooks.ibm.com/redpapers/pdfs/redp4194.pdf

- PowerVM Virtualization on IBM System p: Introduction and Configuration:

  http://www.redbooks.ibm.com/redbooks/pdfs/sg247590.pdf

- DB2 Best Practices - Improving Data Server Utilization and Management through Virtualization:

  http://download.boulder.ibm.com/ibmdl/pub/software/dw/dm/db2/bestpractices/DB2BP_Virtualization_0508I.pdf

- Introduction to the VIOS Whitepaper:

  http://www-03.ibm.com/systems/p/hardware/whitepapers/virtual_io.html

6.1 Virtual I/O benefits

This Power VM feature provides the ability to create multiple LPARs in a system. Each LPAR needs storage for a root disk (AIX rootvg volume group) to install the operating system and network adapter. Without the use of virtual I/O, each LPAR needs a physical I/O adapter to provide access to the needed storage and network adapter. Such requirements can limit the number of LPARs in a system to the number of available I/O adapters. However, the use of virtual I/O eliminates such requirements, and allows creating up to 255 LPARs in a system.

Use of virtual I/O requires an LPAR with a special operating system. This LPAR is called a virtual I/O server (VIOS). VIOS enables LPARs to share I/O resources such as Ethernet adapters, Fibre Channel adapters, and disk storage. When an LPAR (an LPAR running DB2, for example) is using virtual I/O, all I/O requests are passed down to the VIOS, where VIOS performs the actual I/O operation and returns data, in a few cases, directly to a client partition (no double buffering in the case of Virtual SCSI).

Using the VIOS facilitates the following functions:

- Sharing of physical resources between logical partitions on the system
- Sharing of disk space
- Creating logical partitions without requiring additional physical I/O resources
- Creating more logical partitions than there are I/O slots or physical devices available with the ability for logical partitions to have dedicated I/O, virtual I/O, or both
- Maximizing use of physical resources on the system
- Helping to reduce the Storage Area Network (SAN) infrastructure
- Server consolidation

In most cases, DB2 workloads fluctuate over time. The fluctuation leads to variations in the use of I/O infrastructure. Typically, in the absence of virtual I/O, each partition is configured with sizing based of the peak I/O capacity, resulting in under used I/O infrastructure. Sharing resources where the peaks and valleys are complementary (for example, daytime OLTP workload with nightly batch or reporting workloads) can result in a significant reduction in the total amount of physical resource requirements to handle the aggregated workload for all the partitions that have these complementary peaks and valleys. By reducing the number of physical disk drives and networking infrastructure, virtual I/O simplifies the management of these entities.

To improve overall use and performance, an objective of the DB2 sizing and tuning exercise is to reduce or eliminate I/O bottleneck in the DB2 environment. To that end, to attain the best possible I/O bandwidth, DB2 storage requirements are typically sized in terms of number of disk spindles, rather than the amount of data storage. This results in an excessive amount of unused disk space. Using virtual I/O, such disk space can be shared for complimentary or staggered workloads.

6.2 VIOS features

The following list details a few of the key features available by using VIOS:

- Storage virtualization
  - virtual SCSI
  - virtual Fibre Channel (NPIV)
- Network virtualization
- Live Partition Mobility
- Integrated Virtualization Manager
- Shared memory paging for Active Memory Sharing

6.2.1 Storage virtualization

VIOS allows virtualization of physical storage resources. The storage devices are accessed by the client partitions either through virtual SCSI or through virtual Fibre Channel.

Virtual SCSI

Physical adapters with attached disks or optical devices on the VIOS logical partition can be shared by one or more client logical partitions. The VIOS offers a local storage subsystem that provides standard SCSI-compliant logical unit numbers (LUNs). VIOS can export a pool of heterogeneous physical storage as a homogeneous pool of block storage in the form of SCSI disks for client LPAR to use.

Virtual SCSI is based on a client-server paradigm. VIOS owns the physical resources as well as the virtual SCSI adapter, and acts as a server, or SCSI target device. The client logical partitions have a SCSI initiator referred to as the virtual SCSI client adapter, and access the virtual SCSI targets as standard SCSI LUNs. Configure the virtual adapters by using the HMC or Integrated Virtualization Manager. The configuration and provisioning of virtual disk resources is performed by using the VIOS. Physical disks owned by the VIOS can be either exported and assigned to a client logical partition as a whole or can be partitioned into parts, such as logical volumes or files. The logical volumes and files can be assigned to logical partitions. Therefore, using virtual SCSI, you can share adapters as well as disk devices. To make a physical volume, logical volume, or file available to a client logical partition requires that it be assigned to a virtual SCSI server adapter on the VIOS. The client logical partition accesses its assigned disks through a virtual-SCSI client adapter. The virtual-SCSI client adapter recognizes standard SCSI devices and LUNs through this virtual adapter.

Figure 6-1 shows a standard virtual SCSI configuration.

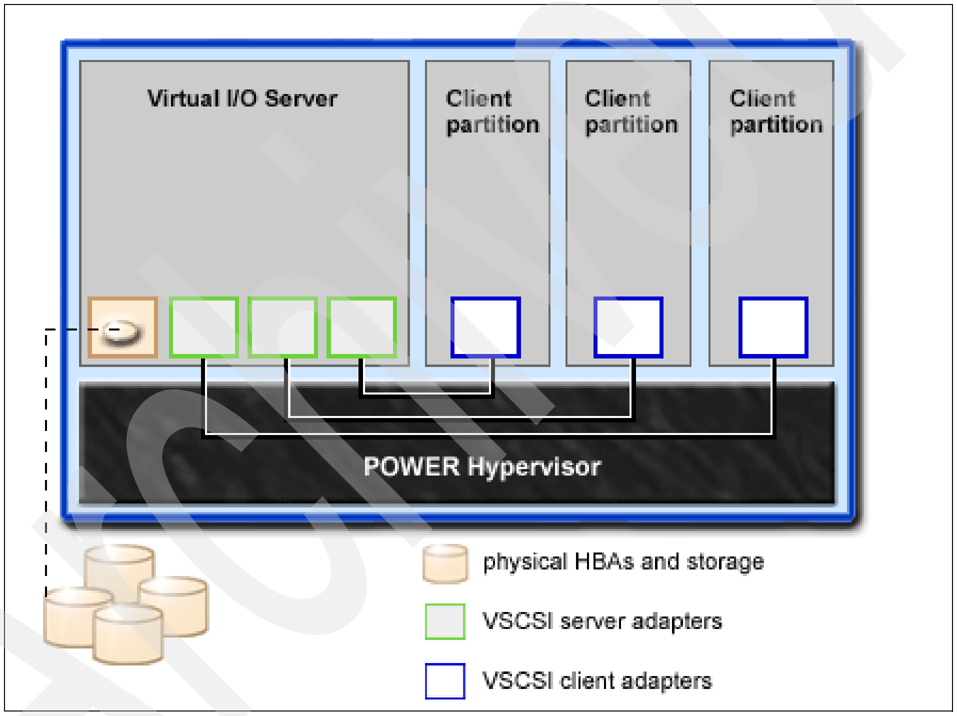


Figure 6-1 Virtual SCSI configuration

After virtual devices are configured, those devices appear as regular hdisks in AIX device configuration. The hdisk can be used like an ordinary hdisk for creating a logical volume group. The important point is that from DB2's perspective, DB2 is unaware of the fact that it is using virtual storage. Therefore, all of the DB2 storage management aspects, such as including containers, table spaces, automatic storage remain as is, even for the virtual devices

Virtual Fibre Channel (NPIV)

With N_Port ID Virtualization (NPIV), you can configure the managed system so that multiple logical partitions can access independent physical storage through the same physical Fibre Channel adapter.

To access physical storage in a typical SAN that uses Fibre Channel, the physical storage is mapped to LUNs. The LUNs are mapped to the ports of physical Fibre Channel adapters. Each physical port on each physical Fibre Channel adapter is identified using one worldwide port name (WWPN).

NPIV is a standard technology for Fibre Channel networks that enables you to connect multiple logical partitions to one physical port of a physical Fibre Channel adapter. Each logical partition is identified by a unique WWPN, which means that you can connect each logical partition to independent physical storage on a SAN.

NPIV support is included with PowerVM Express, Standard, and Enterprise Edition and supports AIX V5.3 and AIX V6.1 partitions on selected POWER6 processor-based servers Power 520, 550, 560, and 570, with an 8 GB Fibre Channel Host Bus Adapter (HBA).

To enable NPIV on the managed system, you must create a VIOS logical partition (version 2.1 or later) that provides virtual resources to client logical partitions. You assign the physical Fibre Channel adapters (that support NPIV) to the VIOS logical partition. Then you connect virtual Fibre Channel adapters on the client logical partitions to virtual Fibre Channel adapters on the VIOS logical partition. A virtual Fibre Channel adapter is a virtual adapter that provides client logical partitions with a Fibre Channel connection to a SAN through the VIOS logical partition. The VIOS logical partition provides the connection between the virtual Fibre Channel adapters on the VIOS logical partition and the physical Fibre Channel adapters on the managed system.

Figure 6-2 on page 247 shows a managed system configured to use NPIV.

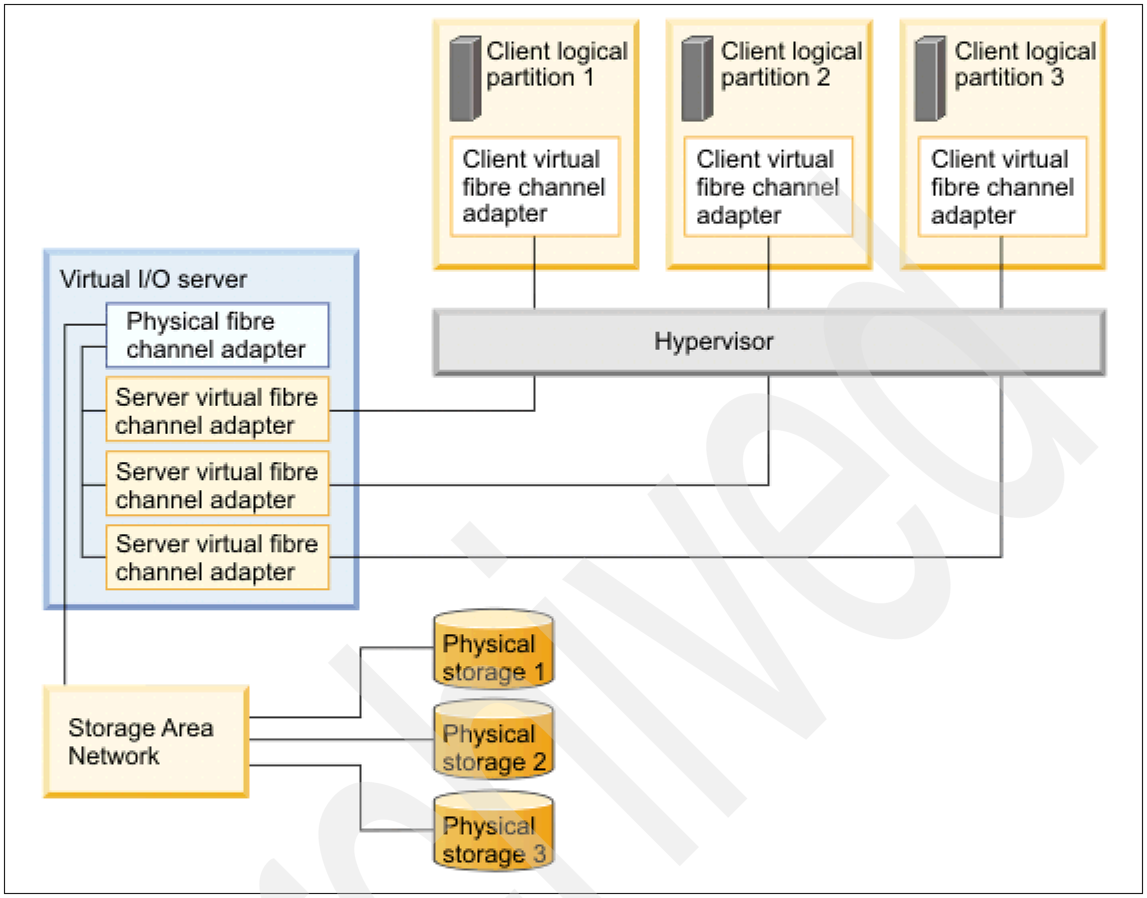

Client logical partition 1
Client virtual fibre channel adapter
Client logical partition 2
Client virtual fibre channel adapter
Client logical partition 3
Client virtual fibre channel adapter
Virtual I/O server
Physical fibre channel adapter
Server virtual fibre channel adapter
Server virtual fibre channel adapter
Server virtual fibre channel adapter
Hypervisor
Physical storage 1
Physical storage 2
Physical storage 3
Storage Area Network

Figure 6-2 a managed system configured to use NPIV

Figure 6-2 shows the following connections:

- A storage area network (SAN) connects three units of physical storage to a physical Fibre Channel adapter that is located on the managed system. The physical Fibre Channel adapter is assigned to the VIOS and supports NPIV.
- The physical Fibre Channel adapter connects to three virtual Fibre Channel adapters on the VIOS. All three virtual Fibre Channel adapters on the VIOS connect to the same physical port on the physical Fibre Channel adapter.
- Each virtual Fibre Channel adapter on the VIOS connects to one virtual Fibre Channel adapter on a client logical partition. Each virtual Fibre Channel adapter on each client logical partition receives a pair of unique WWPNs. The client logical partition uses one WWPN to log into the SAN at any given time. The other WWPN is used when you move the client logical partition to another managed system.

Using their unique WWPNs and the virtual Fibre Channel connections to the physical Fibre Channel adapter, the operating systems that run in the client logical partitions discover, instantiate, and manage their physical storage located on the SAN. In Figure 6-2 on page 247, Client logical partition 1 accesses Physical storage 1, Client logical partition 2 accesses Physical storage 2, and Client logical partition 3 accesses Physical storage 3. The VIOS cannot access and does not emulate the physical storage to which the client logical partitions have access. The VIOS provides the client logical partitions with a connection to the physical Fibre Channel adapters on the managed system.

There is always a one-to-one relationship between virtual Fibre Channel adapters on the client logical partitions and the virtual Fibre Channel adapters on the VIOS logical partition. That is, each virtual Fibre Channel adapter on a client logical partition must connect to only one virtual Fibre Channel adapter on the VIOS logical partition, and each virtual Fibre Channel on the VIOS logical partition must connect to only one virtual Fibre Channel adapter on a client logical partition.

Using SAN tools, you can zone and mask LUNs that include WWPNs that are assigned to virtual Fibre Channel adapters on client logical partitions. The SAN uses WWPNs that are assigned to virtual Fibre Channel adapters on client logical partitions the same way it uses WWPNs that are assigned to physical ports.

Note: DB2 is not aware of the fact that it is using virtual storage. All of the storage management best practices and methods discussed so far are equally applied to virtual storage as well.

6.2.2 Network virtualization

The hypervisor on a POWER 5 and POWER 6 system provide virtual network support. Without requiring additional hardware or external cables, a virtual LAN (VLAN) facilitates high-speed virtual Ethernet communication paths among multiple partitions within a physical system that run AIX, Linux, and other operating systems. You can dynamically create virtual Ethernet segments and restrict access to a VLAN segment to meet security or traffic segregation requirements. A virtual Ethernet has the same characteristics as a high-bandwidth, physical Ethernet network and supports multiple networking protocols, such as IPv4, IPv6, and ICMP.

VIOS provides the virtual networking technologies discussed in the next three sections.

Shared Ethernet adapter (SEA)

The shared Ethernet adapter (SEA) hosted in the VIOS acts as a layer-2 bridge between the internal virtual and external physical networks. The SEA enables partitions to communicate outside the system without having to dedicate a physical I/O slot and a physical network adapter to a client partition

SEA failover

SEA failover provides redundancy by configuring a backup SEA on a VIOS logical partition that can be used if the primary SEA fails. The network connectivity in the client logical partitions continues without disruption.

Link aggregation (or EtherChannel)

A Link Aggregation (or EtherChannel) device is a network port-aggregation technology that allows several Ethernet adapters to be aggregated. The adapters can then act as a single Ethernet device. Link Aggregation helps provide more throughput over a single IP address than is possible with a single Ethernet adapter.

6.2.3 Live Partition Mobility

Live Partition Mobility provides the ability to move AIX and Linux logical partitions from one system to another. The mobility process transfers the system environment, including the processor state, memory, attached virtual devices, and connected users.

Active Partition Mobility allows you to move AIX and Linux logical partitions that are running, including the operating system and applications, from one system to another. The logical partition and the applications running on that migrated logical partition do not need to be shut down.

Inactive Partition Mobility allows you to move a powered off AIX and Linux logical partition from one system to another.

The VIOS plays a main role in live partition mobility. The Live Partition Mobility is initiated from the HMC. The HMC communicates with the VIOS on the two systems to initiate the transfer. The VIO servers on both system talk to each other, and transfer the system environment including the processor state, memory, attached virtual devices, and connected users.

For more information about Live Partition Mobility you can refer to Chapter 7, “Live Partition Mobility” on page 269.

6.2.4 Integrated Virtualization Manager

The Integrated Virtualization Manager (IVM) provides a Web-based system management interface and a command-line interface that you can use to manage IBM Power Systems servers and IBM BladeCenter blade servers that use the IBM VIOS. On the managed system, you can create logical partitions, manage virtual storage and virtual Ethernet, and view service information related to the server. The IVM is included with the VIOS, but it is available and usable only on certain platforms, and where no Hardware Management Console (HMC) is present.

If you install the VIOS on a supported server, and if there is no HMC attached to the server when you install the VIOS, then the IVM is enabled on that server. You can use the IVM to configure the managed system through the VIOS.

6.2.5 Shared memory paging for Active Memory Sharing

Logical partitions can share the memory in the shared memory pool by using the PowerVM Active Memory Sharing technology (or shared memory). Instead of assigning a dedicated amount of physical memory to each logical partition that uses shared memory (hereafter referred to as *shared memory partitions*), the hypervisor constantly provides the physical memory from the shared memory pool to the shared memory partitions as needed. The hypervisor provides portions of the shared memory pool that are not currently being used by shared memory partitions to other shared memory partitions that need to use the memory. When a shared memory partition needs more memory than the current amount of unused memory in the shared memory pool, the hypervisor stores a portion of the memory that belongs to the shared memory partition in auxiliary storage. Access to the auxiliary storage is provided by a VIOS logical partition. When the operating system attempts to access data that is located in the auxiliary storage, the hypervisor directs a VIOS to retrieve the data from the auxiliary storage and write it to the shared memory pool so that the operating system can access the data.

6.3 VIOS resilience

Data storage subsystem performance and high availability is crucial for DB2 workloads. The virtual I/O must continue offering the same level of resiliency as direct attached storage. By design, VIOS is extremely robust. It primarily runs device drivers and does not run any application workloads.

To provide high availability of virtual IO to the client partitions, redundancy can be built into VIOS itself by using a combination of the following items:

- Redundant physical hardware
- Network Interface Backup Ethernet configuration
- SEA failover configuration
- Storage Logical Volume Manager (LVM) mirroring and RAID configurations
- SAN multipath I/O (MPIO)
- RAID protected storage (RAID provided either by the storage subsystem or by a RAID adapter)
- Hot-pluggable network adapters instead of built-in integrated network adapters

Two or more VIOSs and redundant devices can provide improved software maintenance and hardware replacement strategies.

6.3.1 Dual VIOS

With multiple virtual I/O client partitions, running DB2, dependent on a VIOS for I/O demands, you can implement dual VIOSs, duplicate paths and devices to provide additional system service and configuration options.

Fundamentally, the primary reasons for redundant VIOSs configuration include:

- Future hardware expansion and new function
- Unscheduled outages due to human intervention
- Unscheduled outages due to physical device failure or natural events
- Scheduled outages required for VIOS maintenance
- Isolation of network and storage workload to provide increased virtual I/O client partition performance
- Multiple multipath codes such as MPIO and device redundancy

The following sections show the storage and network redundancy options provided by a dual VIOS environment.

Virtual SCSI redundancy

Virtual SCSI redundancy can be achieved using MPIO at client partition and VIOS level. Figure 6-3 displays a setup using MPIO in the client partition.

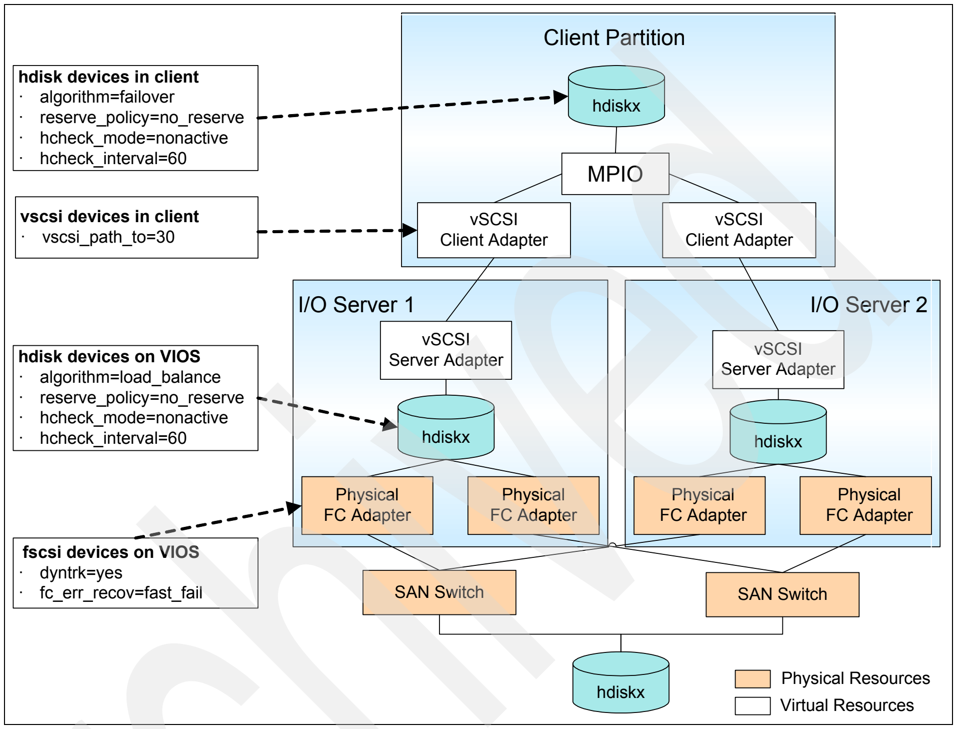


Figure 6-3 MPIO attributes

Two VIOSs host disks for a client partition. The client is using MPIO to access a SAN disk. From the client perspective, the following situations can be handled without causing downtime for the client:

- Either path to the SAN disk can fail, but the client is still able to access the data on the SAN disk through the other path. No action has to be taken to reintegrate the failed path to the SAN disk after repair if MPIO is configured
- Either VIOS can be rebooted for maintenance. This results in a temporary simultaneous failure of one path to the SAN disk, as described before.

Note: It is recommend to use a minimum of two Fibre Channel adapters in each VIOS for adapter redundancy.

The MPIO for virtual SCSI devices only supports failover mode. For any given virtual SCSI disk, a client partition uses a primary path to one VIOS and fail over to the secondary path to use the other VIOS. Only one path is used at a given time even when both paths are enabled.

To balance the load of multiple client partitions across two VIOSs, the priority on each virtual SCSI disk on the client partition can be set to select the primary path and, therefore, a specific VIOS. The priority is set on a per virtual SCSI disk basis using the **chpath** command as shown in the following example (1 is the highest priority):

```
chpath -l hdisk0 -p vscsi0 -a priority=2
```

Due to this granularity, a system administrator can specify whether all the disks or alternate disks on a client partition use one of the VIOSs as the primary path. The recommended method is to divide the client partitions between the two VIOSs.

For detailed information about the various device settings mentioned in Figure 6-3 on page 252, refer to IBM Redbooks publication *IBM PowerVM Virtualization Managing and Monitoring*, SG24-7590.

Virtual network redundancy

The two common methods available to provide virtual I/O client partition network redundancy in dual VIOSs configurations are:

- Network Interface Backup (NIB)
- Shared Ethernet Adapter (SEA) Failover

Network Interface Backup

Figure 6-4 shows a highly-available network configuration using dual VIOSs.

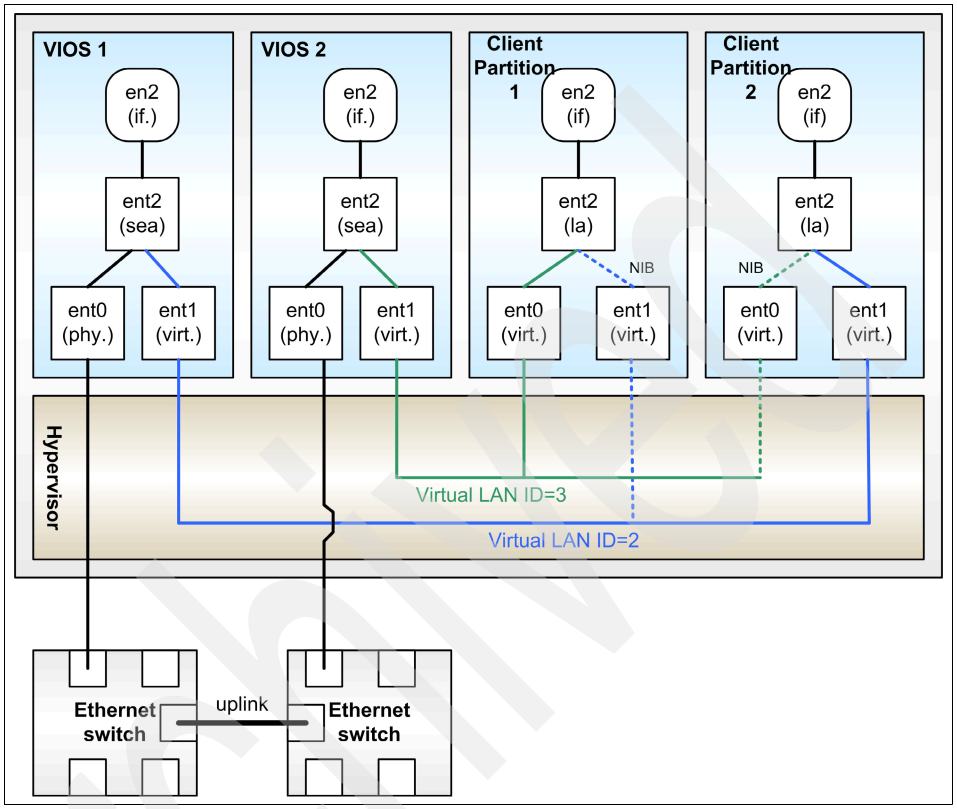


Figure 6-4 Network Interface Backup using dual VIOSs

This configuration uses virtual Ethernet adapters created using default virtual LAN IDs with the physical Ethernet switch using untagged ports only. In this example, VIOS 1 has a SEA that provides external connectivity to client partition 1 through the virtual Ethernet adapter using virtual LAN ID 2. VIOS 2 also has a SEA that provides external connectivity to the client partition through the virtual Ethernet adapter using virtual LAN ID 3. Client partition 2 has a similar setup, except that the virtual Ethernet adapter using virtual LAN ID 2 is the primary and virtual LAN ID 3 is the backup. This enables client partition 2 to get its primary connectivity through VIOS 1 and backup connectivity through VIOS 2.

Client partition 1 has the virtual Ethernet adapters configured using Network Interface Backup such that the virtual LAN ID 3 network is the primary and virtual LAN ID 2 network is the backup, with the IP address of the default gateway to be used for heartbeats. This enables client partition 1 to get its primary connectivity through VIOS 2 and backup connectivity through VIOS 1.

If the primary VIOS for an adapter becomes unavailable, the Network Interface Backup mechanism detects this because the path to the gateway is broken. The Network Interface Backup setup fails over to the backup adapter that has connectivity through the backup VIOS.

SEA failover

SEA failover is implemented on the VIOS using a bridging (layer-2) approach to access external networks. SEA failover supports IEEE 802.1Q VLAN-tagging, unlike Network Interface Backup.

With SEA failover, two VIOSs have the bridging function of the SEA to automatically fail over if one VIOS is unavailable or the SEA is unable to access the external network through its physical Ethernet adapter. A manual failover can also be triggered.

As shown in Figure 6-5, both VIOSs attach to the same virtual and physical Ethernet networks and VLANs, and both virtual Ethernet adapters of both Shared Ethernet Adapters have the access external network flag enabled. An additional virtual Ethernet connection must be set up as a separate VLAN between the two VIOSs and must be attached to the Shared Ethernet Adapter (SEA) as a control channel. This VLAN serves as a channel for the exchange of keep-alive or heartbeat messages between the two VIOSs that controls the failover of the bridging functionality. No network interfaces have to be attached to the control channel Ethernet adapters. The control channel adapter is dedicated and on a dedicated VLAN that is not used for any other purpose.

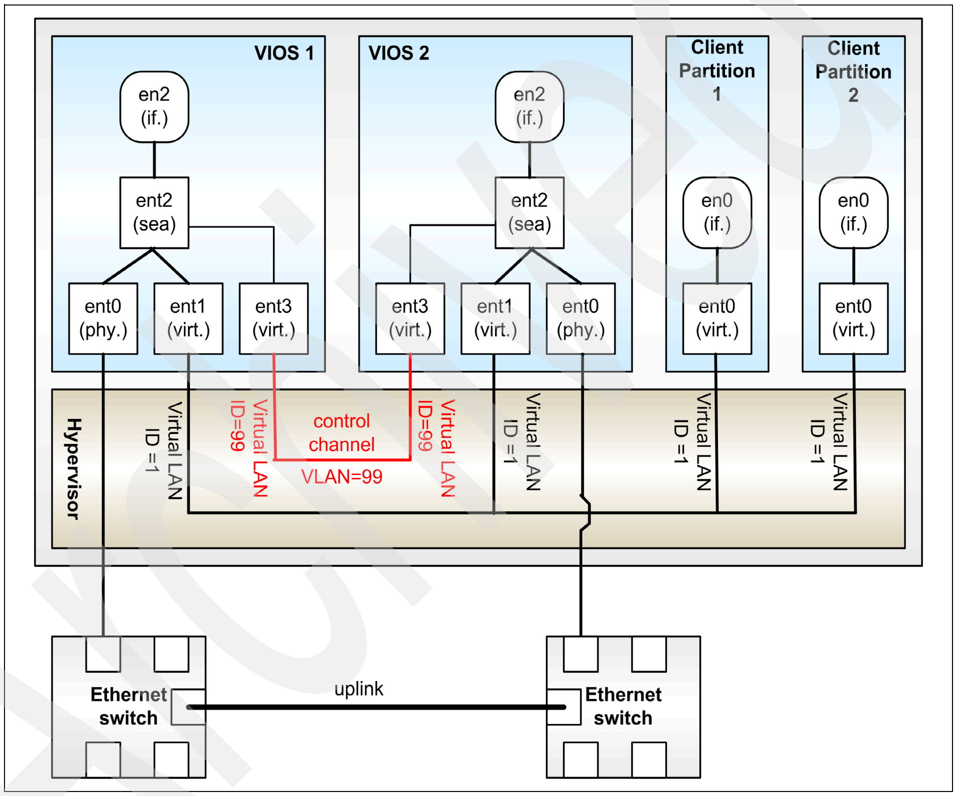


Figure 6-5 SEA failover using dual VIOSs

In addition, the SEA in each VIOS must be configured with differing priority values. The priority value defines which of the two SEAs is the primary (active) and which is the backup (standby). The lower the priority value, the higher the priority (for example, priority=1 is the highest priority).

You can also configure the SEA with an IP address that it periodically pings to confirm that network connectivity is available. This is similar to the IP address to ping that can be configured with Network Interface Backup (NIB). If you use NIB, you have to configure the reachability ping on every client compared to doing it when on the SEA.

It is possible that during an SEA failover, the network drops up to 15–30 packets while the network reroutes the traffic.

6.4 VIOS sizing

The only certain way to size any server is to run it with the real workload, monitor, and then tune.

The following sections provide a starting point for tuning.

- The IBM Systems Hardware Information Center has detailed planning calculations to help with CPU and memory planning:

  http://publib.boulder.ibm.com/infocenter/eserver/v1r3s/topic/iphb1/iphb1_vios_planning.htm

- To use the IBM Systems Workload Estimator tool to help size your server, see:

  http://www-912.ibm.com/wle/EstimatorServlet

The following guidelines are intended to get you started and might need additional adjustments after they are run and monitored in production, but they enable you to accurately plan how much CPU and memory to use.

6.4.1 VIOS memory sizing

The VIOS, similar to other workloads, requires system memory to operate. The memory requirement for servicing virtual I/O is insignificant. The memory requirement for the VIOS mainly depends on network requirements. The memory used for network communication is used to buffer network traffic, so detailed planning involves knowing such factors as the messages sizes (MTU or jumbo frames) and how many SEAs to use

The easiest rule to follow is that if you have a simple VIOS that is only bridging a couple of networks and never uses jumbo frames or a larger MTU, 512 MB is needed by the VIOS. System configurations with 20–30 clients, many LUNs, and a generous vhost configurations might need increased memory to suit performance expectations.

If there is a possibility that you might start bridging more networks and use jumbo frames, use 1 GB of RAM for the VIOS. If you have enough memory to spare, the best situation is to use 1 GB for the VIOS to allow the maximum flexibility. When 10 Gbps Ethernet adapters are commonly implemented, this memory requirement might require revision. If VIOS is using IVE/HEA as an SEA, the LPAR might need more than 1 GB.

6.4.2 VIOS CPU sizing

The CPU planning has two major parts:

- The CPU required to support the virtual network
- The CPU required to support the virtual disk

CPU requirements to support the virtual network

When client partitions with virtual Ethernets need to access the external network, they need to bridge through the VIOS using the SEA so the VIOS sees all the network traffic flowing between the external network and the client partitions.

The CPU requirements on the VIOS for the network depends on the actual network traffic that runs through the VIOS and not on the actual number of network adapters installed on it. Minimal CPU is needed to support a network adapter card. For every network packet, however, the CPU has to calculate things such as the network checksum

Table 6-1 lists CPU requirements based on POWER5 system to push 1 GB network traffic. We can use this has a starting point.

Table 6-1 Approximate CPU amount VIOS needs for 1 GB of network traffic

MTU (bytes)	1500	9000 or jumbo frames
CPU speed (GHz)		
1.5	1.1	0.55
1.65	1.0	0.5
1.9	0.87	0.44
2.2	0.75	0.38

Let us see an example. Suppose that you have a VIOS with 4 gigabit network adapters and you know that during normal operations you have about 100 Mb of traffic overall. However, you also know that during a backup window you use a full gigabit of network traffic for two hours at night.

These values can be translated into a CPU requirement to support the network. This can best be done by using the shared CPU model. If we assume that your server has 1.65 GHz CPUs and you are using an MTU of 1500, we can calculate that during normal operation you only need 0.1 of a CPU. During peak loads, you need 1.0 CPU. If we assume that your user base is not using the system at night (thus the backup), there is plenty of unused CPU in the free pool that can be used for the CPU requirement here. You can configure the VIOS partition as a shared uncapped partition with 0.1 entitlement with 1 virtual CPU. This guarantees 0.1 of a CPU to sustain the daily network usage, but by using the uncapped CPU resources, we can allow the VIOS to grow to 1.0 CPU if required using spare CPU cycles from the CPU pool.

Remember that you need CPU to support network traffic and that adding additional network cards (providing the same network traffic) does not require additional CPU. Using Table 6-1 on page 258, estimate a value for the required network bandwidth to support normal operations. Guarantee this amount (plus the disk value from the following section) to the logical partition as the processing units. Use uncapped CPUs on the servers profile to allow the VIOS to use spare processing from the free pool to handle any spikes that might occur.

CPU requirements to support the virtual disk

The disk CPU requirement is more difficult to work out accurately because it involves knowing detailed information about your I/O, such as block sizes and number of I/Os per second. If you know this information, the IBM Systems Hardware Information Center has the detailed planning calculations.

For a rough guideline, it is probably easier to work out what the disks you are using can provide and make an estimate as to how busy you think these disks will be. For example, consider a simple VIOS that has a basic internal set of four SCSI disks. These disks are used to provide all of the I/O for the clients and are 10 K RPM disks. We use a typical workload of 8 KB blocks. For these disks, a typical maximum is around 150 I/Os per second, so this works out to be about 0.02 of a CPU. A small amount.

If we plot the amount of CPU required against the number of I/Os per second for the 1.65 GHz CPUs for the various I/O block sizes, we get a the data shown in Figure 6-6.

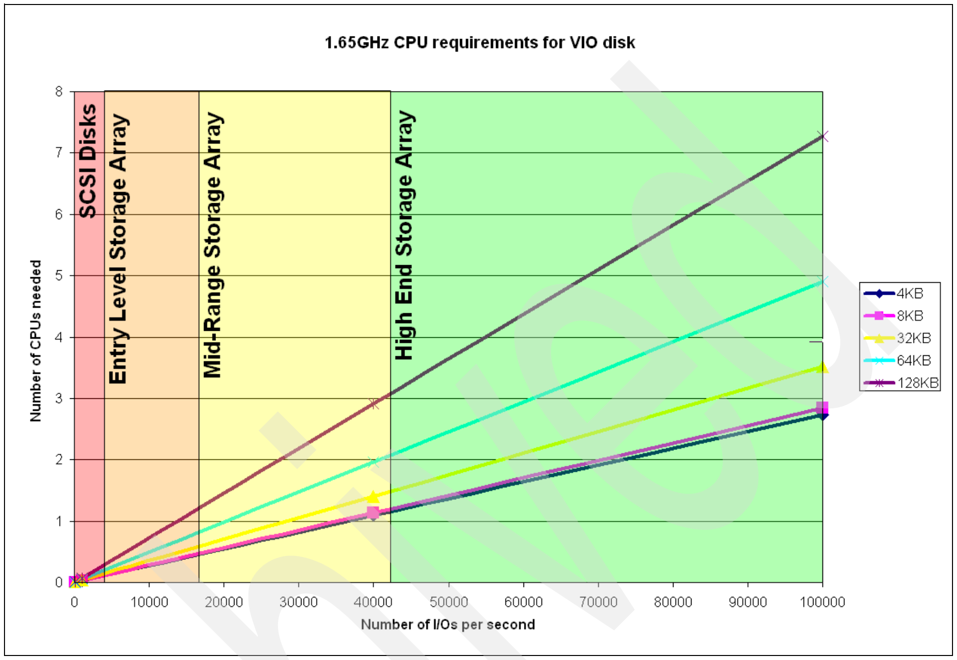


Figure 6-6   Estimated size of storage array to drive I/O versus VIOS CPU

These I/O numbers and the storage subsystem sizing assume that the storage subsystem is being driven at 100% by the virtual I/O clients only (so every I/O is virtual disk) and the disk subsystems have only been placed on the graph to indicate the performance you need to generate this sort of load.

It is important to remember that we are not saying the bigger the disk subsystem, the more CPU power you need. What we actually find is that we need the most powerful storage subsystem offered to require any significant amount of CPU in the VIOS from a disk usage perspective. As an example, a mid-range disk subsystem running at full speed is in the region of the yellow/green boundary (mid-range to high end) on the graph when configured with more than 250 drives. In most cases, this storage is not 100% dedicated to the VIOS. However, if this was the case, even when running at full speed, a Virtual I/O only needs one CPU for most block sizes.

Most I/O requests for systems are in the lower left section, so a starting figure of 0.1 to 0.2 CPU to cater for a disk is a good starting point in most cases.

Note: We always suggest testing a configuration before putting it into production.

6.4.3 Sizing dual VIOSs

If you are planning a system with dual VIOSs, you can size each VIOS to support the workload of one half of the clients. Use uncapped CPU to provide enough reserve CPU in case a VIOS is removed from the configuration for service.

6.4.4 Sizing VIOS for Live Partition Mobility

When configuring a VIOS to do an active Live Partition Mobility, consider the additional CPU that is required on the source and destination VIOS to do the migration. As discussed in Chapter 7, “Live Partition Mobility” on page 269, during an active Live Partition Mobility, the source VIOS transfers the client partitions configuration and memory to the destination VIOS over the network.

Because network traffic requires additional CPU, we need to size both VIOS adequately to minimize the migration time. The actual amount of CPU required varies depending upon network parameters and the processor used on the system. Therefore, it is recommended to configure both the VIOSs as shared uncapped partitions and allocate one additional VCPU on top of what is required for processing the required virtual I/O and virtual network requests. This allows the VIOSs to make use of available CPU from the free pool during the migration and reduces the total time required for doing the migration.

6.5 VIOS best practices for DB2

This section discusses the practices that are suggested in configuring the VIOS for servicing a client partition that runs a DB2 workload.

6.5.1 Multi-path I/O

Because I/O is important to a DB2 workload, we need to provide both redundancy and efficient bandwidth to do the I/O. This can be achieved by using multi-path I/O capability on the VIOS.

For DB2 DataWarehouse workloads, bandwidth might be the key requirement, whereas for DB2 OLTP workloads, the total number of I/Os that can be processed might be the key requirement. MPIO address both the requirements by making use of multiple adapters and providing multiple paths,

Multipathing for the physical storage in the VIOS provides failover physical path redundancy and load-balancing. The multipathing solutions available in the VIOS include MPIO, as well as solutions provided by the storage vendors.

The multipath I/O software needs to be configured only on the VIOS. There is no special configuration that needs to be done on the virtual I/O client partitions to make use of the multipath I/O at the VIOS.

For details on configuring multipath I/O, see the corresponding storage vendor's documentation.

6.5.2 Networking

Because a DB2 workload depends on network to talk to the clients, we need to provide both redundancy as well as efficient bandwidth for the network traffic too. This can be achieved by configuring link aggregation for the SEA adapter on the VIOS.

Link aggregation is a network port aggregation technology that allows several Ethernet adapters to be aggregated together to form a single pseudo-Ethernet adapter. This technology is often used to overcome the bandwidth limitation of a single network adapter and avoid bottlenecks when sharing one network adapter among many client partitions. For more information about the types of link aggregation technologies, refer to *PowerVM Virtualization on IBM System p: Introduction and Configuration Fourth Edition*, SG24-7940.

To provide redundancy in case of port or switch failure, for single VIOS configuration, we can further configure a NIB on the SEA adapter, as shown in Figure 6-7.

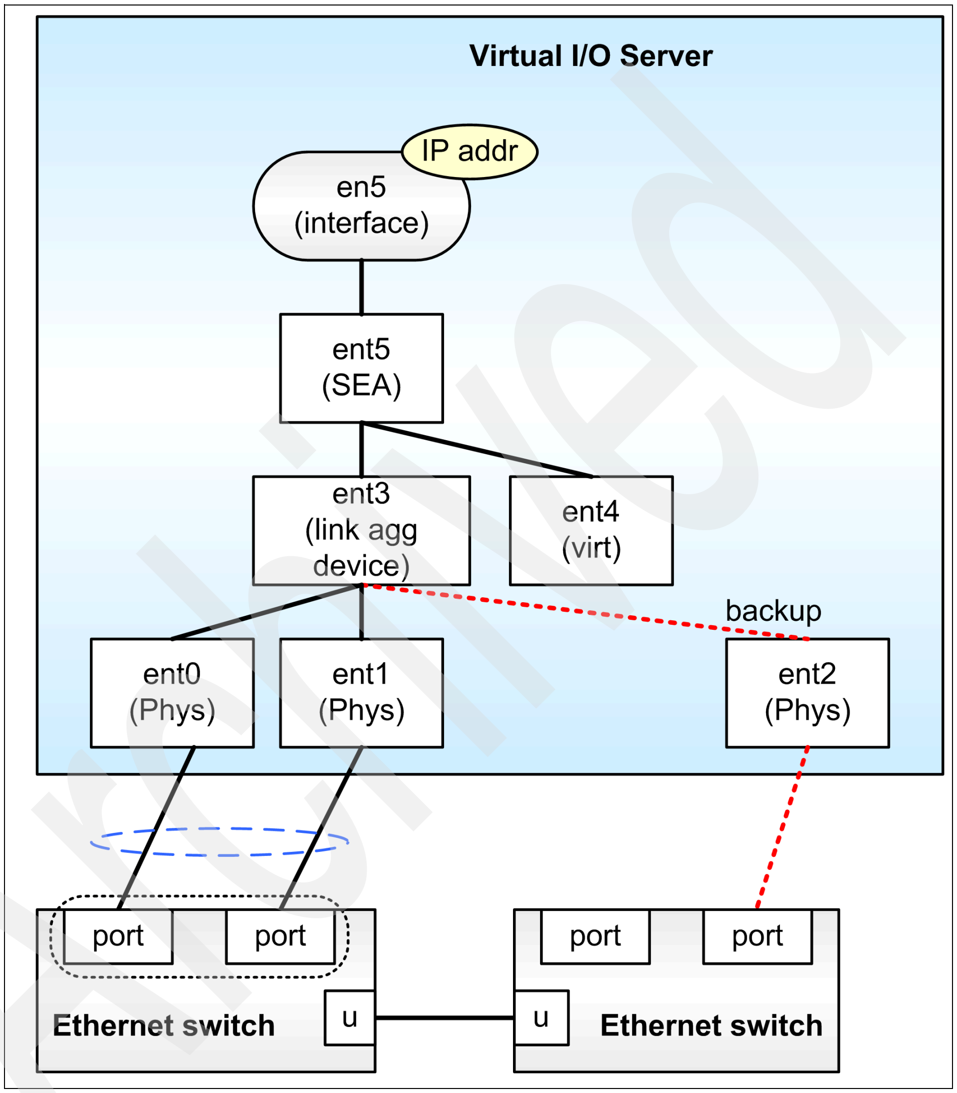


Figure 6-7 Link aggregation using a single VIOS

For dual VIOS configurations, redundancy can be provided by using either SEA failover or configuring a NIB on the client partition as discussed in“Virtual network redundancy” on page 253.

6.5.3 CPU settings for VIOS

If the I/O response time is not important for the DB2 workload running on the client partition, and if the CPU sizing requirement for the VIOS is less than 1.0 CPU entitlement, we can leave the CPU entitlement less than 1.0, but make it as a shared uncapped partition. This allows it to make use of available processing power in the free pool when required during peak load.

On the other hand, if the response time for the I/O is important for the DB2 workload, we need to make sure that the VIOS can get access to a CPU and service the I/O request. This can be done by either configuring the VIOS with a whole CPU entitlement (1.0) and making it a shared uncapped partition or, on a POWER6 system, we can configure it as Dedicate-Donate partition. Both these methods guarantee that the VIOS can immediately service a I/O request. Even though we are allocating more CPU entitlement than required for the VIOS, because we have configured it as shared or dedicate-donate, other partitions can make use of the unused capacity. We need to make sure the other LPARs are configured as shared-uncapped partitions with appropriate VCPUs to make use of the available processing capacity.

6.5.4 SCSI queue depth

Typically a DB2 workload makes use of RAID storage with multiple physical disks configured for each LUN. The default SCSI command queue depth of these LUNs, as seen by the VIOS, might not be sufficient. Increasing them might provide improved performance depending on the configuration. A general rule of thumb is to configure queue depth to eight times the number of physical disks.

Use the following command to change the queue depth on the VIOS:

```
chdev -dev hdiskN -attr queue_depth=x
```

Note: Changing the queue depth of the physical LUN might require changing the queue depth for the virtual disk on the client partition to achieve optimal performance.

There are several other factors to be taken into consideration when changing the queue depth. These factors include the value of the queue_depth attribute for all of the physical storage devices on the VIOS being used as a virtual target device by the disk instance on the client partition. It also includes the maximum transfer size for the virtual SCSI client adapter instance that is the parent device for the disk instance.

The maximum transfer size for virtual SCSI client adapters is set by the VIOS, which determines that value on the resources available on that server and the maximum transfer size for the physical storage devices on that server. Other factors include the queue depth and maximum transfer size of other devices involved in mirrored volume group or MPIO configurations. Increasing the queue depth for a few devices might reduce the resources available for other devices on that same parent adapter and decrease throughput for those devices.

The most straightforward configuration is when there is a physical LUN used as the virtual target device. For the virtual SCSI client device queue depth to be used effectively, it must not be any larger than the queue depth on the physical LUN. A larger value wastes resources without additional performance. If the virtual target device is a logical volume, the queue depth on all disks included in that logical volume must be considered. If the logical volume is being mirrored, the virtual SCSI client queue depth must not be larger than the smallest queue depth of any physical device being used in a mirror.

When mirroring, the LVM writes the data to all devices in the mirror, and does not report a write as completed until all writes have completed. Therefore, throughput is effectively throttled to the device with the smallest queue depth. This applies to mirroring on the VIOS and the client.

We suggest that you have the same queue depth on the virtual disk as the physical disk. If you have a volume group on the client that spans virtual disks, keep the same queue depth on all the virtual disks in that volume group. This is important if you have mirrored logical volumes in that volume group, because the write does not complete before the data is written to the last disk.

In MPIO configurations on the client, if the primary path has a much greater queue depth than the secondary, there might be a sudden loss of performance as the result of a failover.

The virtual SCSI client driver allocates 512 command elements for each virtual I/O client adapter instance. Two command elements are reserved for the adapter to use during error recovery. Three command elements are reserved for each device that is open to be used in error recovery. The rest are left in a common pool for use in I/O requests. As new devices are opened, command elements are removed from the common pool. Each I/O request requires one command element for the time that it is active on the VIOS.

Increasing the queue depth for one virtual device reduces the number of devices that can be open at one time on that adapter. It also reduces the number of I/O requests that other devices can have active on the VIOS.

6.5.5 Dual VIOS

To provide the DB2 workload running on the client partition high availability to virtual I/O and virtual network, it is suggested to configure and use dual VIOSs as discussed in 6.3.1, “Dual VIOS” on page 251.

6.5.6 SEA threading

The VIOS enables you to virtualize both disk and network traffic for the virtual I/O clients. The main difference between these types of traffic is the persistence of them. If the VIOS has to move network data around, it must do this immediately because network data has no persistent storage. For this reason, the network services provided by the VIOS (such as the SEA) run with the highest priority. Disk data for virtual SCSI devices is run at a lower priority than the network because the data is stored on the disk and there is less of a danger of losing it due to time outs. The devices are also normally slower in speed.

The shared Ethernet process of the VIOS prior to Version 1.3 runs at the interrupt level that was optimized for high performance. With this approach, it ran with a higher priority than the virtual SCSI if there was high network traffic. If the VIOS did not provide enough CPU resource for both, the virtual SCSI performance can experience a degradation of service.

With VIOS Version 1.3, the shared Ethernet function can be implemented using kernel threads. This enables a more even distribution of the processing power between virtual disk and network.

This threading can be turned on and off per SEA by changing the thread attribute and can be changed while the SEA is operating without any interruption to service. A value of 1 indicates that threading is to be used and 0 indicates the original interrupt method:

```
$ chdev -dev ent2 -attr thread=0
```

The performance difference without threading works out at around 8% less (using our intensive test loads) CPU needed for the same network throughput. With the burst nature of network traffic, however, we suggest enabling threading (this is the default). By this, we mean that network traffic comes in spikes, as users log on or as Web pages load, for example. These spikes might coincide with disk access. For example, a user logs on to a system, generating a network activity spike, because during the logon process some form of password database stored on the disk is most likely accessed, or the user profile read.

The one scenario where you must consider disabling threading is where you have a VIOS dedicated for network and another dedicated for disk. This is only recommended when mixing extreme disk and network loads together on a CPU constricted server.

As discussed in 6.4, “VIOS sizing” on page 257, usually the network CPU requirements are greater than the disk. In addition, you probably have the disk VIOS setup to provide a network backup with SEA failover if you want to remove the other VIOS from the configuration for scheduled maintenance. In this case, you have both disk and network running through the same VIOS, so threading is recommended.

6.5.7 Best practices summary

THe following list details the best practices in configuring VIOS and the client LPAR for running a DB2 workload using virtual I/O.

- For storage access redundancy and bandwidth, configure MPIO on the VIOS to access the storage
- For network access redundancy and bandwidth, use link aggregation of network adapters to configure the SEA adapter.
- For network access redundancy on the client LPAR, use NIB or SEA failover with dual VIOS configuration.
- VIOS requires modest CPU to process virtual I/O request. However, if the DB2 workload is sensitive to the I/O latency, configure the VIOS as a shared uncapped partition with a whole CPU entitlement, or on a POWER6 system, configure it as Dedicate-Donate partition. This results in faster scheduling of I/O operations and therefore improved Virtual I/O performance.
- The queue depth of the virtual disks on the client LPAR and the physical disks on the VIOS is set properly to achieve optimal performance. The default value is often inadequate for most database configurations
- For high availability of virtual I/O and virtual network for the client LPAR, it is recommended to configure and use dual VIOSs
- Memory required for VIOS partition is insignificant. Allocating more memory does not result in any performance improvement
- When having a mix of both heavy virtual I/O and virtual network requirements, it might benefit to configure a separate VIOS dedicated for the virtual network and turning off threading on the SEA.

7

Live Partition Mobility

Live Partition Mobility (LPM) allows you to migrate partitions that are running AIX and Linux operating systems and their hosted applications from one physical server to another without disrupting the infrastructure services. The migration operation, which takes a few seconds, maintains complete system transactional integrity. The migration transfers the entire system environment, including processor state, memory, attached virtual devices, and connected users. LPM requires a strict environment. Numerous pre-requisites need to be met before being able to migrate a partition (active or inactive migration) to another system.

Other Power technology-based processor types (Power 6 systems), PowerHA and N-PIV attached storage are among the new technologies that are now supported with LPM. For more information about PowerVM features, see 1.5, “Introduction to PowerVM virtualization” on page 21.

This chapter covers LPM pre-requisites in detail. We describe the migration of a logical partition managed by two HMCs, each one managing both servers. This chapter is not meant to describe how to configure LPM but to describe the best practices to configure the environment for a DB2 setup.

We cover the following topics in this chapter:

- “LPM overview” on page 270
- “LPM” on page 271
- “DB2 migration-awareness” on page 273
- “System planning” on page 274
- “Migration process and flow” on page 316
- “Mobility in action” on page 322
- “LPM: Summary of actions” on page 354

7.1 LPM overview

Power VM LPM allows you to migrate running and powered-off AIX partitions and their hosted applications from one POWER6 box to another with minimal disruptions to the applications hosted on these platforms and without shutting down the servers.

7.1.1 Active migration

Using active migration, a running partition is moved from a source system to a destination system with no disruption of partition operation or user service.

An active migration performs the same operations as an inactive migration except that the operating system, the applications, and the services they provide are not stopped during the process. The physical memory content of the logical partition is copied from system to system allowing the transfer to be imperceptible to users.

During an active migration, the applications continue to handle their normal workload. Disk data transactions, running network connections, user contexts, and the complete environment are migrated without any loss. Migration can be activated any time on any production partition. There is no limitation on a partition's computing and memory configuration. Multiple migrations can be executed concurrently. Both inactive and active migrations might involve partitions with any processing unit and memory size configuration.

7.1.2 Inactive migration

Inactive migration moves the definition of a powered off logical partition from one system to another, along with its network and disk configuration. No additional change in network or disk setup is required and the partition can be activated as soon as migration is completed.

The inactive migration procedure takes care of the re-configuration of involved systems.

- A new partition is created on the destination system with the same configuration present on the source system.
- Network access and disk data is preserved and made available to the new partition.
- On the source system, the partition configuration is removed and all involved resources are freed.

If a system is down due to scheduled maintenance or not in service for other reasons, an inactive migration might be performed. It is executed in a controlled way and with minimal administrator interaction so that it can be safely and reliably performed in a short time frame. When the service provided by the partition cannot be interrupted, its relocation can be performed with no loss of service by using the active migration feature.

Extending this technology to DB2 and its server installations, we can make use of LPM to move a database server, without shutting down the database engine, from one physical system to a second physical system while it is servicing transactions. LPM uses a simple and automated procedure to migrate the entire system environment, including processor state, memory, attached virtual devices, and connected users with no application downtime.

7.2 LPM

Infrastructure flexibility has become a key criteria when designing and deploying information technology solutions. Imagine you have a DB2 database server running on a hardware system that has exhausted all its resources. The server is running 2 processors at 100% use. Response times are slow and users are complaining.

- Move that DB2 database server from that system to a system that has additional resources without interrupting service to the users
- Perhaps the system needs to go down for scheduled maintenance such as a hardware upgrade or a firmware upgrade or even a component replacement?
- You need to do a new system deployment so that the workload running on an existing system is migrated to a new, more powerful one.
- If a server indicates a potential failure, you can move its logical partitions to another system before the failure occurs.
- You want to conserve energy by moving a partition to another system during off-peak periods.

How can we do this?

The answer is LPM. With LPM, it is possible to meet continuously stringent service level agreements, efficiently manage the infrastructure, and minimize the impact to users.

Note: LPM can move an entire Logical Partition from one system to another while it is running, with almost no impact to users. It can move the entire LPAR, including the OS. It requires a minimum of a POWER6 system based on the POWER6 processor, PowerVM Enterprise Edition, and all I/O must be through the VIOS.

However, while LPM provides many benefits, it does not perform the following tasks:

- LPM does not do automatic load balancing.
- LPM does not provide a bridge to new functions. Logical partitions must be restarted and possibly reinstalled to take advantage of new features.
- LPM does not protect you from system failures, so it does not replace high-availability software such as the IBM HACMP high-availability cluster technology.

Note: To summarize, LPM is not:

- A replacement for DB2 HADR (high availability disaster recovery) or Tivoli Systems Automation for Multiplatforms
- Automatic
- A disaster recovery solution

To use the partition migration, there are planning steps that are necessary. Although an in-depth discussion on the technologies that supports the flexibility is outside of the scope of this chapter, a quick checklist is provided that assures your partition is able to migrate successfully.

Note: To summarize, LPM benefits are:

- Eliminates planned outages
- Balances workloads across systems
- Energy savings

7.3 DB2 migration-awareness

A migration-aware application is one that is designed to recognize and dynamically adapt to changes in the underlying system hardware after being moved from one system to another.

There are no changes required to make DB2 work when the partition is moved from one system to another. DB2 has built-in autonomics and self-tuning that enables it to adapt to changes in the underlying system after being moved from one system to another:

- Self tuning memory manager (STMM) feature of DB2 queries the operating system for free and available memory at regular intervals and automatically adapts to the destination server's memory to maximize throughput.
- DB2 has the ability to change several parameters, without instance restart.
- DB2 uses OS-based Scheduler and can automatically adapt to the new CPU resources.

Note: Most DB2 parameters are set to AUTOMATIC by default. The general guideline is to leave these parameters AUTOMATIC.

Table 7-1 shows a list of default parameters used by DB2 version 9.7.

Table 7-1 List of default parameters (DB2 9.7) – plus many more

List of default DB2 9.7 parameters	**Parameter**	**Value**
Size of instance shared memory (4 KB)	INSTANCE_MEMORY	AUTOMATIC
Self tuning memory	SELF_TUNING_MEM	ON
Max storage for locK list (4 KB)	LOCKLIST	AUTOMATIC
Percentage of lock lists per application	MAXLOCKS	AUTOMATIC
Package cache size (4 KB)	PCKCACHESZ	AUTOMATIC
Sort heap threshold for shared sorts (4 KB)	SHEAPTHRES_SHR	AUTOMATIC
Sort list heap (4 KB)	SORTHEAP	AUTOMATIC
Number of asynchronous page cleaners	NUM_IOCLEANERS	AUTOMATIC
Number of I/O servers	NUM_IOSERVERS	AUTOMATIC
Default prefetch size (pages)	DFT_PREFETCH_SZ	AUTOMATIC

List of default DB2 9.7 parameters	Parameter	Value
Max number of active applications	MAXAPPLS	AUTOMATIC
Average number of active applications	AVG_APPLS	AUTOMATIC

The performance characteristics (throughput, response time) of your database workloads might change depending on the resources of the target server.

7.4 System planning

This section highlights the steps and actions to perform a Live Partition migration of a logical partition running a DB2 database (for example, from one system to another) without interrupting the service to the client. For an in-depth information and installation steps, refer to the IBM Redbooks publication *IBM PowerVM Live Partition Mobility*, SG24-7460.

Numerous pre-requisites need to be met prior to migration. After all pre-requisites are met and satisfied, the HMC validates the migration. If the result of the validation is successful, the migration can be performed either using the GUI or the command line interface of the HMC.

In the following sections pre-requisites, configuration, and post migrations are covered.

Note: The major part of this chapter covers systems managed by one or more HMCs. Although this book covers systems managed by HMC, the tests on LPM have been done with IVM-based systems. For information about IVM, refer to the IBM Redpaper™ *Integrated Virtualization Manager on IBM System p5*, REDP-4061.

7.4.1 Managed system's requirements

Table 7-2 lists the managed system's requirements.

Table 7-2 Managed system's requirements

Managed system's tasks	Actions to undertake and remarks	For details see
Managed systems type	POWER6 systems and Power Blade systems	"Managed system type" on page 275
Managed system's capabilities	PowerVM APV EE	"Managed system's capabilities" on page 276
Processor clock speed	Mix of clock is supported	"Processor clock speed" on page 278
Time-of-day	Synchronize time-of-day	"Time-of-Day" on page 280
Managed system's minimum firmware level	EH330_046 (p595) or EM320_31 (p570) or EL320_040 (p520 + p550)	"Managed system's firmware level" on page 281
Available memory on destination system	Sufficient resources on destination	"Available memory on destination system" on page 281
Available processor or processing units on destination system	Sufficient resources on destination system	"Available processor on destination system" on page 281
Existing profile	Mobile profile not on destination system	"Existing profile" on page 282
Managed systems' connectivity	Single or dual redundant HMC, separate HMC	"Managed systems' connectivity" on page 283
Logical Memory Block	Must be identical on both systems	"Logical Memory Block (LMB)" on page 284
Battery power	Supported on source system, not destination system	"Battery power" on page 284

Managed system type

Managed systems are those systems able to host a migrating partition. These are Power architectured systems, such as Power System p6 or Power Blade.

Managed servers in an LPM must be of same type (for example source and target servers are POWER6 servers or Power Blades). The same HMC or redundant pair of HMCs are used with Power Systems. IVM is used for Power

Blades and for Power Systems (models below p570). If you want to mix POWER6 servers and Power blades, this is possible if the systems are managed by IVM. Each of the servers can act as a source or a target system to the other as long as it contains the necessary processor hardware to support it. This is called the *migration support.* Of course, to be able to use the LPM a minimum of two systems is necessary.

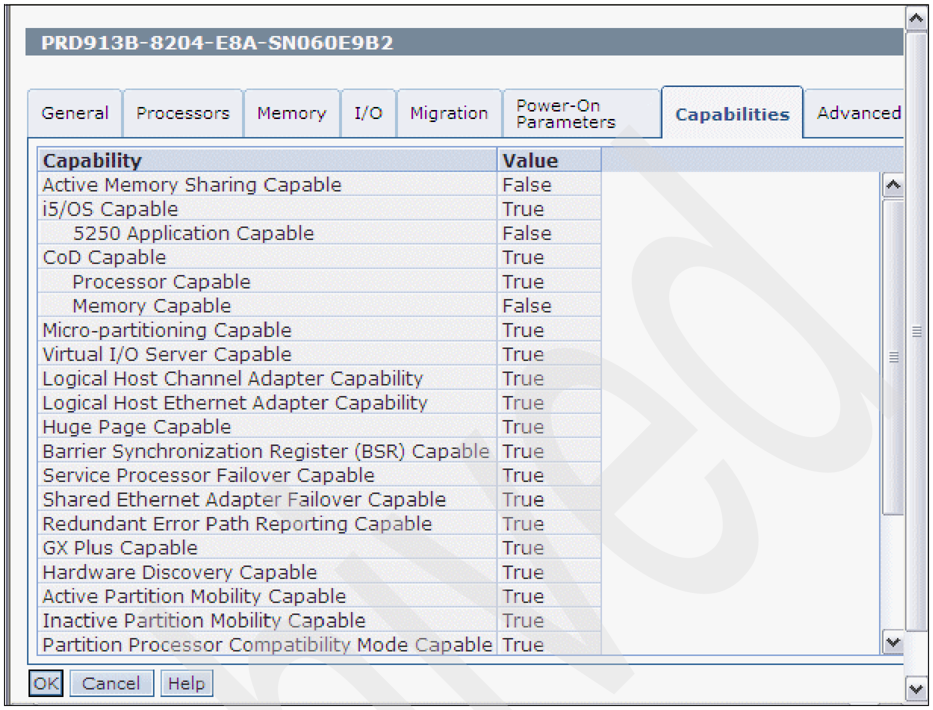
Managed system's capabilities

The Power Hypervisor is the foundation of the PowerVM. The combination of Power processor architecture and the Power Hypervisor offers multiple capabilities among micro-partitioning, virtual processors, virtual Ethernet, virtual SCSI, virtual console. It is, in essence, always active and cannot be de-activated. For more information about the Power Hypervisor, refer to *PowerVM Virtualization on IBM System p: Introduction and Configuration Fourth Edition*, SG24-7940

To perform LPM migrations, the servers must be enabled to do so. Therefore, a valid Advanced Power Virtualization Enterprise Edition or APV EE licence needs to be activated on both the source and target servers. Those capabilities must at least be active for:

- VIOS Capable TRUE
- Active Partition Mobility TRUE
- Inactive Partition Mobility TRUE

Figure 7-1 shows one possible output.



Capability	Value
Active Memory Sharing Capable	False
i5/OS Capable	True
5250 Application Capable	False
CoD Capable	True
Processor Capable	True
Memory Capable	False
Micro-partitioning Capable	True
Virtual I/O Server Capable	True
Logical Host Channel Adapter Capability	True
Logical Host Ethernet Adapter Capability	True
Huge Page Capable	True
Barrier Synchronization Register (BSR) Capable	True
Service Processor Failover Capable	True
Shared Ethernet Adapter Failover Capable	True
Redundant Error Path Reporting Capable	True
GX Plus Capable	True
Hardware Discovery Capable	True
Active Partition Mobility Capable	True
Inactive Partition Mobility Capable	True
Partition Processor Compatibility Mode Capable	True

Figure 7-1 Server capabilities

Notes:

- If the capability you are looking for is not displayed, this means that your HMC is down level or the HMC does not have the appropriate fix pack. See 7.4.2, “HMC requirements” on page 285.
- If the capability you are looking for is shown as False, the Power Hypervisor enablement has not been performed or the firmware is downlevel.

If your server is not configured with the Enterprise Edition, you need to enable your system. Contact your IBM representative to do so.

Processor clock speed

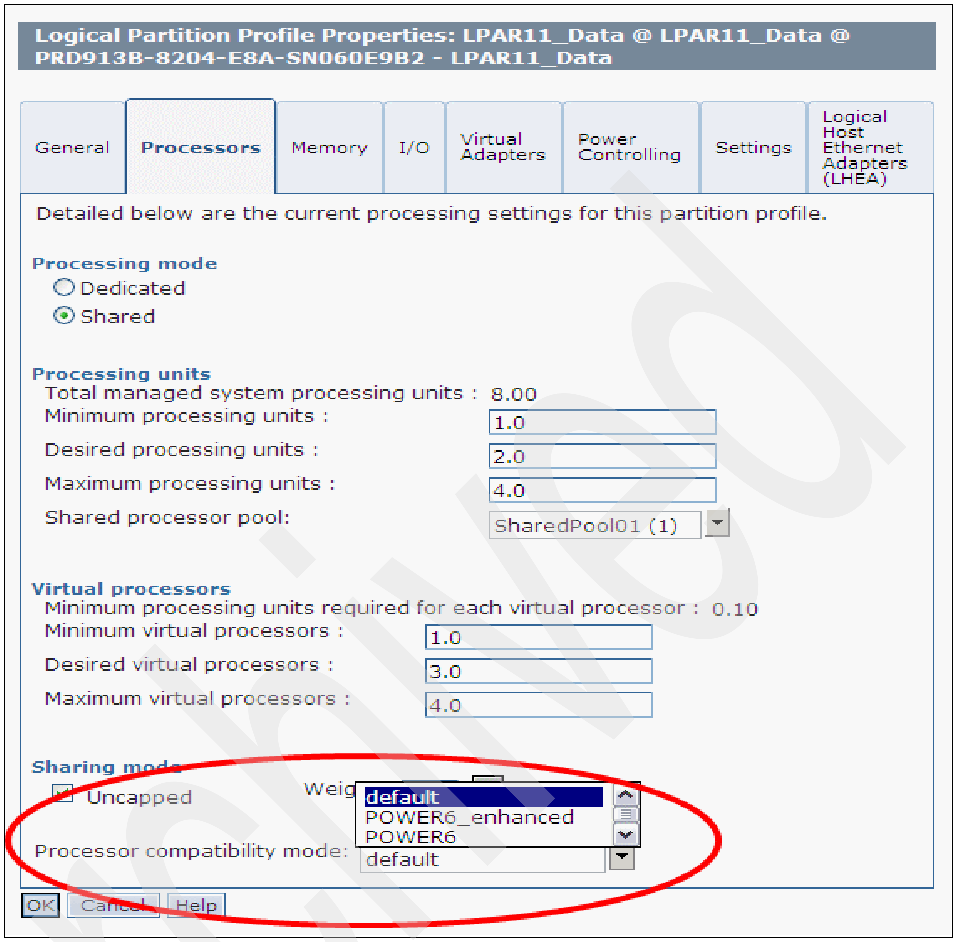
LPM can be used between managed systems with differing clock speeds. For example, the source server can have 4.7 GHz processors and the target can be configured with 5.0 GHz processors. This is also known as processor compatibility mode.

A processor compatibility mode is a value assigned to a logical partition by the Power Hypervisor, which specifies the processor environment on which the logical partition can successfully operate. Processor compatibility mode can be set into the partition's profile. You can configure modes as shown in Figure 7-2 on page 279, as follows:

- POWER6 mode or default mode
- POWER6 enhanced mode

  This mode provides additional floating point instructions to applications running on the logical partition.
- POWER5 mode

  Typically when operating system's version installed on the partition does not support POWER6 processor mode.


Logical Partition Profile Properties: LPAR11_Data @ LPAR11_Data @ PRD913B-8204-E8A-SN060E9B2 - LPAR11_Data

Detailed below are the current processing settings for this partition profile.

Processing mode
[ ] Dedicated
[✓ Shared]

Processing units

Total managed system processing units :	8.00
Minimum processing units :	1.0
Desired processing units :	2.0
Maximum processing units :	4.0
Shared processor pool:	SharedPool01 (1)

Virtual processors

Minimum processing units required for each virtual processor :	0.10
Minimum virtual processors :	1.0
Desired virtual processors :	3.0
Maximum virtual processors :	4.0

Sharing mode
[✓ Uncapped] Weig

Processor compatibility mode: default

Figure 7-2 Processor compatibility mode

The processor compatibility mode is checked by the managed system across the partition's profile when the partition is activated and determines whether the installed operating system supports this mode. If not, the partition uses the most fully featured mode that is supported by the operating system.

Note: Processor compatibility mode cannot be changed dynamically. You need to shut down the partition, modify the partition's profile to the desired mode, and restart the partition.

Note: A POWER6 processor cannot emulate all features of a Power5 processor.

The compatibility mode is important when using LPM. When you move a logical partition from one system to another that has a different processor type, the processor compatibility mode enables that logical partition to run in a processor environment on the destination system in which it can successfully operate.

For more information, see *PowerVM Virtualization on IBM System p: Introduction and Configuration Fourth Edition*, SG24-7940.

Time-of-Day

Although not mandatory, it is suggested to synchronize the time-of-day between the source and the destination system, be it for active or inactive partition mobility. Therefore, a new attribute, the Time Reference, is available in the settings tab of the properties of the partition. Any VIOS partition can be designated as a Time Reference Partition (TRP). Values for the TRP is enable or disable (default). Changes take effect as soon as you click **OK**.

Important: It is important to have a the complete environment's time and date synchronized for DB2 to remain coherent. You can imagine the problems that can arise when, after the migration of the logical partition from one system to another, the time and date suddenly went back in the past. This causes a major database corruption.

Notes:

- Time synchronization is a suggested step in the active partition migration. If you chose not to synchronize using the TRP attribute, the source and target systems synchronizes the clocks during the migration process from the source server to the destination server.
- The TRP capability is only supported on systems capable of active partition migration.
- Other TRPs are supported by server. The longest running TRP is recognized as the TRP of the system.
- Ensure that the HMC and all the partitions have the same date and time to avoid discordances and errors when migrating to another system.

Managed system’s firmware level

Firmware level of both systems acting in a LPM needs to be at a minimum level of EH330_046 for Power System p595, EM320_31 for Power Systems p570 servers or EL320_040 for Power Systems p520 and p550 servers. PowerBlades systems need a firmware level of Es320 or later.

Recommendation: It is suggested to have your system at the last firmware level, because it contains the last features and options that you can use and the latest correctives.

Refer to the Power code matrix at the following Web page for more information:

`http://www14.software.ibm.com/webapp/set2/sas/f/power5cm/power6.html`

Note: Firmware level of source and target systems might defer. However, the level of the source system firmware must be compatible with the target server.

Refer to the following Web page for more information:

`http://www14.software.ibm.com/webapp/set2/sas/f/pm/migrate.html`

To upgrade the firmware of your managed system, refer to the IBM fixes web site:

`http://publib.boulder.ibm.com/infocenter/systems/scope/hw/index.jsp?topic=/ipha5/fix_serv_firm_kick.htm`

Recommendation: Firmware level E*340_039 is recommended for all systems.

Available memory on destination system

Ensure that the destination system has as at least as much memory as the source system to support the migration. If necessary, use the DLPAR capability of your system to free memory.

Available processor on destination system

Ensure that the destination system has at least as many processors or processor unit resources to support the migration. Because the DB2 client profile from the source system is copied to the destination system, ensure that you have the requested dedicated processors or processing units to re-construct the DB2 client profile on the destination server.

Existing profile

Determine the name of the logical partition profile for the mobile partition on the destination system. This is an optional step. As part of the migration process, the HMC creates a new migration profile containing the partition's current state. You must also verify that no Logical Host Ethernet Adapter (LHEA) devices are configured, because these are also considered as physical I/O. Inactive migration is still possible if LHEA are configured. 78 IBM PowerVM LPM replaces the existing profile that was last used to activate the partition.

Also, if you specify an existing profile name, the HMC replaces that profile with the new migration profile. If you do not want the migration profile to replace any of the partition's existing profiles, you must specify a unique profile name. The new profile contains the partition's current configuration and any changes that are made during the migration.

When migrating a mobile partition to another system, an LPAR with the same name as the source LPAR might not exist on the destination system. Check the HMC console or through the command line on the HMC using the **lssyscfg** command. See Example 7-1 as an example.

Example 7-1 Output of the lssyscfg command

```
hscroot@:~> lssyscfg -r lpar -F name,msp,state -m
PRD913B-8204-E8A-SN060E9B2
VIOS_Beta,0,Not Activated
VIOS_Alpha,0,Not Activated
LPAR13-Etl,0,Not Activated
LPAR12_Data,0,Running
LPAR11_Data,0,Running

hscroot@:~> lssyscfg -r lpar -F name,msp,state -m
PRD913B-8204-E8A-SN0653E02
lpar1-data,0,Running
```

In Example 7-1, each defined LPAR is only defined on one single system.

Managed systems' connectivity

Next, we discuss managed systems' connectivity.

Single or dual HMC setup

When going for LPM, both the source and the target servers need to be connected to the same HMC or redundant pair of HMCs, as illustrated in Figure 7-3. They also need to be connected on the same network so they can communicate with each other. Only HMC-to-HMC managed systems or IVM-to-IVM managed systems can intervene in the migration.

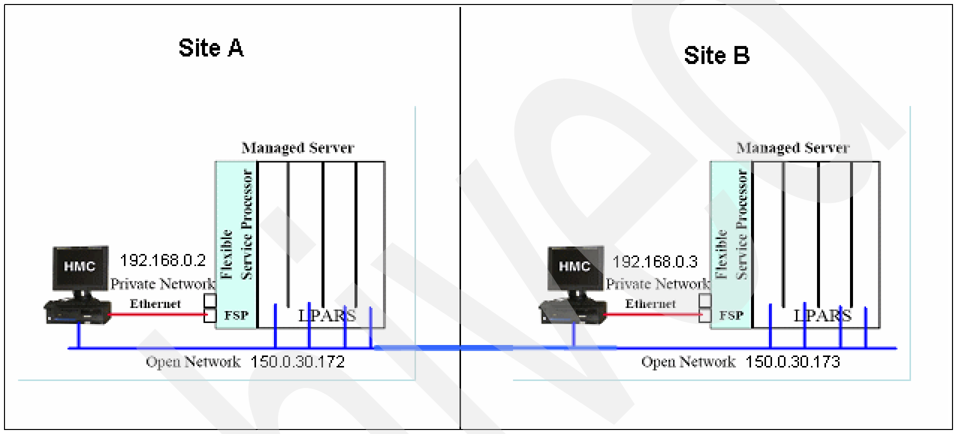


Figure 7-3 Dual HMC setup

Use the **lssyscfg** command to verify that both servers are seen by the same HMC, as shown in Example 7-2.

Example 7-2 Output of the command lssyscfg

```
hscroot@:~>lssyscfg -r sys -F name,type_model
PRD913B-8204-E8A-SN060E9B2,8204-E8A
PRD913B-8204-E8A-SN0653E02,8204-E8A
```

Separate HMC

Using one HMC per server is another supported environment for LPM. This is known as *Remote LPM*. Pre-requisites for Remote LPM are covered in 7.4.2, "HMC requirements" on page 285.

Note: For information about remote partition mobility, refer to IBM Redbooks publication *IBM PowerVM Live Partition Mobility*, SG24-7460.

Resource monitoring and control (RMC) is a feature that can be configured to monitor resources such as disk space, CPU usage, and processor status and allows performing an action in response to a defined condition. Remote migration operations require that each HMC has RMC connections (see "RMC connections" on page 298) to its individual system's VIOSs and a connection to its system's service processors. The HMC does not have to be connected to the remote system's RMC connections to its VIOSs, nor does it have to connect to the remote system's service processor

Logical Memory Block (LMB)

The logical memory block, LMB, must be identical on both the source and the target server. The default LMB size depends on the amount of memory installed in the CEC. It varies between 16 MB and 256 MB.

Use the following guidelines when selecting logical block sizes:

- On systems with a small amount of memory installed (2 GB or less), a large logical memory block size results in the firmware consuming an excessive amount of memory. Firmware must consume at least 1 logical memory block.

  As a general rule, select the logical memory block size to be no greater than 1/8th the size of the system's physical memory.

- On systems with a large amount of memory installed, small logical memory block sizes result in a large number of logical memory blocks. Because each logical memory block must be managed during boot, a large number of logical memory blocks can cause boot performance problems.

  As a general rule, limit the number of logical memory blocks to 8 K or less.

Note: The logical memory blocksize can be changed at run time, but the change does not take effect until the system is restarted.

Battery power

Ensure that the target system is not on battery power. If so, bring the target system in a stable state prior to migrate.

Note: The source system can run on battery power even if you want to migrate your partition to another system.

7.4.2 HMC requirements

Table 7-3 summarizes HMC requirements.

Table 7-3 HMC requirements

HMC tasks	Actions to undertake and remarks	For details see
HMC model	► 7310-CR2 desktop ► 7310-C03 rack-mount	"HMC model and version" on page 285
HMC level (single or redundant dual environment)	► 7.3.2.0 or later ► fix MH01062	"HMC model and version" on page 285
HMC level in remote LPM	7.3.4 or later	"HMC model and version" on page 285
Non-symmetric environments	► 7.3.5 or later ► VIOS 2.1.2.10 FP22	"HMC model and version" on page 285
HMC connectivity	Interconnected	"HMC model and version" on page 285

HMC model and version

Two types of HMC can be used in an LPM environment. You can use either a desktop model 7310-CR2 or later, or a rack-mounted model 7310-C03. See Figure 7-4.

Figure 7-4 Desktop and rack-mount HMCs

Both servers need to be connected to the same HMC or redundant pair of HMCs as illustrated in Figure 7-3 on page 283. They also must be connected on the same network so they can communicate with each others, as shown in Figure 7-3 on page 283.

Notes:

- HMC can perform multiple LPM migrations simultaneously.
- IVM is also supported to perform LPM migrations.

For the LPM to be successful in a single or dual redundant HMC environment, the HMCs' version need to be at a minimum level of 7.3.2 or later with required fix MH01062. Each HMC can see all the managed systems. Both HMC in a redundant dual HMC environment need to be at the same level.

For information about how to upgrade your HMC, refer to the following Web page:

http://www-933.ibm.com/support/fixcentral/

Tip: Migrations across a non-symmetric HMC configuration is possible if you meet the following requisites:

- HMC is at level 7.3.5 or later
- GUI "override errors" option or command line with *-force* flag

For example, migration of a logical partition to a destination server whose VIOS does not provide the same level of redundancy as the source system.

- Source system has two redundant VIOS serving the mobile partition through two SCSI channel
- Destination system only has one VIOS with only one HBA.

In other configurations, the source server can be connected to one HMC, whereas the target server can be connected to another HMC. LPM operations are possible. This situation where a logical partition needs to migrate from one server to another server is known as *remote LPM*. To perform a remote LPM ensure that:

- The source and target HMCs are connected to the same network so they can communicate with each other.
- The source and target HMC's version need to be at level 7.3.4 or later.
- To display the HMC version, use the `lshmc -V` command,
- A secure shell (SSH) must be established between both the source and the target HMC. For more information about how to set up SSH keys authentication refer to the following Web page:

  http://publib.boulder.ibm.com/infocenter/powersys/v3r1m5/index.jsp

- The migration of the mobile partition is managed by the HMC containing the mobile partition.

7.4.3 IVM requirements

IVM is closely tied to the VIOS media support you have. Each VIOS media support contains indeed the IVM package you need to install on your system to manage your partitions.

Therefore the requirements on IVM are tied to the VIOS level requirements. You can refer to VIOS Requirements, part “VIOS profile” on page 288 for more information.

Note: IVM actually helps to mask the complexity of the underlaying commands that run at VIOS level. However for particular reason among requested by the IBM support, you might need to run commands at VIOS level.

7.4.4 VIOS requirements

Table 7-4 lists the VIOS requirements.

Table 7-4 VIOS requirements

VIOS asks	Actions to undertake and remarks	For details, see
VIOS level	► VIOS 1.5.2.1-FP11 ► Fixes	“Operating system level” on page 288
VIOS profile	► Mover Service Partition (not needed for inactive migration) ► Destination server – Not on power battery – Not containing existing profile – Has sufficient resources (CPU, memory, # virtual slots) ► CPU & memory ► VASI Interface (not needed for inactive migration) ► Virtual SCSI ► IVE ► Virtual Ethernet ► Virtual slots ID higher than 11 ► Time and date synchronization (optional for active migration, not needed for inactive migration)	“VIOS profile” on page 288
Dual VIOS	► Symmetric environment recommended ► Support for non-symmetric environment if VIOS 2.1.2.10-FP22	“Dual VIOS” on page 292
Disk mapping to client	Only disk mapping	“Disk mapping to client partition” on page 293

For VIOS settings recommendations, refer to Chapter 6, “Virtual I/O” on page 241.

Operating system level

The VIOS needs to be at a minimum level of 1.5 with its limited released fixes (VIOS 1.5.2.1-FP11) to migrate a logical partition successfully.

Tip: Version 2.1.2.10 FP22 of the VIOS introduced a new functionality that consists of the preservation of the customized VTD names, support for non-symmetric environments, and multiple IP addresses for the VIOS configured as a MSP partition.

All VIOS need to be at the same release level.

For more information about how to change your current VIOS level, refer to the following Web page:

http://www14.software.ibm.com/webapp/set2/sas/f/vios/download/home.html

Tip: The client profile must only be created on the source system. The profile on the destination system is created automatically during the migration process.

VIOS profile

This section discusses the VIOS profile.

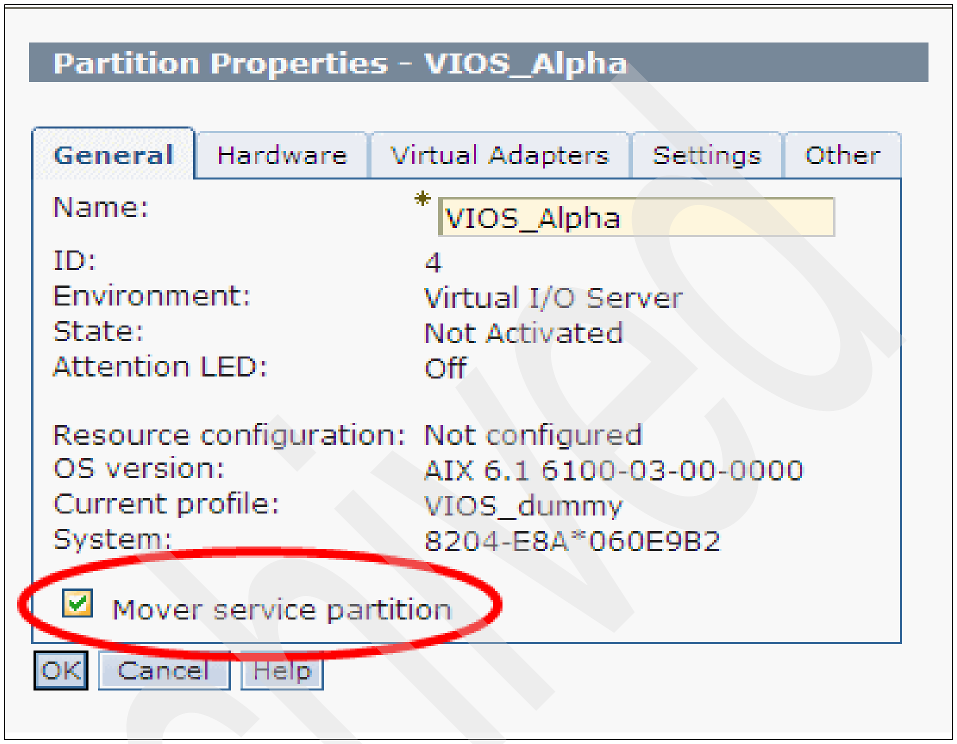
Mover Service Partition

Mover Service Partition (MSP) is an attribute of the VIOS partition. It enables the VIOS partition to allow the function that asynchronously extracts, transports, and installs a partition state. Two mover service partitions are involved in an active migration: one on the source system and one on the destination

There must be at least one VIOS declared as a mover service partition per system. A mover VIOS partition is the specific ability of that partition to interact with the Power Hypervisor and to migrate a mobile partition from the source VIOS to the target VIOS.

The mover service partition attribute can be configured either when you configure your VIOS profile (refer to IBM Redbooks publication *PowerVM Virtualization on IBM System p: Introduction and Configuration Fourth Edition*, SG24-7940 for more information) or after the VIOS has been configured by changing its properties. You can proceed as described in Figure 7-5 on page 289 by editing the properties of the VIOS partition and selecting the **Mover Service Partition** box.

Note: Mover service partitions are not used during inactive migration partition.


Partition Properties - VIOS_Alpha

Name: * VIOS_Alpha

ID: 4

Environment: Virtual I/O Server

State: Not Activated

Attention LED: Off

Resource configuration: Not configured

OS version: AIX 6.1 6100-03-00-0000

Current profile: VIOS_dummy

System: 8204-E8A*060E9B2

[✓] Mover service partition

Figure 7-5 VIOS Mover Service Partition capability for LPM

Note: If the mover service partition is disabled on either the source or destination VIOS, the mobile partition can participate only in inactive partition mobility.

Tip: If you have multiple network interfaces configured on an MSP, you can chose, through the HMC command line, which IP address the mover uses to transport the mobile partition's data. For this to work, you need VIOS level 2.1.2.10-FP22 or later and HMC level 7.3.5 or later.

CPU and memory requirements

VIOS requires modest additional CPU and memory. Assuming that CPU cycles required to host VIOS are available, there is modest performance impact with VIOS from the client partition perspective. Check the IO latency impact to your application from a host perspective (DB2 server).

During the migration of a LPAR, each VIOS of both the source and the destination system need extra CPU to manage the migration. Uncapping the VIOSs assures the necessary additional CPU to be given to each of them.

The following list details considerations regarding VIOS CPU and memory requirements:

- Allocating dedicated CPU to a VIOS partition results in faster scheduling of I/O operations and improved Virtual I/O performance.
- When the overall pool use is high, and using uncapped shared micropartition for VIOS, allocating less than 1.0 Entitled CPU (ECPU) to VIOS partition might result in longer latency for IO.
- Memory required for a VIOS partition is insignificant. Allocating more memory does not result in any performance improvement.

Tip: To size the VIOS CPU appropriately, take the following factors into consideration:

- LPM requires additional CPU during LPAR mobility.
- Shared uncapped CPU gives you the flexibility to acquire additional CPU units whenever needed.

For more information about VIOS CPU and memory requirements, refer to 6.4, “VIOS sizing” on page 257.

Destination server requisites

A destination system might not contain the existing LPAR name of the LPAR on the source system.

The destination system needs to have sufficient resources to host the migrating partition, not only those such as CPU processing units, and memory, but also virtual slots. Virtual slots are needed to create the required virtual SCSI adapter after the mobile partition has moved to the destination system. You might check for the maximum number of virtual adapters and the actual number of configured virtual adapters.

Recommendations:

- It is suggested to have symmetric configuration between source and destination systems. (see the Tip in “Dual VIOS” on page 292).

- It is suggested that the destination system not be running on a battery. If it does, the situation needs to be corrected before the start of the migration.

VASI interface

The Virtual Asynchronous Service Interface (VASI) provides communication between both mover service partitions and the Power Hypervisor to gain access to the partition state. This VASI interface is automatically created when the VIOS is installed and the VIOS is declared as a mover service partition.

Tips:

- To list partition migration information, use the `lslparmigr` command.
- VASI interface must be available and is mandatory for active LPM. It is not required for inactive partition mobility.

Virtual SCSI interface

Virtual SCSI interfaces need to be defined as Desired (not required) in the VIOS profile to allow the migration process to remove those interfaces during a migration.

When configuring the virtual SCSI interface, it is suggested to create a virtual link between the VIOS and the logical partition. This can be done by selecting **Only selected client partition can connect** in the VIOS profile. Moreover, the VIOSs on both the source and the destination systems must be capable of providing virtual access to all storage devices that the mobile partition is using.

Integrated virtual Ethernet IVE

If you plan to use IVE on your VIOS, ensure it has been configured in promiscuous mode. This mode guarantees that only your particular partition can use the complete port of the associated port group of your IVE card.

For more information about configuring IVE, refer to *Integrated Virtual Ethernet Adapter Technical Overview and Introduction*, REDP-4340.

Virtual Ethernet

Virtual Ethernet adapters are created in the VIOS’ profile to ensure the communication between the VIOSs and the logical partitions configured on a same system.

It is mandatory to create shared Ethernet adapters (SEA) on each the source and the destination server to bridge to the same Ethernet network used by the mobile partition. For more information about SEA settings, refer to “SEA configuration” on page 303.

Virtual slots

Make sure sufficient virtual slots are available on the target system.

> **Note:** Any user defined virtual slot must have an ID higher than 11.

Time and date synchronization

Time and date between VIOSs need to be synchronized. An attribute, the Time Reference, is available in the settings tab of the properties of the partition. Any VIOS partition can be designated as a Time Reference Partition (TRP). Values for the Time Reference is enable or disable (default). Changes take effect as soon as you click **OK**.

Time-of-day synchronization is optional for both active and inactive partition migration, but it is a suggested step in the active partition migration. If you chose not to synchronize using the TRP attribute, the source and target systems synchronizes the clocks during the migration process from the source server to the destination server.

> **Notes:**
>
> - The TRP capability is only supported on systems that are capable of active partition migration.
> - Other TRP are supported by servers. The longest running TRP is recognized as the TRP of the system.
> - Ensure that the HMC and all the partitions have the same date and time to avoid discordances when migration to another server.

Dual VIOS

For more information about dual VIOS setups, refer to 6.3.1, “Dual VIOS” on page 251.

> **Tip:** Non-symmetric configurations are possible when using partition mobility, even though this is not a recommended situation.
>
> If you are in such a situation, only VIOS version 2.1.2.10-FP22 or later is allowed as operating system and HMCs need to be at level 7.3.5.0 or later for LPM to function properly.

Disk mapping to client partition

Any client partition willing to act in a live partition needs to have its mapped disk of a whole LUN. Logical volumes mapping is not supported with LPM.

In other words, the disks zoned on a VIOS (these are LUNs on the storage box and seen as hdisk on the VIOS) need to be mapped entirely to the logical partition that needs to be migrated from one source system to another.

Moreover, these LUNs must be accessible to VIOSs on both the source and the destination server. Other parameters at disk level and HBA level need to be adapted. For more information about those parameters, refer to 3.4, "Tuning storage on AIX" on page 115.

Tips:

- For all disks in a mobile logical partition, ensure the following settings for the HBAs on all of your VIOSs:
  - reserve_policy: no_reserve policy
  - hcheck_interval: 20
  - fc_err_recov: fast_fail
  - dyntrk: yes
- For all HBAs in VIOSs ensure the following settings:
  - fc_err_recov: fast_fail
  - dyntrk: yes.

7.4.5 LPAR requirements

Table 7-5 lists the LPAR requirements for active migration.

Table 7-5 LPAR requirements for active migration

LPAR tasks	Actions to undertake and remarks	For details see
LPAR profile	Unique name	"LPAR name" on page 296
Operating system	▶ AIX 5300-07-01 ▶ AIX 6100-00-01 ▶ RedHat RHEL1[a] or later + kernel security update ▶ Sles[b] 10 or later + kernel security update	"Operating system requirements" on page 296
Processor setting	▶ Dedicated / Shared ▶ Capped ▶ Uncapped	"Processor settings" on page 298
RMC connections	For active migrations: active RMC connection, rsct daemons running	"RMC connections" on page 298
Physical I/O	Only virtual adapters	"Physical I/O" on page 299
MAC address	Unique across both systems	"Network" on page 300
IVE (HEA) adapter	No IVE / HEA adapters	"Integrated Virtual Ethernet (IVE)" on page 300p
Partition workload group	Might not be part of partition workload group	"Partition workload group" on page 300
Virtual serial adapter	Only serial IDs 0 & 1	"Virtual Serial adapter" on page 301
Barrier Synchronization Register (BSR)	Number of arrays = 0	"Barrier Synchronization Register (BSR)" on page 301
Huge pages	Requested huge pages = 0	"Huge pages" on page 301
Redundant Error Path	Disabled	"Redundant Error Path Reporting" on page 303

a. RHEL: RedHat Enterprise Linux
b. SLES: Suze Linux Enterprise Server

Table 7-6 lists the LPAR requirements for inactive migration.

Table 7-6 LPAR requirements for Inactive migration

LPAR tasks	Actions to undertake and remarks	For details see
LPAR profile	Unique name	"LPAR name" on page 296
Operating system	► Prior AIX 5L if OS supports POWER6 and virtual devices ► 5200-10 or later (only for 9117 MMA systems) ► 5300-06 or later ► 6100 or later ► RedHat RHEL1 or later + kernel security update ► Sles 10 or later + kernel security update	"Operating system requirements" on page 296
RMC connections	Not needed	"RMC connections" on page 298
Physical I/O	Can have dedicated I/O	"Physical I/O" on page 299
MAC address	Unique across both systems	"Network" on page 300
IVE (HEA) adapter	Can be used	"Integrated Virtual Ethernet (IVE)" on page 300
Partition workload group	Might be part of partition workload group	"Partition workload group" on page 300
Virtual serial adapter	Only serial ID's 0 & 1	"Virtual Serial adapter" on page 301
Barrier Synchronization Register (BSR)	Can be used	"Barrier Synchronization Register (BSR)" on page 301
Huge pages	Can be used	"Huge pages" on page 301
Redundant Error Path	Can be used	"Redundant Error Path Reporting" on page 303

LPAR name

Partition mobility is only possible for those partition profiles configured to run AIX or Linux operating systems. Even though VIOSs are configured as a Mover Service Partition (MSP), they cannot be migrated.

Note: The target server might not contain a LPAR with the same name as the partition you want to move.

For the migration to be successful, the LPAR name from the migrating partition must not exist on the destination system. You can, however, determine a new name for your partition. The HMC creates a new profile containing the partition's current state, the configuration, and any changes that are made during the migration.

Note: For a logical partition to participate in an active partition migration, it cannot have any physical I/O. All I/O must be virtual. If the mobile partition has required or physical I/O, it can participate in inactive partition migration.

Tip: Devices configures as Desired can be dynamically removed from a partition's profile. Others marked as Required require you to change the profile and shutdown. Reboot your partition for the changes to take effect.

Operating system requirements

Operating system requirements for logical partitions differ depending on the type of migration you want to perform: active or Inactive migration (see Table 7-7 on page 297 and Table 7-8 on page 297).

Notes:

- Active partition migration

  This is the ability to move a running logical partition with its operating system and application from one system to another without interrupting the service / operation of that logical partition.

- Inactive partition migration

  This is the ability to move a powered off logical partition with its operating system and application from one system to another.

Table 7-7 lists the operating system level needed for active partition mobility.

Table 7-7 Operating system level needed for active partition mobility

OS level	Active migration
AIX 5L	Version 5.3 Technology Level 7 or later
AIX 6	Version 6.1 or later
RedHat Enterprise Edition	Version 5 (RHEL5) Update 1 or later Kernel security update
RedHat Enterprise Edition	Version 10 (SLES 10) Service Pack 1 or later Kernel security update

Table 7-8 lists the operating system level needed for inactive partition mobility.

Table 7-8 Operating system level needed for inactive partition mobility

OS level	Inactive migration
Prior AIX 5L	Any version, as long as supported on POWER6, supports virtual devices
AIX 5L	Version 5.2 Technology Level 10 or later (Only on 9117 MMA Systems) Version 5.3 Technology Level 6 or later
AIX 6	Version 6.1 or later
RedHat Enterprise Edition	Version 5 (RHEL5) Update 1 Kernel security update
Suse Enterprise Edition	Version 10 (SLES 10) Service Pack 1 Kernel security update

Notes:

- For earlier versions of the operating systems, inactive partition mobility is still possible. Your operating system, AIX or Linux, needs to support virtual devices.
- Inactive partition migration is less restrictive in terms of pre-requisites than the active partition migration. The following list outlines the elements that you can configure for inactive partition mobility:
  - Dedicated I/O (I/O is removed from the partition before the migration occurs.)
  - Barrier Synchronization Register (BSR)
  - Huge pages
  - Some dynamic changes (for example, partition workload groups, MSP, time-reference and VASI)
  - Static changes (BSR and redundant error path reporting)
  - No need to have a MSP, neither on source nor on target system
  - No need to have a VASI adapter on the VIOSs
  - No need to synchronize the time-of-day clocks
  - No need to have RMC connection

Processor settings

Configuring processors is of high importance, be it on VIOSs or logical partitions.

For optimal balanced performance with an uncapped shared processor LPAR, we suggest setting a maximum two virtual processors per physical processor: 1 physical CPU ==> max 2VP.

For capped shared processor LPAR, set the number of virtual processor equal to a round-up of the entitled capacity.

Note: For a more detailed information about processor settings, refer to Chapter 5, “LPAR considerations” on page 225.

RMC connections

Resource monitoring and control (RMC) is a feature that can be configured to monitor resources such as disk space, CPU usage, and processor status. It allows performing an action in response to a defined condition. It is actually technically a subset function of the Reliable Scalable Cluster Technology (RSCT). For more information about RMC, refer to IBM Redbooks publication *A Practical Guide for Resource Monitoring and Control (RMC)*, SG24-6615.

Prior to an active migration of your logical partition, ensure that RMC connections are established between:

- VIOSs of both systems intervening in the mobility.
- Mobile partition and the both the source and destination systems.

Tip: Use the `lsrsrc IBM.ManagementServer` command to verify the RMC connections are active and check for ManagerType=HMC. Alternatively, you can use the `lspartition -all` command on the HMC and check against your partition for Active:<3>.

Tips:

- To re-synchronize the RMC connection run the `/usr/sbin/rsct/bin/refrsrc IBM.ManagementServer` command. Running this command suppresses the wait time after synchronization.
- Your LPAR is not listed with the `lspartition` command if it is not in running state (it is not powered on).
- RMC needs about five minutes to synchronize after network changes or partition activation.

Physical I/O

When going for LPM, no physical or required adapters can be configured in the mobile partition's profile. All I/O must be virtual. However, if the mobile partition has physical or dedicated adapters, it can participate in an inactive partition migration. Physical adapters marked as `Desired` can be removed dynamically with a dynamic LPAR operation. For more information about using DLPAR, refer to *PowerVM Virtualization on IBM System p: Introduction and Configuration Fourth Edition*, SG24-7940.

Tips:

- You can chose to use physical adapter in your mobile partition. If you do so and want an active partition mobility, move your physical adapters to virtual adapters prior the migration can occur. Downtime might be necessary to perform these tasks.
- For your particular LPAR running your DB2 database, it is suggested not to share the bandwidth of your I/O card with other partitions to avoid unnecessary overhead on the VIOS. Evaluate the need to share it, keeping in mind performance.

Network

The logical partition's MAC address need to be unique across both systems. You might consider using the **`netstat`** or **`entstat`** command to check this.

Moreover, the mobile partition's network must be virtualized using one or more VIOSs. Logical Host Ethernet Adapters (LHEA) are not supported for a partition to be active in a migration process. see "Integrated Virtual Ethernet (IVE)" on page 300 for more information about Integrated Virtual Ethernet.

Notes:

- Systems configured with IVM have only one VIOS.
- For systems configured with HMC, it is suggested to configure two VIOSs.

Integrated Virtual Ethernet (IVE)

An integrated virtual Ethernet adapter (IVE), also known as a Host Ethernet adapter, is available in a two or four port integrated Ethernet card and is directly attached to the Power Hypervisor. The power Hypervisor can create up to 32 logical Ethernet ports that can be given to the logical partition. These ports are seen as LHEA and provide the logical partitions with a virtual Ethernet communication link without the necessity to use a VIOS to virtualize an Ethernet card given to it.

If you plan to use the LPM capabilities of your Power System, you cannot use a logical host, as LHEA adapters are considered as physical adapters.

Partition workload group

For a mobile partition to migrate from one source system to a destination system, the mobile partition must not be part of a partition workload groups.

Notes:

- A partition workload group is a group of logical partitions whose resources are managed collectively by a workload management application. A partition workload group identifies a set of partitions that reside on the same system.
- Workload management applications can balance memory and processor resources within groups of logical partitions without intervention from the HMC (Hardware Management Console). Workload management applications are installed separately and can be obtained from a solution provider company.

For more information about partition workload group and workload management applications, refer to Chapter 5, “LPAR considerations” on page 225.

Virtual Serial adapter

Virtual serial adapters are often used for virtual terminal connections to the operating system. The two first serial adapters (adapter ID’s 0 and 1) are reserved for the HMC.

Validate that no physical adapters are in the mobile partition and that no virtual serial adapters are in virtual slots higher than 1. In other words, the only exception for virtual serial adapter is for the virtual terminal connection.

Barrier Synchronization Register (BSR)

Barrier synchronization registers provide a fast, lightweight barrier synchronization between CPUs. This facility is intended for use by application programs that are structured in a single instruction, multiple data (SIMD) manner. Such programs often proceed in phases where all tasks synchronize processing at the end of each phase. The BSR is designed to accomplish this efficiently. Barrier synchronization registers cannot be migrated or re-configured dynamically. Barrier synchronization registers cannot be used with migrating partitions.

Notes:

- BSR can be used in inactive partition migration.
- Disabling BSR cannot been changed dynamically. Modify the partition’s profile and shutdown (not reboot) the partition for changes to take effect.
- If your migrating partition contains dedicated physical resources, it is mandatory to move those physical resources to virtual resources prior to migrating or the migration fails.

Huge pages

Huge pages can improve performance in specific environments that require a high degree of parallelism. The minimum, desired, and maximum number of huge pages can be specified to assign to a partition when you create a partition profile.

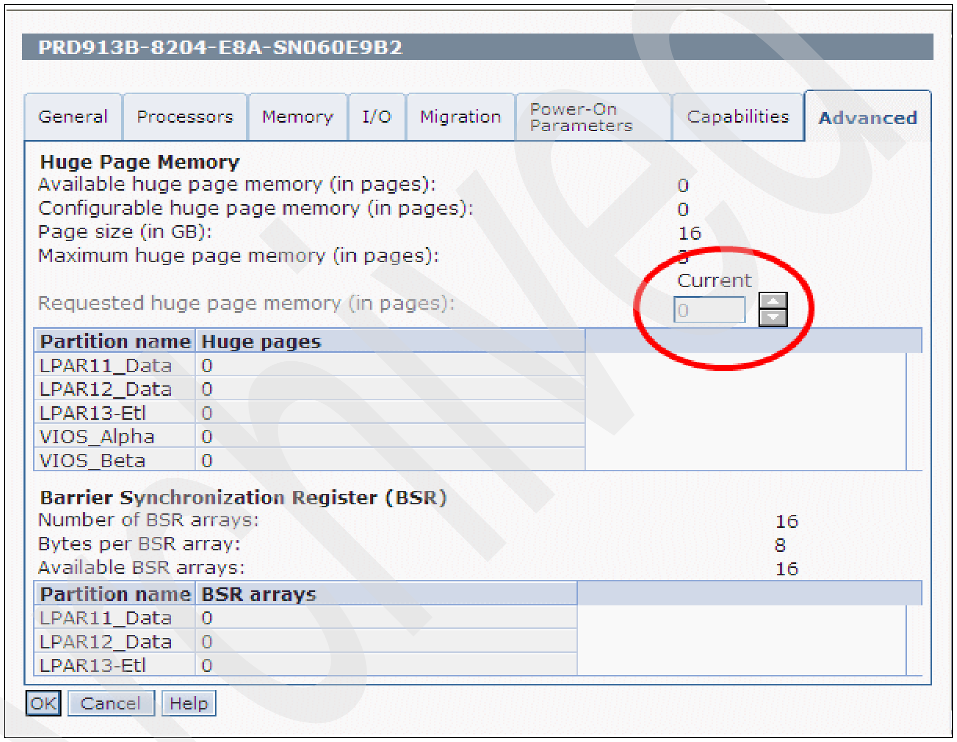
However, for a logical partition to participate in active partition migration, it cannot use huge pages. For more information about Huge pages, refer to 2.1.4, “Large page considerations” on page 46.

Note: If your mobile partition does use huge pages, it can participate in an inactive partition migration.

Huge page settings can be checked on the HCM console as follows:

1. Select your system in the navigation area.
2. Choose **Properties**.
3. Select the Advanced tab
4. Verify that the Current Requested Huge Page Memory field indicates 0 (zero). See Figure 7-6.

   If this field is not equal to zero, only an inactive partition mobility works.



Huge Page Memory

Available huge page memory (in pages):	0
Configurable huge page memory (in pages):	0
Page size (in GB):	16
Maximum huge page memory (in pages):	3
	Current
Requested huge page memory (in pages):	0

Partition name	Huge pages
LPAR11_Data	0
LPAR12_Data	0
LPAR13-Etl	0
VIOS_Alpha	0
VIOS_Beta	0

Barrier Synchronization Register (BSR)

Number of BSR arrays:	16
Bytes per BSR array:	8
Available BSR arrays:	16

Partition name	BSR arrays
LPAR11_Data	0
LPAR12_Data	0
LPAR13-Etl	0

Figure 7-6 Huge page settings

Note: If you need to use huge pages and active partition mobility, you need to set to 0 all fields relative to huge page memory of your mobile partition for the migration to be successful. As this is not a dynamic operation, you need to modify the partition's profile and shutdown (not reboot) the partition for the changes to take effect.

Redundant Error Path Reporting

This indicates whether the logical partition is set to report server common hardware errors to the HMC. The service processor is the primary path for reporting server common hardware errors to the HMC. Selecting this option allows you to set up redundant error reporting paths in addition to the error reporting path provided by the service processor. You can change this setting by activating the logical partition using a partition profile set to enable redundant error path reporting. For mobile partitions, this profile attribute must be de-activated.

7.4.6 Network requirements

Table 7-9 shows the network requirements.

Table 7-9 Network requirements

Tasks	Actions to undertake Remarks	For details see
SEA configuration	► For active partition mobility: SEA failover mechanism ► For inactive partition mobility: SEA	"SEA configuration" on page 303
Network access	Communication between both systems necessary	"Access to the network" on page 305
IVE/LHEA adapter	► For active partition migration: not supported ► For inactive partition mobility: supported	"LHEA adapter" on page 305

The network is important to consider in an environment where you want to use the migration capabilities of your servers.

SEA configuration

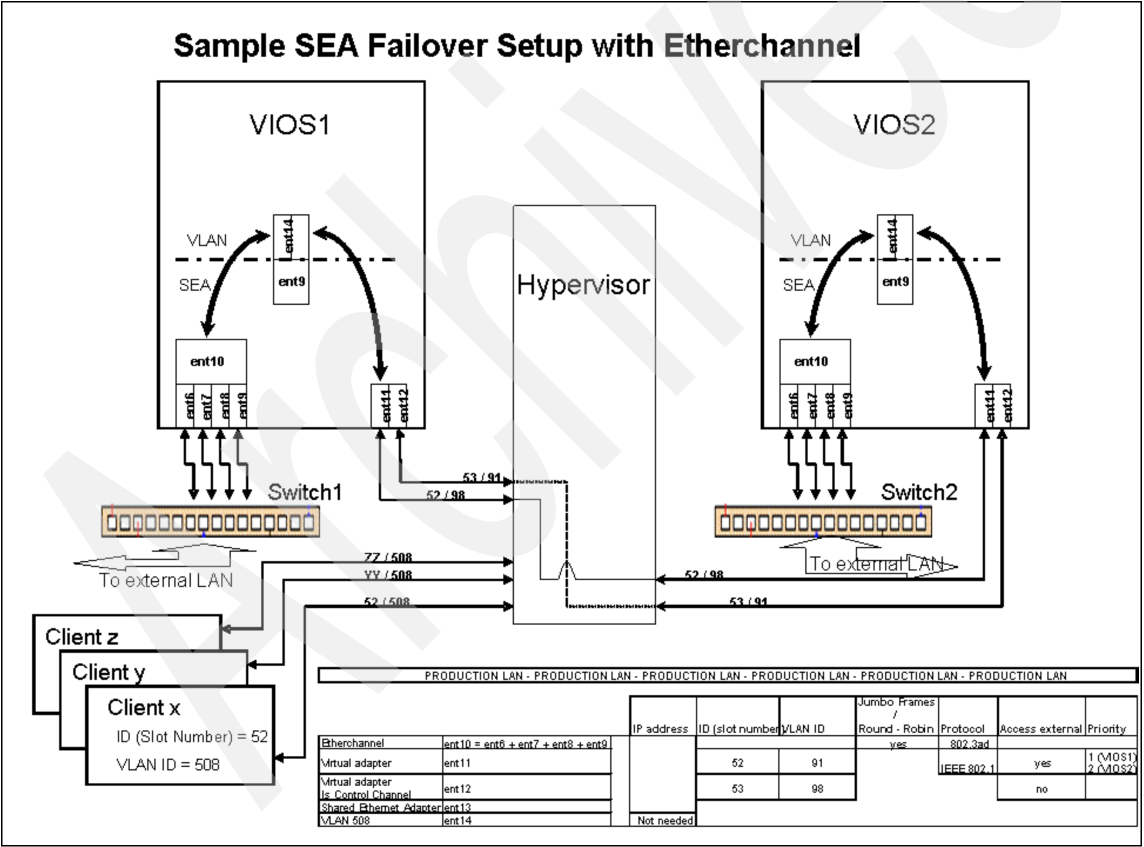
You need to complete several tasks to ensure your network is ready for the migration.

You have to create a shared Ethernet adapter (SEA) on both of your source and destination system VIOSs. The SEA bridges your external network to your internal virtual network (Layer-2 bridge). It provides the ability to your client partitions to share one physical Ethernet adapter. When using dual VIOS setup in an HMC environment, the network can be made highly available by creating the SEA failover mechanism, as shown in Figure 7-7 on page 304.

Notes:

- SEA can only be hosted on VIOSs.
- VIOS running on IVM cannot implement the SEA failover mechanism because it can only contain one single VIOS. For more information about SEA failover mechanism, refer to IBM Redpaper *IBM System p Advanced POWER Virtualization (PowerVM) Best Practices*, REDP-4194.

SEA can be built of physical interfaces or IVE cards. If you want to use the IVE card, ensure that the IVE port you are using is set to promiscuous mode. This mode ensures that the complete port is dedicated to your VIOS and that no other partition is able to use it. For more information about IVE, refer to *Integrated Virtual Ethernet Adapter Technical Overview and Introduction*, REDP-4340.



		IP address	ID (slot number)	VLAN ID	Jumbo Frames / Round - Robin	Protocol	Access external	Priority
Etherchannel	ent10 = ent6 + ent7 + ent8 + ent9				yes	802.3ad		
Virtual adapter	ent11		52	91		IEEE 802.1	yes	1 (VIOS1) 2 (VIOS2)
Virtual adapter Is Control Channel	ent12		53	98			no	
Shared Ethernet Adapter	ent13							
VLAN 508	ent14	Not needed						

Figure 7-7 Sample SEA failover setup with Etherchannel and dual VIOS setup

Recommendation: It is suggested to configure your IP address of your VIOS on an additional virtual adapter that you need to create, apart from the SEA failover mechanism.

Access to the network

The source and target systems need to communicate with each other. You must give access to both systems to the same network segment and VLAN. Moreover, partitions and HMC also need to exist on the same network.

LHEA adapter

When configuring your mobile logical partition, make sure that no LHEA device is configured. LHEA devices are considered as physical devices. Active partition mobility requires no physical adapter to migrate successfully. However, inactive partition mobility can contain LHEA devices if you want to use these.

For more information about Integrated Virtual Ethernet adapters (IVE / HEA) refer to *WebSphere Application Server V6.1: JMS Problem Determination*, REDP-4330.

7.4.7 Storage requirements

Table 7-10 lists the storage requirements.

Table 7-10 Storage requirements

Tasks	Actions to undertake Remarks	For details see
LUN creation on SAN storage box	► Create LUN in particular RAID level ► Divide LUN into desired capacity ► Map to host group	"LUN creation on SAN storage box" on page 310
Disks and HBA parameters	Disks parameters: ► algorithm=round_robin ► hcheck_interval=20 ► max_transfer= ► reserve_policy=no_reserve ► queue_depth=(depends on the storage vendor and type) HBA parameters: ► dyntrk=yes ► fc_err_recov=fast_fail ► max_xfer_size= ► lg_term_dma= ► num_cmd_elems=	"Disk and HBA parameters" on page 310
Disk mapping versus LUN masking	Only disk mapping supported	"Disk mapping versus LUN masking on VIOS" on page 314

Note: Some of the recommended parameters depend upon the storage subsystem you are using. For more information about those specific parameters, refer to 3.4, "Tuning storage on AIX" on page 115.

LPM functions under a specific set of rules. As we have already discussed, it is licensed under PowerVM APV Enterprise Edition. All storage and networking resources must be virtual, as no physical device can be moved. So, any supported virtual storage device (SAN, iSCSI, and so forth) can be configured to provide storage space to your mobile logical partition. The virtual storage device cannot be a logical volume contained within a volume group assigned to the VIOS. To enable multiple system access, the reserve policy must be set to allow concurrent access. This is accomplished through the no_reserve policy attribute. This needs to be performed before the vtscsi device is created, using the following command:.

```
$ chdev - dev hdiskn -attr reserve_policy =no_reserve
```

For our testing we used a DS5300 SAN Storage Solution. We set up transaction logs and data on separate file systems and, ideally, on external storage systems on their own dedicated LUNs. In our example, we set up transaction logs under the `/db2_log_bck` directory and data under `/db2_data`. These file systems were located in the SAN on their own dedicated LUNs. We defined our database paths as shown using the database creation clause. The database was configured as an automatic storage database, with the containers placed on `/db2_data`.

```
CREATE DATABASE TPCE AUTOMATIC STORAGE YES on /db2_data DBPATH on
/home/db2inst1  USING CODESET ISO8859-1 TERRITORY US COLLATE USING
IDENTITY;
```

Using automatic storage significantly reduces time spent on maintenance when there is a need to add storage paths to our database. Automatic storage provides optimal performance, so it is an ideal choice for storage type. When you need more storage space or I/O you need to provide additional LUNs from your SAN system and make them available to your OS. After this, you can add the new storage path to your database environment. This increase in storage paths for your database table spaces increases the amount of storage, and I/O capacity.

The following example shows the command for adding two additional storage paths to our database. Ideally, each storage path is equal in size and lay on its own file system under dedicated LUNs.

```
ALTER DB virtdb ADD STORAGE PATH
```

When the I/O capacity has increased, you might need more CPU capacity to handle the increased I/O. For an uncapped shared processor logical partition, this is not a problem as long as there is enough CPU capacity in the shared processor pool. If the increase in the needed CPU capacity is permanent, consider increasing the entitled processor capacity for the database logical partition. With dynamic logical partitioning, this can be achieved without any effect to service. You must change the value of the desired processor units for the LPAR. For a Power Systems server environment you must choose between the virtualized storage and the locally attached storage. Although the virtualized storage provides easier maintenance, it has a slight overhead. On most systems this overhead is not noticeable, so we are able to take advantage of the benefits for easier maintenance when using virtual storage. VIOS makes it easier to add more storage to your system without a service break. With more than one virtual server, you are always able to add physical adapters and SAN systems to your environment without service breaks on your production environment. This ability to provide additional storage and I/O resources without service downtime combined with the ease of use and maintenance of DB2 automatic storage, makes Power Systems environments ideal for dynamic computing environments with dynamic resource needs.

For more information about the alternate approaches (NFS, iSCSI and FCP) to attach storage to a DB2 server, see *IBM DB2 9 on AIX 5L with NFS, iSCSI, and FCP using IBM System Storage N series*, REDP-4250.

A whitepaper covering the testing that was done using DB2 and LPM using a network attached storage (NAS) over iSCSI can be found at the following Web page.

```
https://www-304.ibm.com/partnerworld/wps/servlet/ContentHandler/whitepa
per/power/lpm/use
```

There is also a demo that has been built from the tests done with DB2 and NAS over iSCSI. It can be found at the following Web page:

```
http://www.ibm.com/partnerworld/wps/servlet/ContentHandler/VPAA-7M59ZR
```

Storage is a crucial point in the setup of an environment that needs to take benefit of the migration capabilities of Power Systems.

The storage that you define on your SAN storage box must be on an external SAN storage box, accessible by both the source and the destination systems, as shown in Figure 7-8.

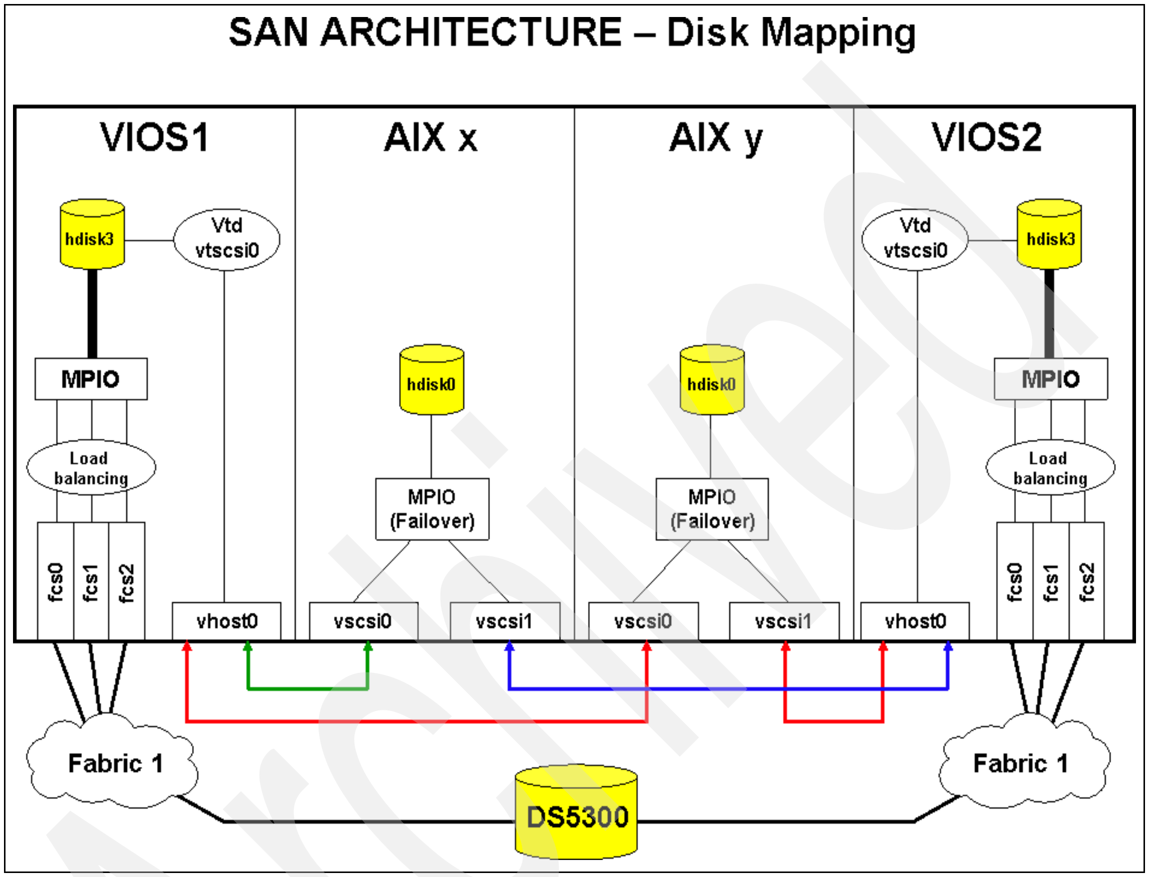


Figure 7-8 SAN Architecture: Example of Disk Mapping

DB2 performance is heavily dependent on the I/O subsystem performance. To attain the best possible I/O throughput, the data layouts of database tables demand special attention from database administrators and system administrators. The chosen I/O type has great impact on the manageability and extensibility of the DB2 storage requirements. Therefore, it is critical to consider workload priorities and to examine trade-offs between disk I/O types. You can choose between locally attached I/O or VIO, or both, within a partition.

All client logical partition requests are passed down to the VIOS, where it performs the actual disk I/O operation and returns data directly to a client partition (no double buffering in the case of Virtual SCSI).

LUN creation on SAN storage box

LUNs need to be defined, following the best practices for storage configuration, using RAID techniques as described in 3.3, “Storage hardware” on page 100.

From the array configuration on the SAN, LUNs from equal size are created. A host group is defined and contains the hosts. Those hosts are referenced by their world wide port name (WWPN). Host groups contain those hosts that share the same disks. When migrating a logical partition from a source to a destination system, all disks defined into that logical partition need to be visible from both the source and the destination system for the migration to be successful. As for the network that needs to be accessible by both systems, the SAN disks must be able to attach to either the source or the destination system. It is one or the other, not both system at the same time. Therefore, a few parameters need to be set not only at disk level, but also at Fibre Channel adapter level, as explained in the next sections.

Disk and HBA parameters

Prior to map disks to logical partition, to change the disk and HBA parameters to allow both the systems to attach the shared disks.

Here are the suggested settings for the disks parameters (See the list that follows these settings for explanations about the parameters):

- algorithm: round_robin
- hcheck_interval: 20
- reserve_policy: no_reserve
- max_transfer: see the Note, “Setting the max-transfer size” on the next page.
- queue_depth: This parameter depends on the storage box used and on the storage box firmware.

More information about these parameters is given in the following list:

- algorithm

  The algorithm attribute defines the methodology that the Path Control Module (PCM) uses to manage I/O across the paths configured for a device.

  (failover: all I/O go through one single path and paths are kept in a list to determine the next path to use, round_robin: all I/O is equally distributed across enabled paths)

- hcheck_interval

  This parameter defines how often the health check is performed on the paths for a device. The attribute supports a range from 0 to 3600 seconds. When a value of 0 is selected, health checking is disabled.

- reserve_policy

  The reserve_policy attribute determines the type of reserve methodology the device driver implements when the device is opened. It can be used to limit device access from other adapters, whether on the same system or another system. (single_path, no_reserve, PR_exclusive, PR_shared)

- max_transfer

  The maximum transfer parameter determines the size, in bytes, of the largest single data transfer request that can be handled by this adapter.

Setting the max-transfer size: If a backing device (such as hdisk or logical volume) is exported through a vhost adapter, and its max_transfer size is greater than the max_transfer size of an already associated backing device of that vhost adapter, the new virtual disk is not presented to the client partition.

You might want to configure the max_transfer size of your newly added disk to the largest max_transfer size of the existing backing devices. To change the max_transfer size run the following command:

```
chdev -dev hdiskx -attr max_transfer=<new size>
```

In this command, `hdiskx` represent the disk whose parameters need to be changed and `<new size>` is the new size in hex format for your max_transfer parameter.

- queue_depth

  The queue_depth parameter represents the number of requests a disk can hold in its queue. The default value for IBM disks is three. For non-IBM disks this value is zero. Changes to this value are immediate and permanent.

Setting the queue_depth parameter:

Here are recommendations for setting the queue_depth parameter:

- For storage systems running DS4000 /DS5000 controller firmware 07.10.xx.xx, use the following formula to determine maximum queue depth:

  ```
  DSxxx queue depth / [(number of hosts) * (number of LUNs per
  hosts)]
  ```

  In this formula, `DSxxx queue depth` is the queue depth of your storage box.

  For DS4800 or DS5000, queue depth is 4096.

- Disk queue_depth is aligned on the queue_depth of the storage box.
- The queue depth of the disk on the VIOS must match the queue depth of the virtual disk on the logical partition.

For more information about setting your storage system, refer to *IBM Midrange System Storage Hardware Guide*, SG24-7676 and to 3.3, "Storage hardware" on page 100.

Fibre Channel devices such as fscsix, need other parameters to be adapted:

The suggested settings for the HBA parameters are as follows (More information about these parameters can be found in the list after the Note below):

- dyntrk=yes
- fc_err_recov=fast_fail
- max_xfer_size= See Note
- lg_term_dma=See Note
- num_cmd_elems=See Note

Note: As these parameters depend upon the storage type you are using, refer to Chapter 3, "Storage layout" on page 85 for the optimum values for the particular storage type you are using.

More information about these parameters is given in the following list:

- dyntrk

  This parameter enables dynamic tracking and enables the FC adapter driver to detect when the Fibre Channel N_Port IDD of a device changes. The FC adapter reroutes traffic destined for that device to the new address while the device are still online.

- fc_err_recov

  AIX supports Fast I/O Failure for Fibre Channel devices after link events in a switched environment. If the Fibre Channel adapter driver detects a link event, such as a lost link between a storage device and a switch, the Fibre Channel adapter driver waits a short period of time, approximately 15 seconds, so that the fabric can stabilize. At that point, if the Fibre Channel adapter driver detects that the device is not on the fabric, it begins failing all I/Os at the adapter driver. Any new I/O or future retries of the failed I/Os are failed immediately by the adapter until the adapter driver detects that the device has rejoined the fabric.

  Fast Failure of I/O is controlled by a new fscsi device attribute, fc_err_recov. The default setting for this attribute is delayed_fail, which is the I/O failure behavior seen in previous versions of AIX.

  Note: When you only have one single Fibre Channel interface, it is recommended to leave the fc_err_recov attribute on delayed_fail.

  For more information about dyntrk and fc_err_recov, refer to the following Web page:

  http://publib.boulder.ibm.com/infocenter/aix/v6r1/index.jsp?topic=/com.ibm.aix.baseadmn/doc/baseadmndita/dm_mpio.htm&resultof=%2522rmpath%2522%2520&searchQuery=rmpath&searchRank=1&pa

- max_xfer_size

  This is an AIX setting that can directly affect throughput performance with large I/O blocksizes. For more information about max_xfer_size refer to *IBM Midrange System Storage Hardware Guide*, SG24-7676.

- lg_term_dma

  This is an AIX setting that can directly affect throughput performance with large I/O blocksizes. For more information about lg_term_dma refer to *IBM Midrange System Storage Hardware Guide*, SG24-7676.

- num_cmd_elems

  This tunable represents the maximum number of requests that can be outstanding on or queued to a disk adapter. Change this tunable to adapt to various memory and system conditions. The default is 200.

Attention:

- There is a relation between the queue_depth tunable on a disk and the num_cmd_elems on a FC card.

  You can tune the num_cmd_elems as follows:

  ```
  num_cmd_elems = (number of paths that form the vpath) * (value of
  queue_depth of the disk)
  ```

  For example, if you have four paths forming the vpath to your storage and your queue depth is 20, num_cmd_elems need to be set at 4 * 20 = 80.
- Fast I/O Failure is useful in situations where multipathing software is used. Setting the fc_err_recov attribute to fast_fail can decrease the I/O fail times because of link loss between the storage device and switch. This supports faster failover to alternate paths.
- In single-path configurations, especially configurations with a single path to a paging device, the delayed_fail default setting is recommended.

For more information about how to set those parameters, refer to Chapter 3, “Storage layout” on page 85.

Disk mapping versus LUN masking on VIOS

It is important to perform disk mapping on the VIOSs to the client logical partition for the migration to be successful. LUN masking is not supported. In other words, the LUN that you create on the SAN storage box, when discovered on the VIOS with the **`cfgdev`** command must be mapped entirely to the mobile logical partition.

Tips:

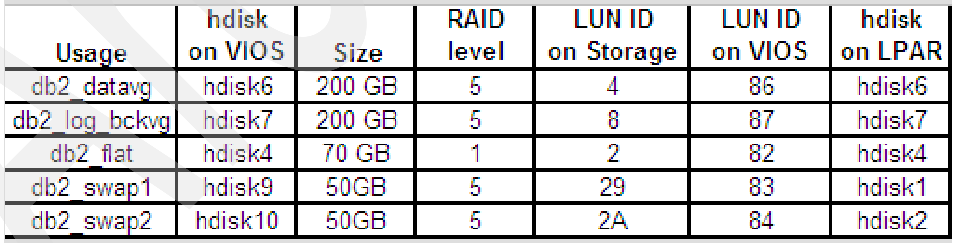
- Run the **`cfgdev`** command on the VIOS as padmin to discover newly added devices that you want to see after you booted the system.
- When using the **`mkvdev`** command to map your disk from the VIOS to the logical partition as shown in the following example, think of adding the -dev parameter that enables you to customize the name of your mapping:

  For example, `mkvdev -vdev hdisk1 -vadapter vhost0 -dev tmp_swap_mobile`
- Both rootvg as data volume groups need to be mapped entirely from the VIOSs to the logical partitions. This is known as disk mapping. In other words, there is no need to define any logical volume on the disk to map it to the client LPAR.

In the example we created a disk mapping on hdisk5, giving it the name `tmp_swap_mobile` to the target device. The Universal Device ID (UDID), the Physical Volume ID (PVID) or the IEEE attributes can be used to verify that your have the same disk on the source and destination system.

Note: The device cannot have an IEEE volume attribute identifier. Use the UDID or PVID of the device to identify uniquely your disk on both VIOSs.

Tip: You can establish a correspondence table to describe the LUNs' IDs on the storage side compared to the LUNs' IDs created with the **`mkvdev`** command on the VIOSs. See the following table for an example:



Usage	hdisk on VIOS	Size	RAID level	LUN ID on Storage	LUN ID on VIOS	hdisk on LPAR
db2_datavg	hdisk6	200 GB	5	4	86	hdisk6
db2_log_bckvg	hdisk7	200 GB	5	8	87	hdisk7
db2_flat	hdisk4	70 GB	1	2	82	hdisk4
db2_swap1	hdisk9	50GB	5	29	83	hdisk1
db2_swap2	hdisk10	50GB	5	2A	84	hdisk2

7.4.8 Summary

Power System virtualization offers a set of resource partitioning and management features such as LPARs, the DLPAR facility, and virtual I/O, under which you can implement SEA, virtual Ethernet, virtual SCSI, or a VLAN. Shared processor partitions allow you to create an LPAR using as little as 0.10 of a processor. The DLPAR facility enables you to change the LPAR resources (processor, memory, and I/O slots) at run time, without rebooting the operating system.

The versatility of the DB2 data server and a variety of possible combinations of Power System virtualization features help DB2 applications to perform optimally in many situations. The Power System virtualization technology can be configured to realize both computer system and business benefits, which includes high performance, workload isolation, resource partitioning, maximum resource use, and high availability at low cost. This technology reduces total cost of ownership (TCO) while also enhancing expandability, scalability, reliability, availability, and serviceability.

The best practices presented in this document are essentially lessons that have already been learned through our own testing. These best practices serve as an excellent starting point for using the DB2 product with Power System virtualization. You can use them to help to avoid common mistakes and to fine-tune your infrastructure to meet your goals for both your business and IT environment. To validate the applicability of these best practices before using them in your production environment, establish a baseline and perform sufficient testing with the various virtualization features.

7.5 Migration process and flow

When you move a logical partition from one system to another, a few of its attributes might change (such as the logical partition ID number) and a few of its attributes remain the same (such as the logical partition configuration). You can choose the way the migration is done by giving a new name to your migrated partition, changing form VIOS mover, and so forth.

Table 7-11 summarizes those major attributes that might change.

Table 7-11 Logical partition attributes that might change after a migration

LPAR attributes that remain the same	LPAR attributes that can be optionally changed by the user
The logical partition type ► dedicated versus shared processor ► dedicated versus shared memory	The logical partition name
The logical partition’s profile	The logical partition ID number

LPAR attributes that remain the same	LPAR attributes that can be optionally changed by the user
Simultaneous Multi-Threading (SMT) state of each processor	
The virtual MAC address, IP addresses	
The disk mappings to the target devices	
The affinity characteristics of the Logical Memory Block (LMB)	

7.5.1 Migration validation

Prior to migrating a logical partition from a source system to a destination system, you want to validate that the migration process is successful. Perform this validation using either the HMC GUI or the command line. Both come up with warnings and errors messages if any relevant problems are detected during the validation process.

The high level procedure of a migration validation, after all requisites for LPM are met, are described in the following steps:

1. Both the source and destination systems are capable of migrating and both are Power6 processor-based, as shown in Figure 7-9.

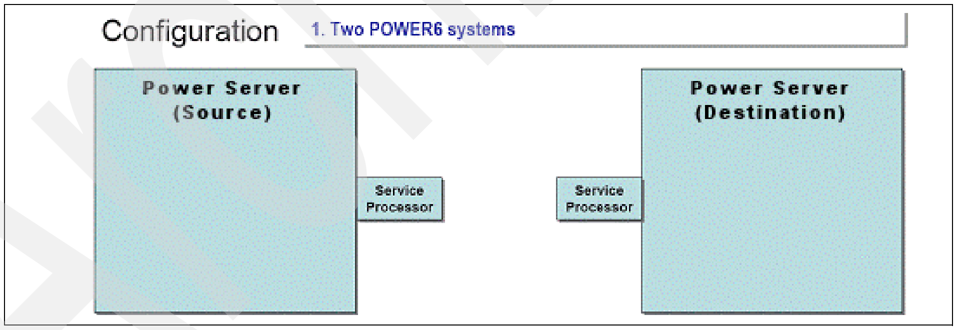


Figure 7-9 POWER6 systems capable to migrate

2. HMC and remote HMC setup as shown in Figure 7-8 on page 309. (See also 7.4.2, “HMC requirements” on page 285).

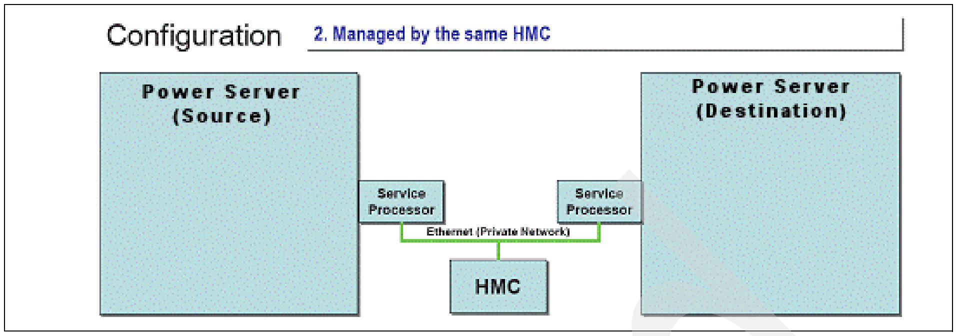

Configuration
2. Managed by the same HMC
Power Server (Source)
Power Server (Destination)
Service Processor
Service Processor
Ethernet (Private Network)
HMC

Figure 7-10 Same HMC management (Remote HMC support under conditions)

3. VIOS configured as shown in Figure 7-11.

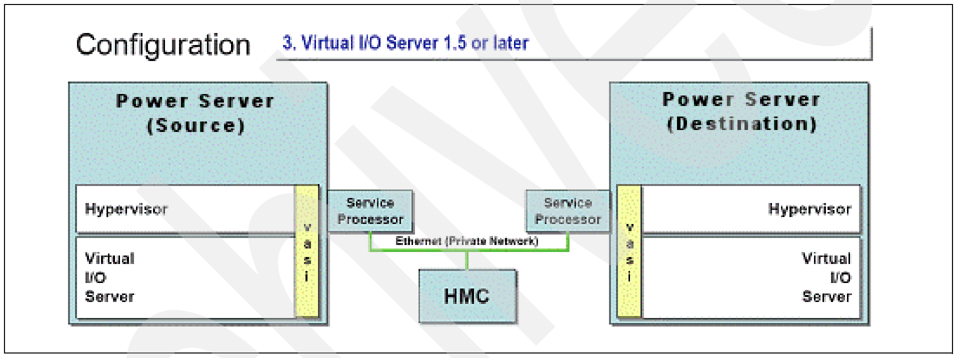


Figure 7-11 VIOS configured

4. LPAR configured for migration.
5. Valid external storage accessible by the logical partition from both the source or destination VIOS.
6. Logical partition accesses the SAN disks through virtual SCSI or Fibre Channel only (or a combination of both). See Figure 7-12 on page 319.

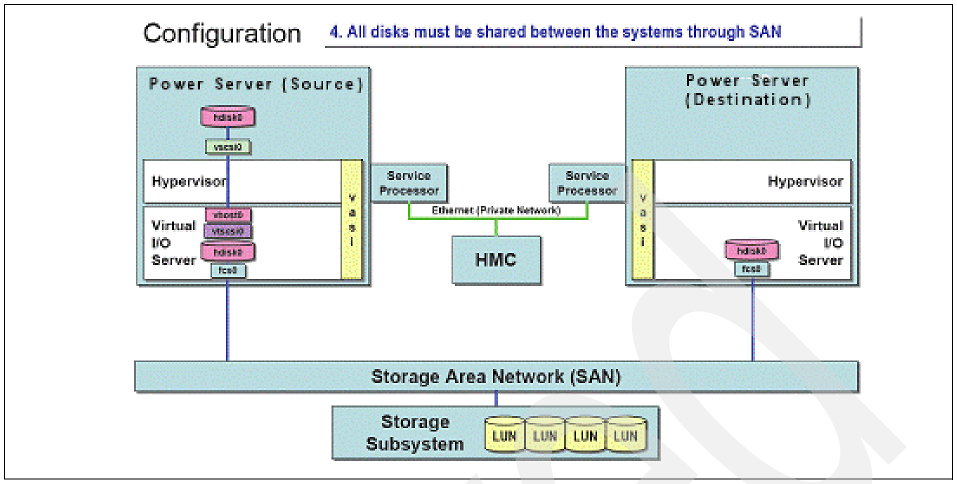


Figure 7-12 All disks must be accessible by both systems

7. Active or inactive migration setup (OS level). See Figure 7-13.

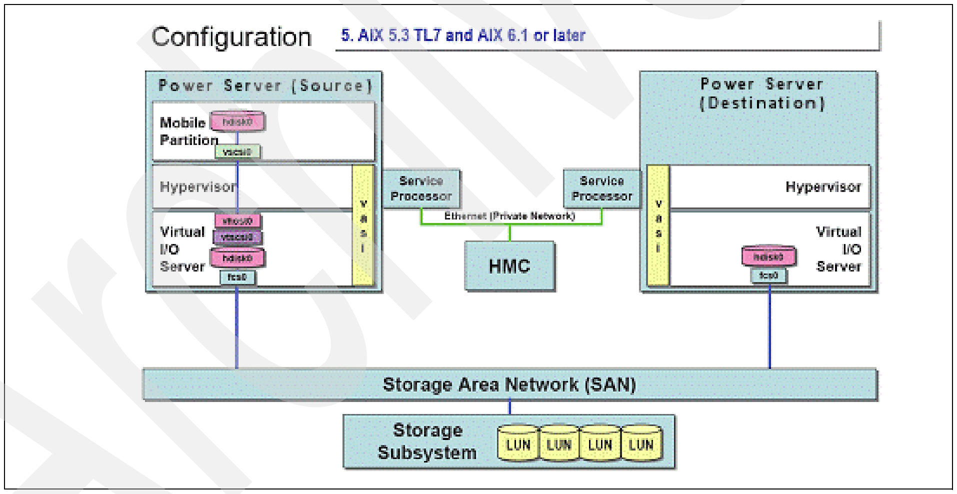


Figure 7-13 Operating System level

8. One or multiple physical IP networks providing communication between the LPAR and both the source and destination systems.
9. The logical partition uses virtual Ethernet adapters, no Integrated Host Ethernet Adapter (IHEA) or RMC connectivity. See Figure 7-14 on page 320.

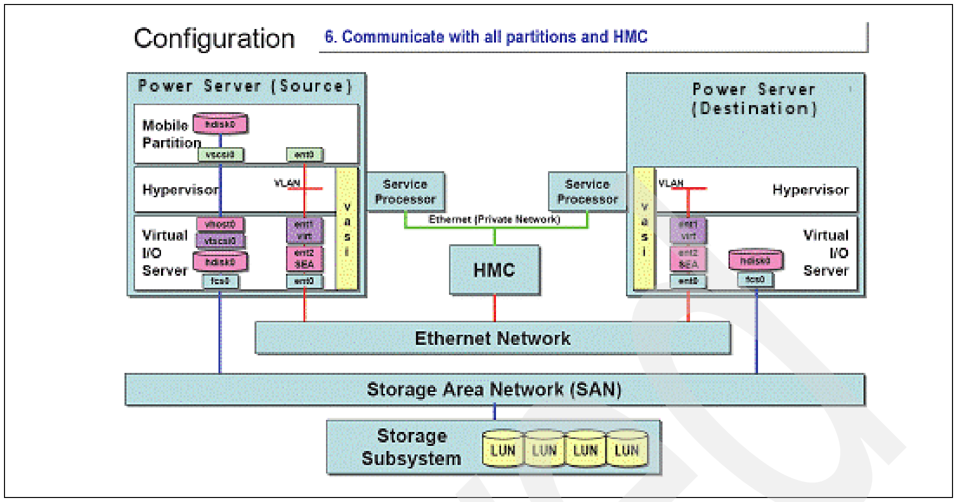


Figure 7-14 Network connectivity with all partitions and HMC

7.5.2 Active partition migration

The migration is successful if the validation ended successfully. After all requirements to allow partition mobility are met and the validation is successful, you are ready to perform the migration.

The overall migration flow is described in the following process:

1. Run the migration process, either through the HMC GUI or through the command line.
2. Determine the destination profile name.
3. Specify the remote HMC (see “Dual VIOS” on page 292 for non-symmetric configurations).
4. Select the destination managed system.
5. Partition validation errors and warnings.
6. Select the MSP.
7. Select the VLAN to be used on the destination system.
8. Select the virtual SCSI adapters to use on the destination system.
9. Select the shared processor pool to be used on the destination system.
10. Select the maximum wait time to migrate (5 minutes is the default).

7.5.3 Inactive partition migration

After all requirements to allow partition mobility are met and the validation is successful, you are ready to perform the migration.

Tip: The major differences in the migration flow between active and inactive partition migration are:

- The MSP attribute is only mandatory for active partition mobility.
- The wait time can be modified when migrating an active partition.

The overall migration flow is described in the following steps:

1. Run the migration process, either through the HMC GUI or through the command line.
2. Determine the destination profile name.
3. Specify the remote HMC (see “Dual VIOS” on page 292 for non-symmetric configurations).
4. Select the destination system.
5. Partition validation errors and warnings.
6. Select the VLAN to be used on the destination system.
7. Select the virtual SCSI adapters to use on the destination system.
8. Select the shared processor pool to be used on the destination system.

To perform the migration, perform the following steps:

1. In the navigation area, select **Systems Management** and select your inactive system.
2. Select your mobile partition.
3. Select **Operations and Mobility**.
4. Click **Migrate**.

Note: During migration, you can monitor the progress of the migration on both systems, using IVM or HMC, depending on your environment.

7.6 Mobility in action

Figure 7-15 shows the setup that was used for DB2 and LPM performance characterization. Throughout the test, the setup revolved around best practices to ensure ease in management and higher performance throughput. As mentioned, preparation for mobility requires careful storage, system and network planning.

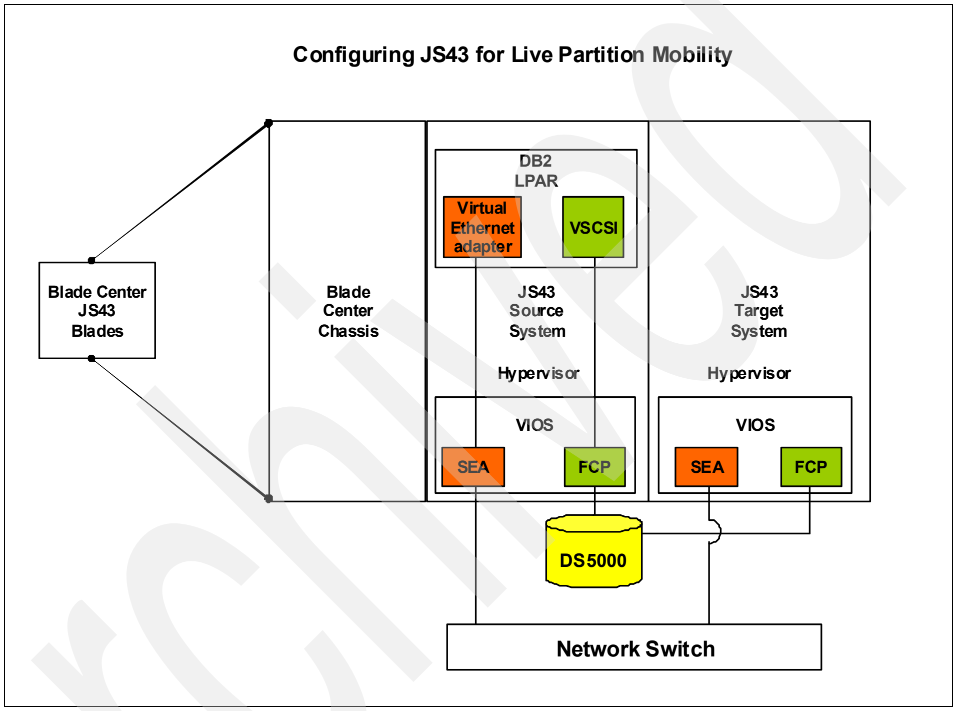
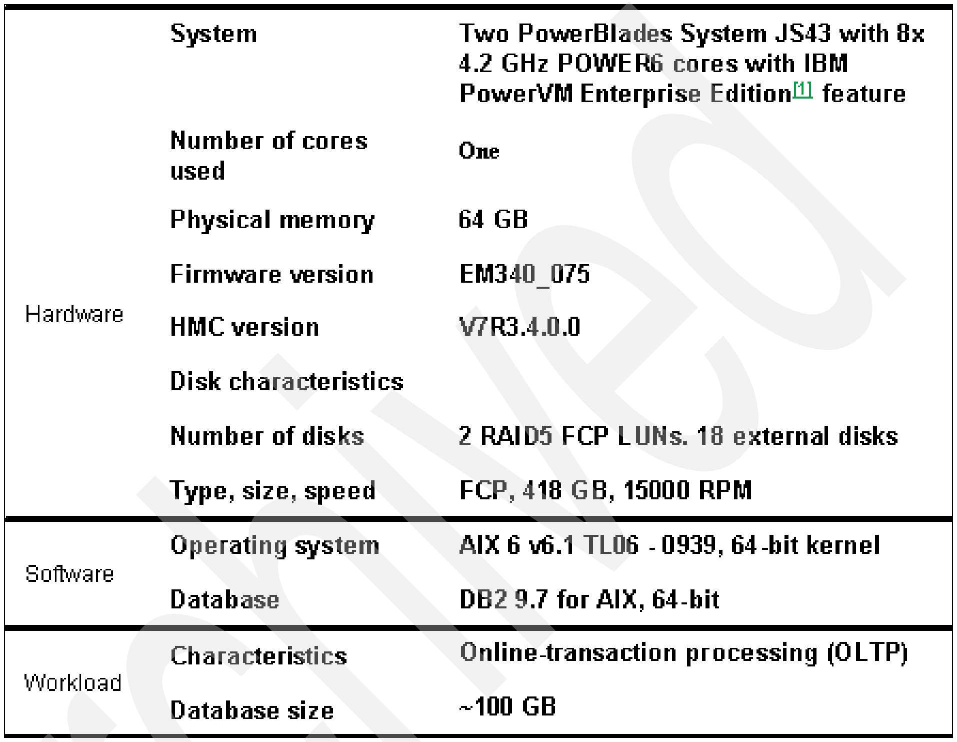


Figure 7-15 Configuring JS43 for LPM

An OLTP workload was set up on DB2 9.7 for the LPM experiment. The objective of this experiment was to demonstrate that DB2 can be moved transparently from one server to another and that client applications connected to the database does not experience any downtime. The OLTP workload is a mixture of read-only and update-intensive transactions that simulate the activities found in complex OLTP application environments. Storage and Virtualization best practices were followed to optimize the throughput of the workload.

The remote DB2 client created 50 connections to the database server, each connection executing a set of transactions. The database server LPAR was moved from one server to another while the OLTP workload was executing.

Figure 7-16 shows the DB2 environment settings used in the LPM experiment.



Hardware	System	Two PowerBlades System JS43 with 8x 4.2 GHz POWER6 cores with IBM PowerVM Enterprise Edition[1] feature
	Number of cores used	One
	Physical memory	64 GB
	Firmware version	EM340_075
	HMC version	V7R3.4.0.0
	Disk characteristics	
	Number of disks	2 RAID5 FCP LUNs. 18 external disks
	Type, size, speed	FCP, 418 GB, 15000 RPM
Software	Operating system	AIX 6 v6.1 TL06 - 0939, 64-bit kernel
	Database	DB2 9.7 for AIX, 64-bit
Workload	Characteristics	Online-transaction processing (OLTP)
	Database size	~100 GB

Figure 7-16 DB2 environment settings

7.6.1 Configuring the test environment

The first step to configuring the environment was to install a VIOS on each JS43. This was accomplished by installing a VIOS mksysb image using NIM. The internal disk within the blade can be used to house the VIOS server or you might choose to boot the blade with the SAN, as this is also supported. We chose the internal disk for the blades.

After the VIOS was installed on each blade, we can connect to the Web-based IVM. This interface provides a HMC-like GUI that allows an administrator to configure LPARs, virtual network, and virtual storage on the blade and VIOS. As this is a Web-based tool, you can point your Web browser at the VIOS host name, and you are presented with the IVM login page. To log in, use the VIOS padmin user ID and password.

Before we can test mobility with the JS43, ensure that the environment is prepared appropriately to support it. Update the firmware levels of the JS43 and associated components such as the Fibre Channel (FC) adapters. Download the latest firmware images for the JS43 and the FC adapters from the JS43 support site. and apply them to each Blade. Install the latest VIOS fixpacks. (Refer to “CPU and memory requirements” on page 290).

With the correct software and firmware levels installed, prepare the Blade, the VIOS, and the LPAR for partition mobility.

What follows is a brief checklist of the tasks performed with the IVM:

1. Enter the PowerVM Enterprise Edition APV key on both Blades. This key is required to enable the mobility feature on the JS43 Blade.
2. Confirm that the memory region size is the same on both Blades. This information can be found under **View/Modify System Properties** in the Memory tab.
3. Configure an SEA on both VIOS. Enable the **Host Ethernet Adapter for Ethernet bridging**. This is required for the virtual Ethernet devices to access the physical Ethernet adapter and the external network. This is performed under the View/Modify Host Ethernet Adapter, Properties tab. Select **Allow virtual Ethernet bridging.** Under View/Modify Virtual Ethernet and the Virtual Ethernet Bridge tab, and select the physical adapter to be used as the SEA. A message displays stating that the operation was successful. The SEA is now configured.
4. Create an LPAR on the source Blade. Select **View/Modify Partition → Create Partition**. Enter the LPAR name, memory, and processor requirements. Ensure that none of the physical HEA ports are selected. Under Virtual Ethernet, select the SEA to use (for instance, ent0). Under Storage Type, select **Assign existing virtual disks and physical volumes**. Select the SAN disk assigned to the VIOS, which in our environment was the DS5300 disks.
5. Click **Finish** to create the LPAR.

   The next step is to install AIX. This can be achieved using a NIM `mksysb` (or rte) install.
6. With the AIX installation and configuration complete, you can configure DB2.

Note: PowerVM Enterprise Edition is mandatory to enable LPM on your Power System.

You are now ready to perform a live partition migration. Do one final review and check. On each VIOS an SEA has been configured. You might use the **`lsmap -all net`** command to confirm the device configuration on both VIOS.

Verify that the same SAN disk can be seen by both VIOS. Using the **lspv** command, check that both VIOS have the same PVID associated with the SAN storage. Confirm that the AIX LPAR is configured with only virtual devices (meaning no physical adapters, another prerequisite for mobility).

Note: To size CPU for VIOS appropriately, take these into consideration:

- LPM requires additional CPU during LPAR mobility.
- Shared uncapped CPU give you the flexibility to acquire additional CPU units whenever needed.

7.6.2 Performing the LPM

In this section we discuss the steps for performing the LPM.

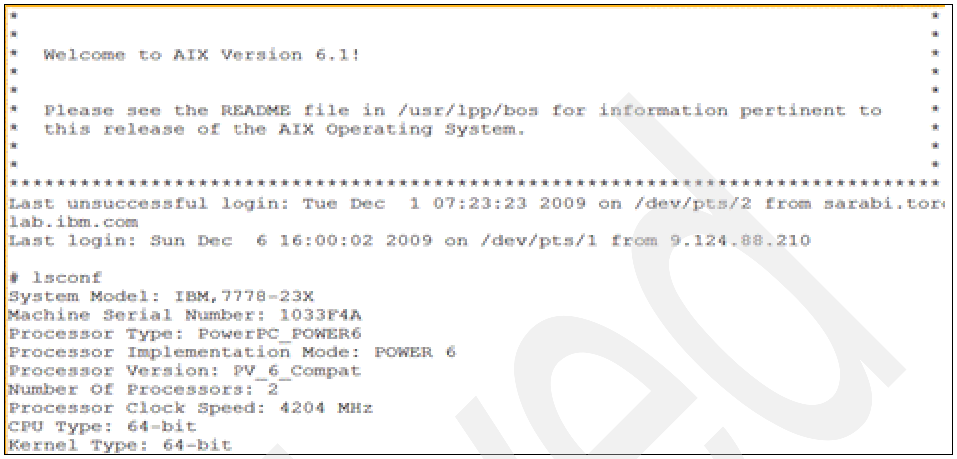
Note: As already mentioned we used PowerBlades during our tests. For more information about Power Systems using HMC, refer to the IBM Redbooks publication *IBM PowerVM Live Partition Mobility*, SG24-7460.

Preliminary test

For the purpose of our tests, a single VIOS has been configured, one per Blade, and one active AIX LPAR running on the first Blade as a VIO client (VIOC). We are now ready to perform a live partition migration. During the migration, the first Blade is known as the source system and the second Blade is the destination system.

The objective is to move the LPAR, from the Blade in the source system to the Blade in the destination system. At the end of the migration, the AIX LPAR is running as a VIOC from the destination system on the other physical Blade. DB2 continues to function throughout the entire migration.

Prior to the migration, run the **lsconf** command from AIX, and note the system serial number (see Figure 7-17).



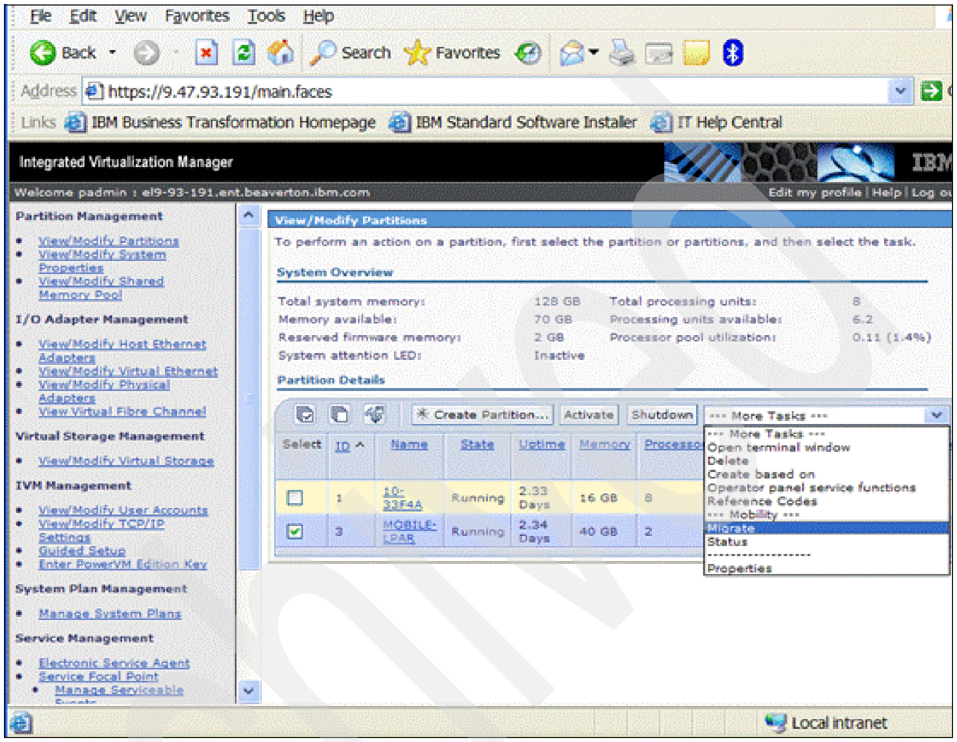
```
*                                                                              *
*                                                                              *
*   Welcome to AIX Version 6.1!                                                *
*                                                                              *
*                                                                              *
*   Please see the README file in /usr/lpp/bos for information pertinent to    *
*   this release of the AIX Operating System.                                  *
*                                                                              *
*                                                                              *
********************************************************************************
Last unsuccessful login: Tue Dec  1 07:23:23 2009 on /dev/pts/2 from sarabi.tor
lab.ibm.com
Last login: Sun Dec  6 16:00:02 2009 on /dev/pts/1 from 9.124.88.210

# lsconf
System Model: IBM,7778-23X
Machine Serial Number: 1033F4A
Processor Type: PowerPC_POWER6
Processor Implementation Mode: POWER 6
Processor Version: PV_6_Compat
Number Of Processors: 2
Processor Clock Speed: 4204 MHz
CPU Type: 64-bit
Kernel Type: 64-bit
```

Figure 7-17 lsconf output prior to the migration

During the migration, DB2 jobs are running on the LPAR. Monitor the system using the **topas** command and observe that DB2 processes are consuming processors during the migration.

In the mean time VIOSs are hand-shaking and transmitting information through the network.

All tasks to perform partition mobility are executed from the IVM, on the source Blade. To start the migration, select the box next to the LPAR and choose **Migrate** from the **More Tasks** drop-down menu. Refer to Figure 7-18.


File Edit View Favorites Tools Help
Back
Search
Favorites
Address https://9.47.93.191/main.faces
Links IBM Business Transformation Homepage IBM Standard Software Installer IT Help Central
Integrated Virtualization Manager
IBM
Welcome padmin : el9-93-191.ent.beaverton.ibm.com
Edit my profile | Help | Log ou
Partition Management
View/Modify Partitions
View/Modify System Properties
View/Modify Shared Memory Pool
I/O Adapter Management
View/Modify Host Ethernet Adapters
View/Modify Virtual Ethernet
View/Modify Physical Adapters
View Virtual Fibre Channel
Virtual Storage Management
View/Modify Virtual Storage
IVM Management
View/Modify User Accounts
View/Modify TCP/IP Settings
Guided Setup
Enter PowerVM Edition Key
System Plan Management
Manage System Plans
Service Management
Electronic Service Agent
Service Focal Point
Manage Serviceable Events
View/Modify Partitions
Create Partition...
Activate
Shutdown
--- More Tasks ---
--- More Tasks ---
Open terminal window
Delete
Create based on
Operator panel service functions
Reference Codes
--- Mobility ---
Migrate
Status

Properties
Local intranet

To perform an action on a partition, first select the partition or partitions, and then select the task.

System Overview

Total system memory:	128 GB	Total processing units:	8
Memory available:	70 GB	Processing units available:	6.2
Reserved firmware memory:	2 GB	Processor pool utilization:	0.11 (1.4%)
System attention LED:	Inactive		

Partition Details

Select	ID ^	Name	State	Uptime	Memory	Processo
☐	1	10-33F4A	Running	2.33 Days	16 GB	8
✓	3	MOBILE-LPAR	Running	2.34 Days	40 GB	2

Figure 7-18 Migration menu

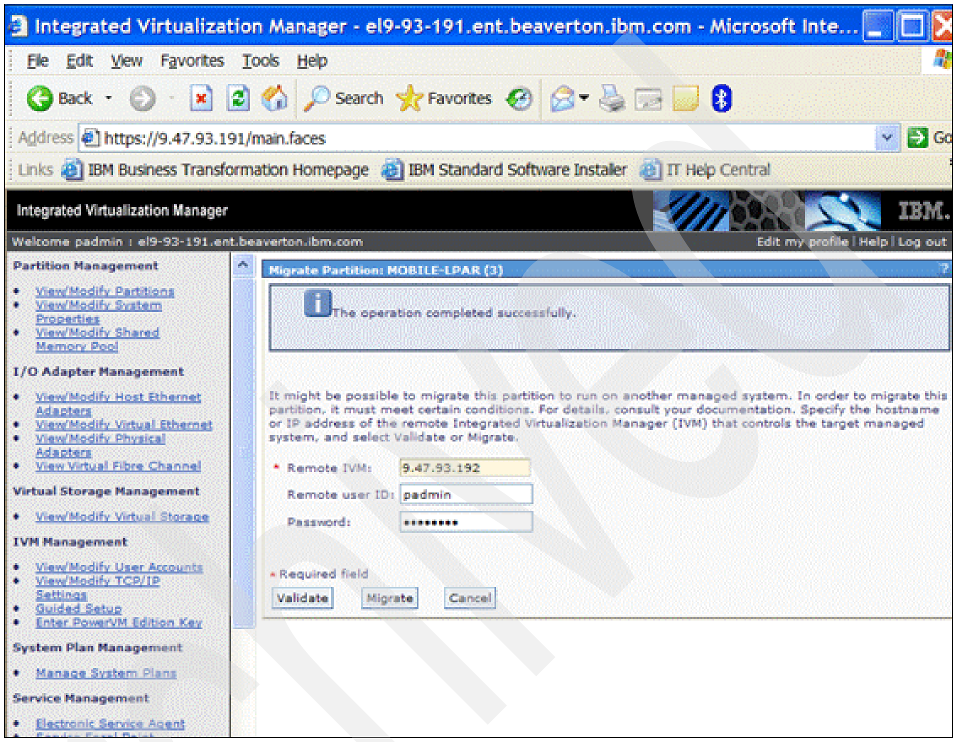
You are presented with a panel to enter the target system details. Enter the details and then click **Validate**. Refer to Figure 7-18.

Migration validation

During the validation phase, several configuration checks are performed. Some of the checks include:

- Ensuring the target system has sufficient memory and processor resources to meet the LPAR's current entitlements.
- Checking there are no dedicated physical adapters assigned to the LPAR.
- Verifying that the LPAR does not have any virtual SCSI disks defined as logical volumes on any VIOS. All virtual SCSI disks must be mapped to whole LUNs on the SAN.
- RMC connections to the LPAR and the source and target VIOS are established.
- The partition state is active, meaning Running.
- The LPAR's name is not already in use on the target system.
- A virtual adapter map is generated that maps the source virtual adapter/devices on to the target VIOS. This map is used during the actual migration.

After the validation completes successfully, a message stating `It might be possible to migrate the this partition ...` appears (Figure 7-19). Click **Migrate** and the migration to the other Blade begins. Monitor the status of the migration by clicking **Refresh**.



Integrated Virtualization Manager - el9-93-191.ent.beaverton.ibm.com - Microsoft Inte...

Address https://9.47.93.191/main.faces

Welcome padmin : el9-93-191.ent.beaverton.ibm.com

Migrate Partition: MOBILE-LPAR (3)

The operation completed successfully.

It might be possible to migrate this partition to run on another managed system. In order to migrate this partition, it must meet certain conditions. For details, consult your documentation. Specify the hostname or IP address of the remote Integrated Virtualization Manager (IVM) that controls the target managed system, and select Validate or Migrate.

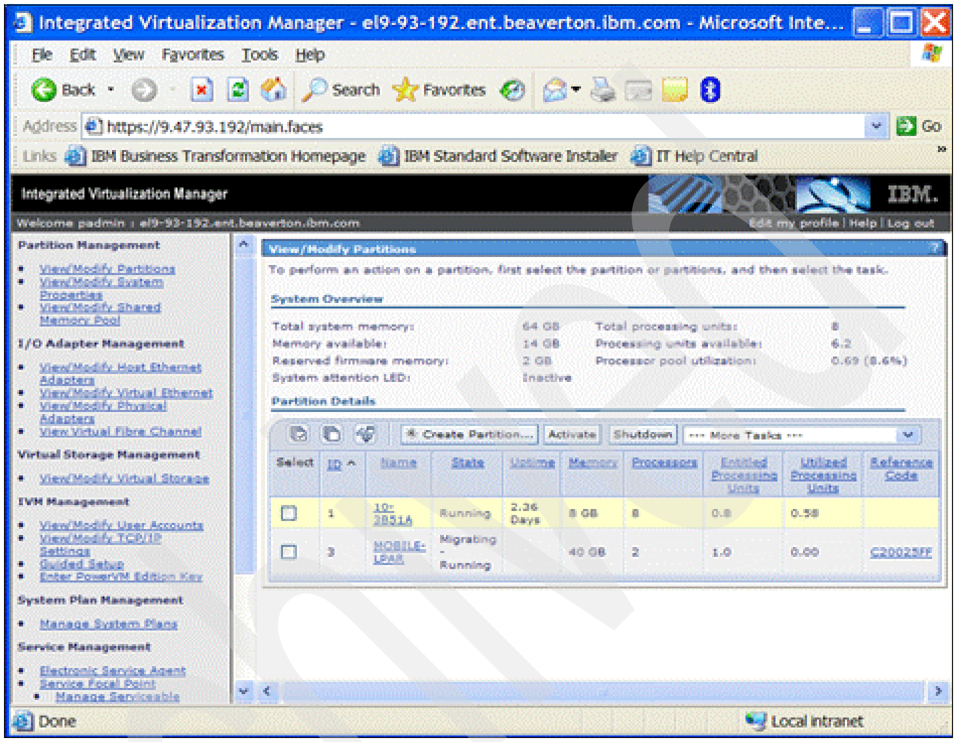
* Remote IVM:	9.47.93.192
Remote user ID:	padmin
Password:	*********

* Required field

Integrated Virtualization Manager
IBM.
Edit my profile | Help | Log out
Partition Management
View/Modify Partitions
View/Modify System Properties
View/Modify Shared Memory Pool
I/O Adapter Management
View/Modify Host Ethernet Adapters
View/Modify Virtual Ethernet
View/Modify Physical Adapters
View Virtual Fibre Channel
Virtual Storage Management
View/Modify Virtual Storage
IVM Management
View/Modify User Accounts
View/Modify TCP/IP Settings
Guided Setup
Enter PowerVM Edition Key
System Plan Management
Manage System Plans
Service Management
Electronic Service Agent
Validate
Migrate
Cancel

Figure 7-19 Migration validation

On the target Blade, observe that a new LPAR has been created with the same name as the LPAR on the source Blade. It has a state of Migrating - Running, as shown in Figure 7-20.



View/Modify Partitions

To perform an action on a partition, first select the partition or partitions, and then select the task.

System Overview

Total system memory:	64 GB	Total processing units:	8
Memory available:	14 GB	Processing units available:	6.2
Reserved firmware memory:	2 GB	Processor pool utilization:	0.69 (8.6%)
System attention LED:	Inactive		

Partition Details

Select	ID	Name	State	Uptime	Memory	Processors	Entitled Processing Units	Utilized Processing Units	Reference Code
☐	1	10-3B51A	Running	2.36 Days	8 GB	8	0.8	0.58	
☐	3	MOBILE-LPAR	Migrating - Running		40 GB	2	1.0	0.00	C20025FF

Figure 7-20   Observe the status of the migration on the destination server

7.6.3 What happens during the partition migration phase?

During the active migration of the LPAR, state information is transferred from the source to the target system. This state information includes such things as partition memory, processor state, virtual adapter state, NVRAM (non-volatile random access memory), and the LPAR configuration.

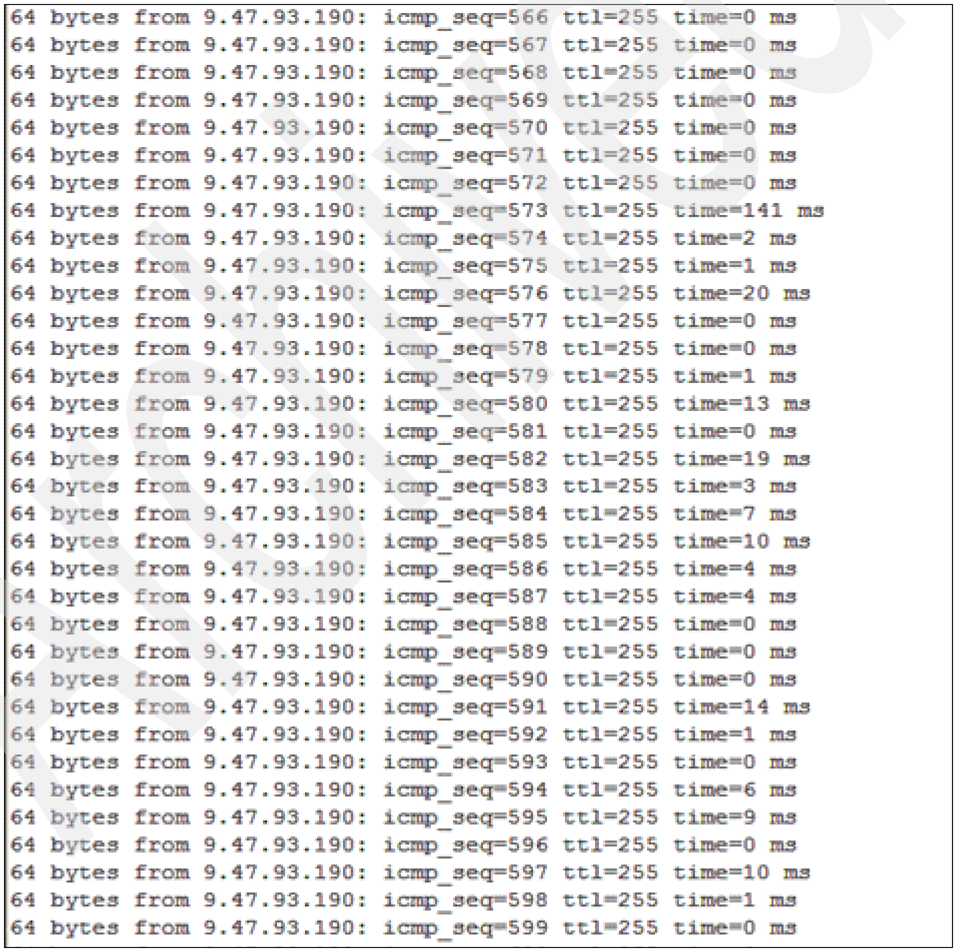
The following list details events and actions that occur during the migration:

- A partition shell is created on the target system. This shell partition is used to reserve the resources required to create the inbound LPAR, or processor entitlements, memory configuration, and virtual adapter configuration.
- A connection between the source and target systems and their respective Power Hypervisor is established through a device called the Virtual Asynchronous Service Interface (VASI) on the VIOS. The source and target VIOS use this new virtual device to communicate with the Power Hypervisor to gain access to the LPAR's state and to coordinate the migration. You can confirm the existence of this device with the **`lsdev`** command on the VIOS.

Tip: Use the **`vasistat`** command to display the statistics for the VASI device. Run this command on the source VIOS during the migration. Observe that Total Bytes to Transfer indicates the size of the memory copy and that Bytes Left to Transfer indicates how far the transfer has progressed.

- The virtual target devices and virtual SCSI adapters are created on the target system. Using the **`lsmap`** command on the target VIOS before the migration, notice that there are no virtual SCSI or virtual target device mappings. Running the same command after the migration shows that the virtual disk mappings have been created as part of the migration process.
- The LPAR's physical memory pages are copied to the shell LPAR on the target system. Using the **`topas`** command on the source VIOS, you might observe network traffic on the SEA as a result of the memory copy.
- Because the LPAR is still active, with DB2 still running, its state continues to change while the memory is copied. Memory pages that are modified during the transfer are marked as dirty. This process is repeated until the number of pages marked as dirty is no longer decreasing. At this point, the target system instructs the Power Hypervisor on the source system to suspend the LPAR.
- The LPAR confirms the suspension by quiescing all its running threads. The LPAR is now suspended.
- During the LPAR suspension, the source LPAR continues to send partition state information to the target server. The LPAR is then resumed.

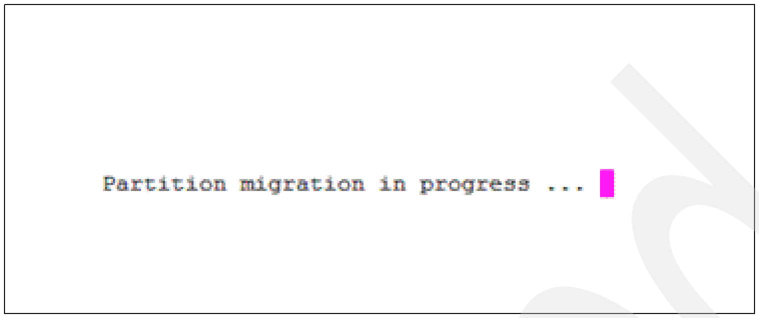
- The LPAR resumes execution on the target system. If the LPAR requires a page that has not yet been migrated, then it is demand-paged from the source system.
- The LPAR recovers its I/O operations. A gratuitous ARP request is sent on all virtual Ethernet adapters to update the ARP caches on all external switches and systems in the network. The LPAR is now active again.
- When the target system receives the last dirty page from the source system, the migration is complete. The period between the suspension and resumption of the LPAR lasts a few milliseconds, as you can see in Figure 7-21. In the meantime, the migrating LPAR displays the message `Partition Migration in progress ...` as shown in Figure 7-22 on page 333.



```
64 bytes from 9.47.93.190: icmp_seq=566 ttl=255 time=0 ms
64 bytes from 9.47.93.190: icmp_seq=567 ttl=255 time=0 ms
64 bytes from 9.47.93.190: icmp_seq=568 ttl=255 time=0 ms
64 bytes from 9.47.93.190: icmp_seq=569 ttl=255 time=0 ms
64 bytes from 9.47.93.190: icmp_seq=570 ttl=255 time=0 ms
64 bytes from 9.47.93.190: icmp_seq=571 ttl=255 time=0 ms
64 bytes from 9.47.93.190: icmp_seq=572 ttl=255 time=0 ms
64 bytes from 9.47.93.190: icmp_seq=573 ttl=255 time=141 ms
64 bytes from 9.47.93.190: icmp_seq=574 ttl=255 time=2 ms
64 bytes from 9.47.93.190: icmp_seq=575 ttl=255 time=1 ms
64 bytes from 9.47.93.190: icmp_seq=576 ttl=255 time=20 ms
64 bytes from 9.47.93.190: icmp_seq=577 ttl=255 time=0 ms
64 bytes from 9.47.93.190: icmp_seq=578 ttl=255 time=0 ms
64 bytes from 9.47.93.190: icmp_seq=579 ttl=255 time=1 ms
64 bytes from 9.47.93.190: icmp_seq=580 ttl=255 time=13 ms
64 bytes from 9.47.93.190: icmp_seq=581 ttl=255 time=0 ms
64 bytes from 9.47.93.190: icmp_seq=582 ttl=255 time=19 ms
64 bytes from 9.47.93.190: icmp_seq=583 ttl=255 time=3 ms
64 bytes from 9.47.93.190: icmp_seq=584 ttl=255 time=7 ms
64 bytes from 9.47.93.190: icmp_seq=585 ttl=255 time=10 ms
64 bytes from 9.47.93.190: icmp_seq=586 ttl=255 time=4 ms
64 bytes from 9.47.93.190: icmp_seq=587 ttl=255 time=4 ms
64 bytes from 9.47.93.190: icmp_seq=588 ttl=255 time=0 ms
64 bytes from 9.47.93.190: icmp_seq=589 ttl=255 time=0 ms
64 bytes from 9.47.93.190: icmp_seq=590 ttl=255 time=0 ms
64 bytes from 9.47.93.190: icmp_seq=591 ttl=255 time=14 ms
64 bytes from 9.47.93.190: icmp_seq=592 ttl=255 time=1 ms
64 bytes from 9.47.93.190: icmp_seq=593 ttl=255 time=0 ms
64 bytes from 9.47.93.190: icmp_seq=594 ttl=255 time=6 ms
64 bytes from 9.47.93.190: icmp_seq=595 ttl=255 time=9 ms
64 bytes from 9.47.93.190: icmp_seq=596 ttl=255 time=0 ms
64 bytes from 9.47.93.190: icmp_seq=597 ttl=255 time=10 ms
64 bytes from 9.47.93.190: icmp_seq=598 ttl=255 time=1 ms
64 bytes from 9.47.93.190: icmp_seq=599 ttl=255 time=0 ms
```

Figure 7-21 Suspension: Resumption of the LPAR during Migration

Note: In Figure 7-21 on page 332 we observed a suspension of about 141 ms.



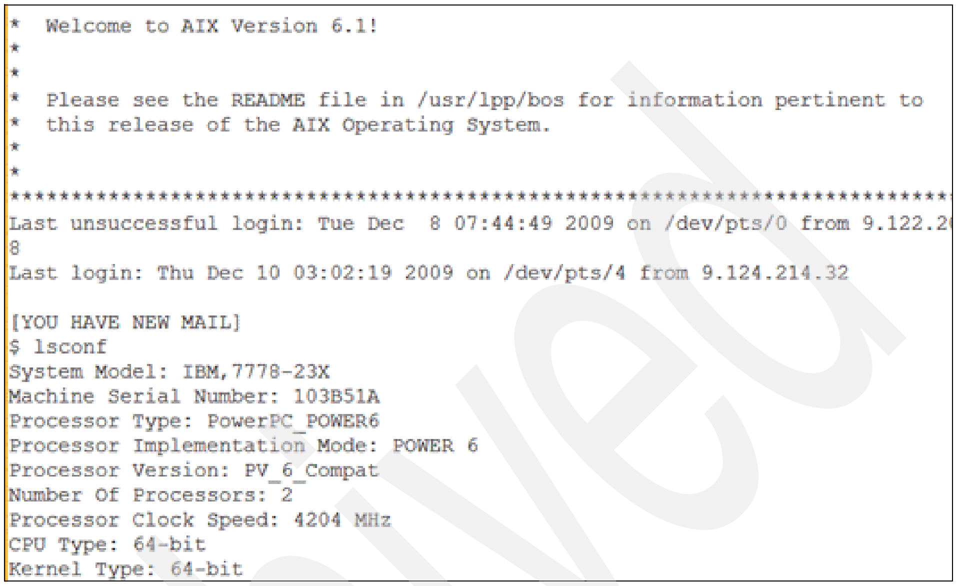
```
Partition migration in progress ...
```

Figure 7-22 Partition migration in progress

With the memory copy complete, the VIOS on the source system removes the virtual SCSI server adapters associated with the LPAR and removes any device to LUN mapping that existed previously.

The LPAR is deleted from the source Blade. The LPAR is now in a Running state on the target Blade. The migration is 100% complete.

Now that the LPAR is running on the other Blade, run the **lsconf** command to confirm that the serial number has changed with the physical hardware. See Figure 7-23.



```
*  Welcome to AIX Version 6.1!
*
*
*  Please see the README file in /usr/lpp/bos for information pertinent to
*  this release of the AIX Operating System.
*
*
******************************************************************************
Last unsuccessful login: Tue Dec  8 07:44:49 2009 on /dev/pts/0 from 9.122.2
8
Last login: Thu Dec 10 03:02:19 2009 on /dev/pts/4 from 9.124.214.32

[YOU HAVE NEW MAIL]
$ lsconf
System Model: IBM,7778-23X
Machine Serial Number: 103B51A
Processor Type: PowerPC_POWER6
Processor Implementation Mode: POWER 6
Processor Version: PV_6_Compat
Number Of Processors: 2
Processor Clock Speed: 4204 MHz
CPU Type: 64-bit
Kernel Type: 64-bit
```

Figure 7-23 lsconf output after migration occurred

To confirm and verify that DB2 is not impacted by the migration, check the DB2 alert log for any errors. The ssh login sessions on MOBILE-LPAR remained active and did not suffer any connectivity issues as a result of the live migration (see Figure 7-21 on page 332).

Mobility activity is logged on the LPAR and the source and target VIOS. Review the logs with the **errpt** (AIX) and **errlog** (VIOS) commands. On AIX, notice messages similar to CLIENT_PMIG_STARTED and CLIENT_PMIG_DONE. Additional information from DRMGR, on AIX is also logged to syslog (for instance, starting CHECK phase for partition migration). On the VIOS, find messages relating to the suspension of the LPAR and the migration status (Client partition suspend issued and Migration completed successfully).

Output of the **errpt** command from the AIX LPAR and from both VIOSs relate to the migration activity.

7.6.4 Post migration observations

During the migration, we used **nmon** to capture data that we analyze hereafter.

The migration took about 20 minutes to complete. The LPAR being moved was configured with 40 GB of memory. Most of the time required for the migration was for the copying of the LPAR's memory from the source to the target system. The suspend of the LPAR itself lasted no more than 141 milliseconds. Consider using a high-performance network between the source and target systems. Also, prior to the migration, we suggest reducing the LPAR's memory update activity. Taking these steps improves the overall performance of the migration. (See “Mover Service Partition” on page 288.)

Note: The time to migrate depends on the quantity of memory allocated to the logical partition. The more memory allocated, the more time you need to migrate.

Looking at the error reports show the migration has occurred and reveals the status of the migration.

LPM has enormous potential for dramatically reducing scheduled downtime for system maintenance activities. Being able to perform scheduled activities, such as preventative hardware maintenance or firmware updates, without disruption to user applications and services is a significant enhancement to any System p environment. Additionally, this technology can assist in managing workloads within a System p landscape. It gives administrators the power to adjust resource usage across an entire farm of System p servers. LPARs can be moved to other physical servers to help balance workload demands.

Note: For the tests, we used Power Blade systems, each configured with IVM and one VIOS (one entire CPU, 3 GB of memory). The mobile LPAR had one entire CPU and 40 GB of memory allocated to it. As such, screen captures taken from the IVM might differ from screen captures taken with an HMC.

OLTP workload

The OLTP workload was running when the migration started. It ran on the source server during the pre-migration phase. There was a short interval where the throughput dipped. This was when the hand-over happened. The workload resumed execution on the target server after the handover. See Figure 7-24.

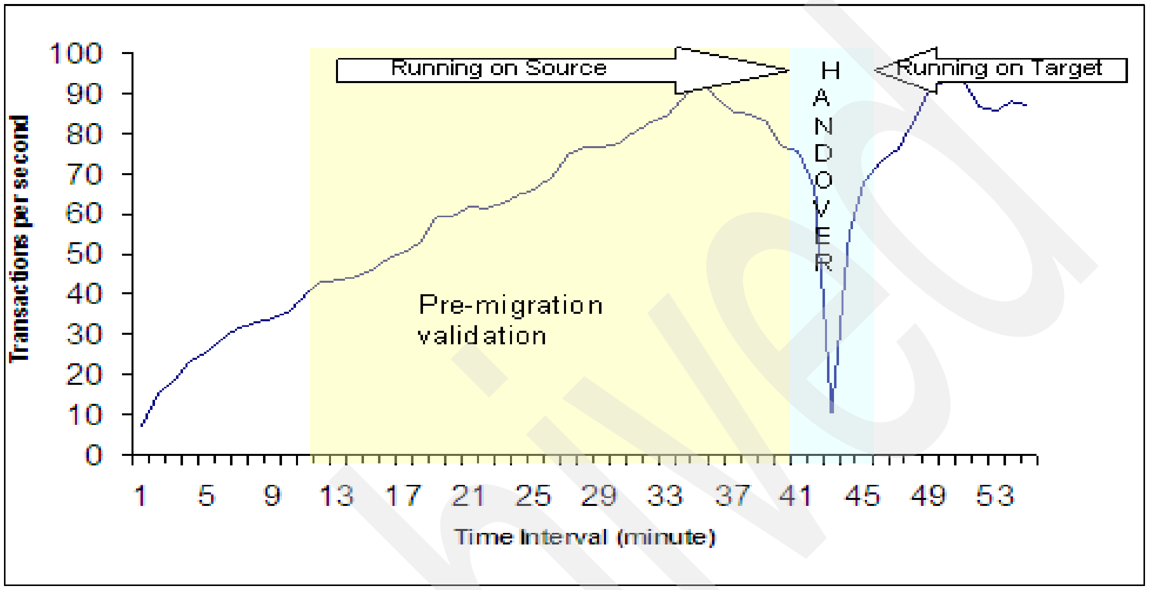


Figure 7-24 LPM hand over

The following diagrams show you what happened at the moment the real hand over occurs.

The migration process itself started at 11:20 and finished around 11:39. The load was run about 11:00 and finished after 12:00. The handover from the source VIOS to the destination system took place at 11:32. The following processes were running when the migration command was issued:

- ctrlproc, the mover process
- migrlpar, the migration command that performs the validation and migration of the logical partition.
- seaproc process, part of the shared Ethernet network and related to network.
- accessproc process.

CPU level

In the mobile partition, we see a lessening of the CPU activity at the moment the hand-over happened from the source to the destination system. See Figure 7-25.

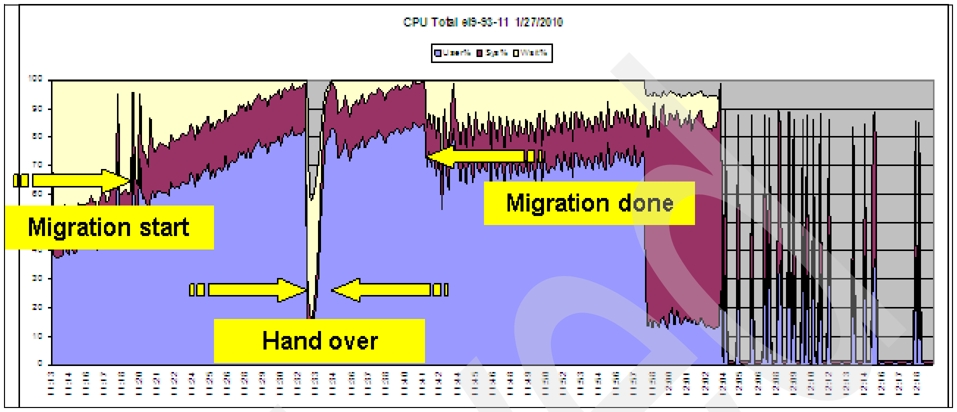


Figure 7-25 Mobile partition: Total CPU

Processes Run Queue slowly increased to attain a maximum value of about 80 and decreased to its nominal value after the hand over finished, as shown in Figure 7-26.

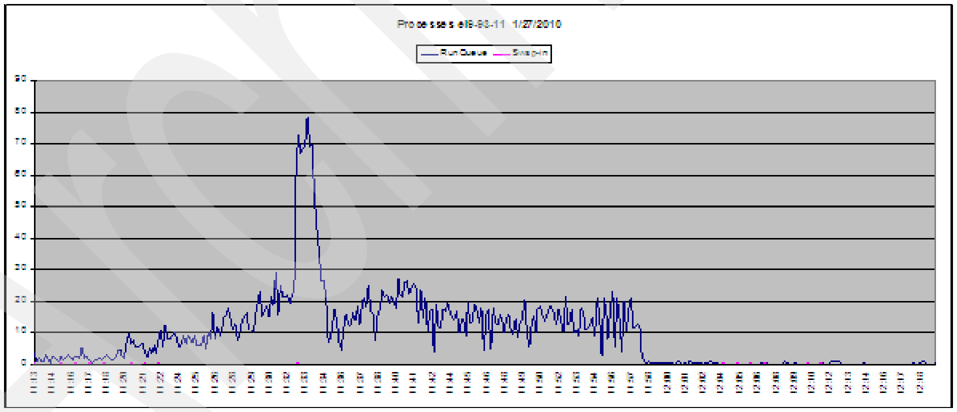


Figure 7-26 Mobile partition: Processes RunQueue

The VIOSs in Figure 7-27 and Figure 7-28, reveals that the source mover service partition uses about 40% more CPU than its entitled capacity during the whole migration process. The migration process started at 11:20 and finished at 11:39. In the same way VIOS2 exceeded its entitled capacity by about 30%.

Note: After the migration is finished, you can check the error report for Client Partition Migration Completed. After you see this in the error report, a few remaining tasks still need to be finalized.

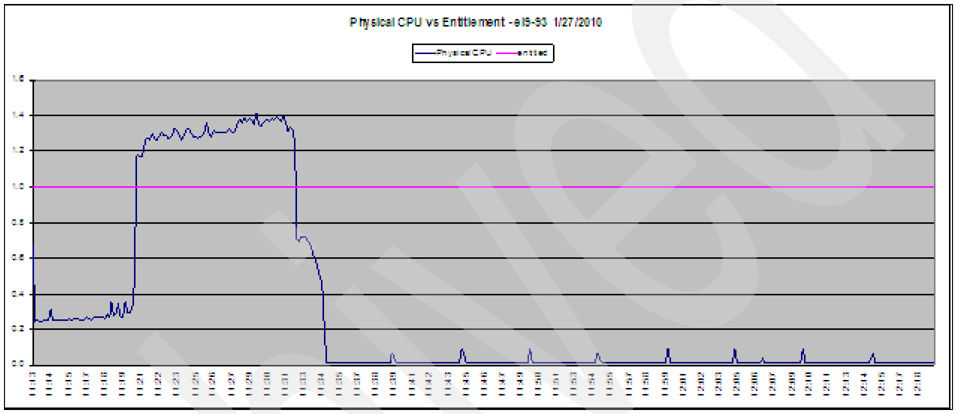


Figure 7-27 VIOS1: Physical CPU versus EC

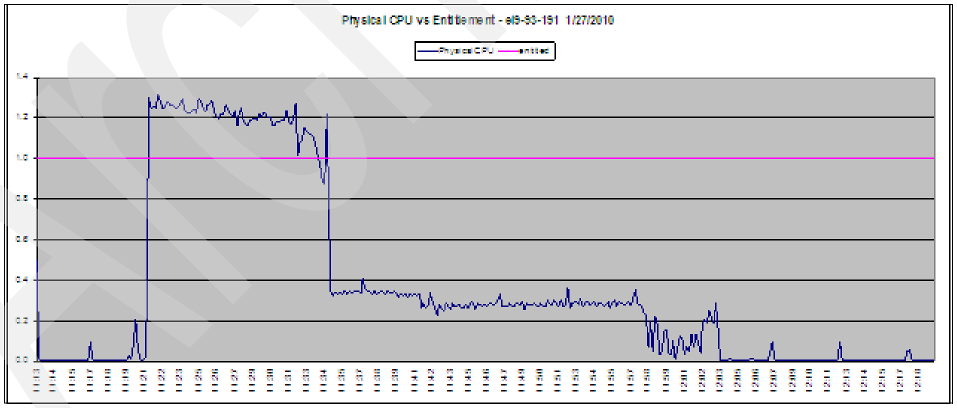


Figure 7-28 VIOS2: Physical CPU versus EC

Disk level

In Figure 7-29 we clearly see the "gap", moment corresponding to the point where the source system releases the disks for the destination system (11:32).

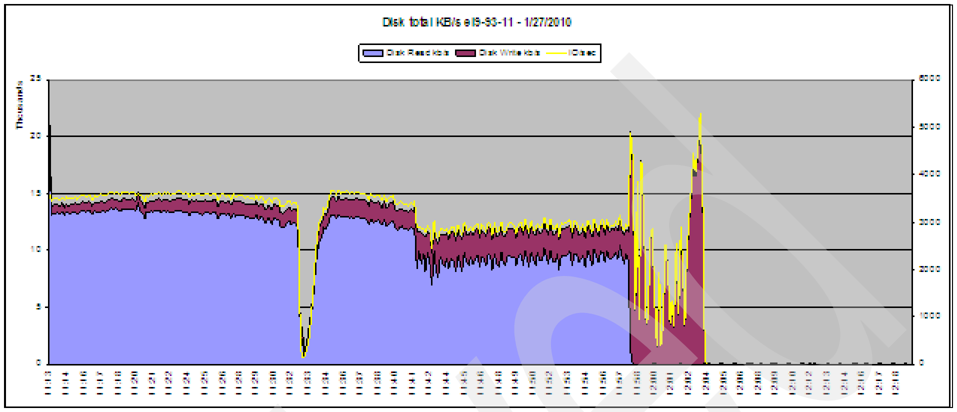


Figure 7-29 Mobile partition: Disk use: Total (KB/second)

Figure 7-30, shows a sharp decline in disk activity (hdisk5), again showing the time (11:32) when the source system hands over to the destination server.

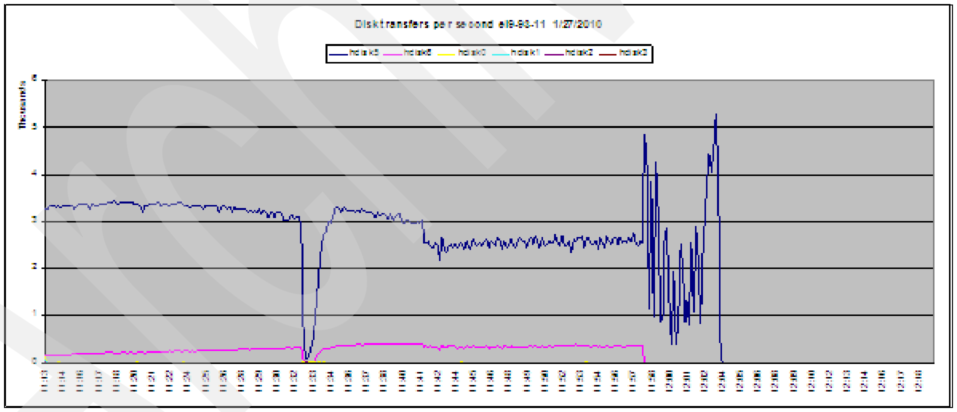


Figure 7-30 Mobile partition: Disk transfer

You can see the same in Figure 7-31 at the disk adapter level.

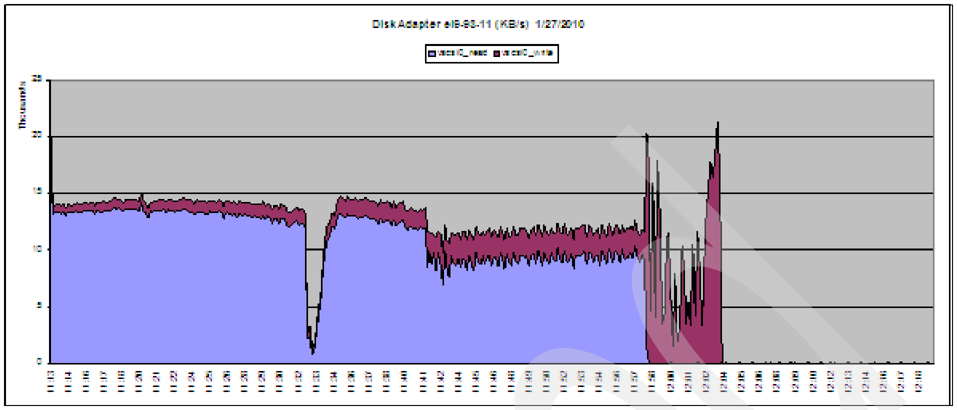


Figure 7-31 Mobile partition: Disk adapter use

This phenomenon is much more clear when looking at the VIOSs.

VIOS1 shows the I/O activity prior the hand-over as shown in Figure 7-32. We see that after the migration, VIOS1 has nothing more to deal with I/O throughput.

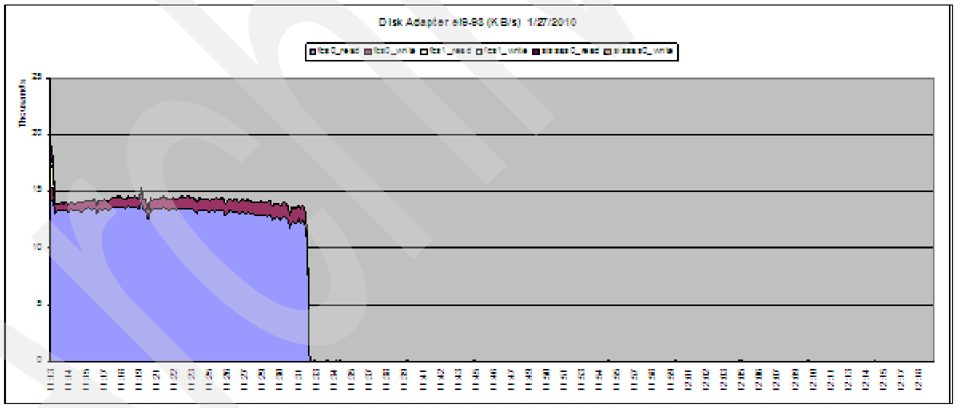


Figure 7-32 VIOS1: Disk adapter use

However, after the migration occurred (11:32), VIOS2 takes over all I/O, as shown in Figure 7-33. From that moment, the mobile partition gets its I/O traffic through VIOS2 on the destination server.

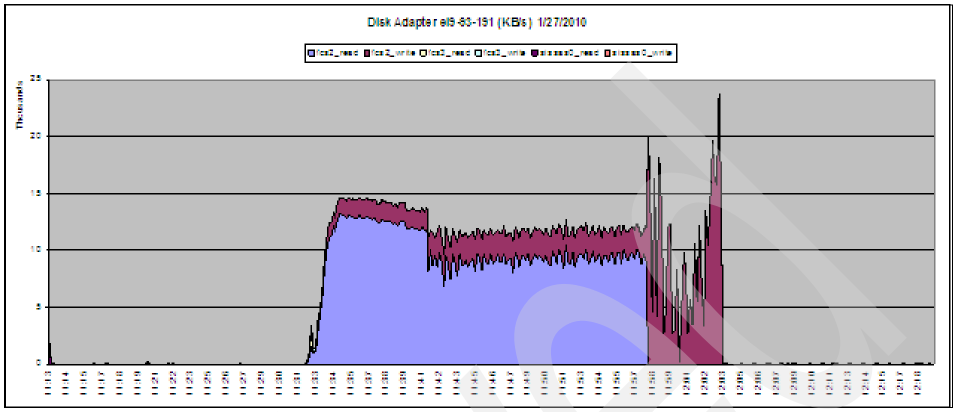


Figure 7-33 VIOS2: Disk adapter use

Memory level

When looking at memory, we can hardly see a change, as shown in Figure 7-34. This is as expected, as during the migration the application binds to the destination system's memory while its already bound memory is copied over the network or is freed when unused by the application. Nearly all of the memory is allocated. We see the load on the system finished at 12:04, which released the memory while the migration itself finished at 11:39.

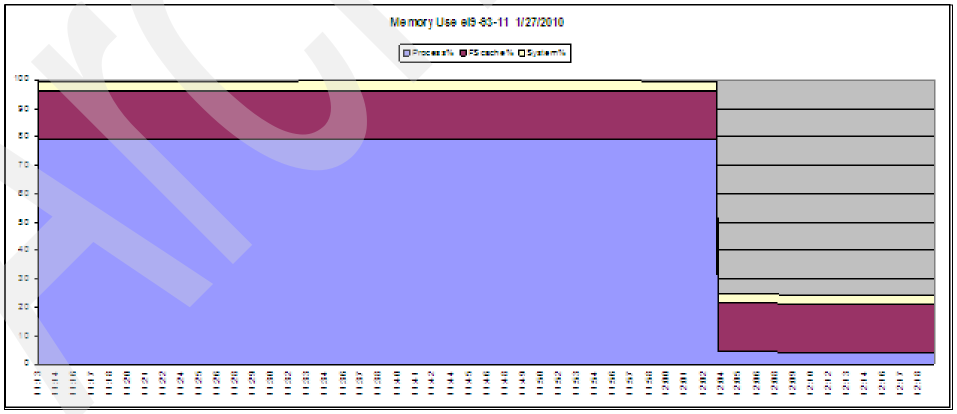


Figure 7-34 Mobile partition: Memory use

Memory usage of each VIOS is not impacted by the migration, as demonstrated in Figure 7-35 and Figure 7-36. We see later in “Memory level” on page 351 that the same happens without any load on the system, which correlates our findings.

Recommendation: Consider decreasing network traffic on your network while the migration is running.

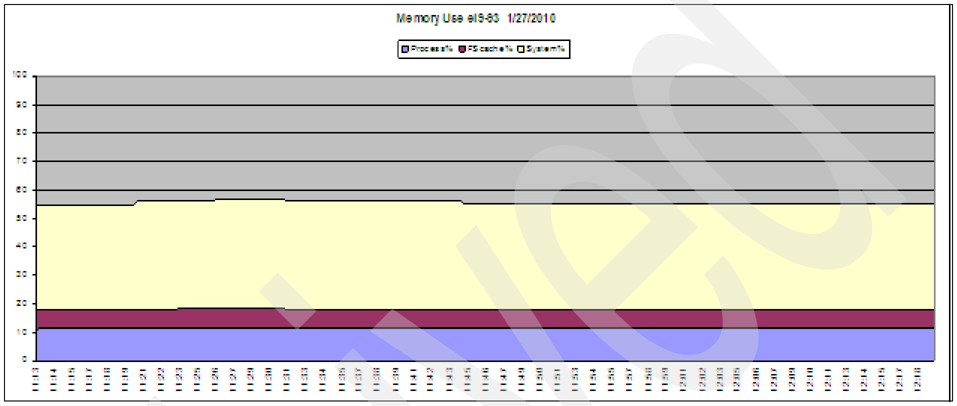


Figure 7-35 VIOS1: Memory use

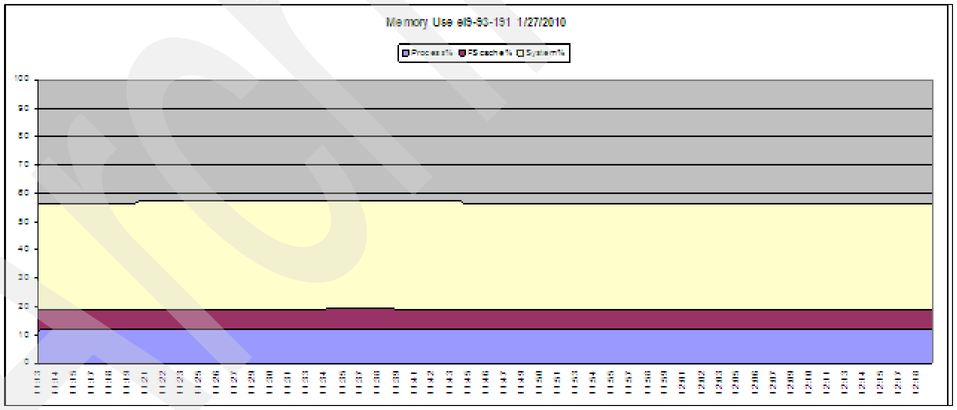


Figure 7-36 VIOS2: Memory use

Network level

The only downtime that occurs during a migration is the network connectivity, which might be inaccessible for about two seconds. Those two seconds are negligible for most clients that are connected to the application.

In the example and test we made, we did not notice the loss of connectivity. at Figure 7-37 does not show a loss, but a drastic decrease in activity during a fraction of a second followed by noticeable raise in power.

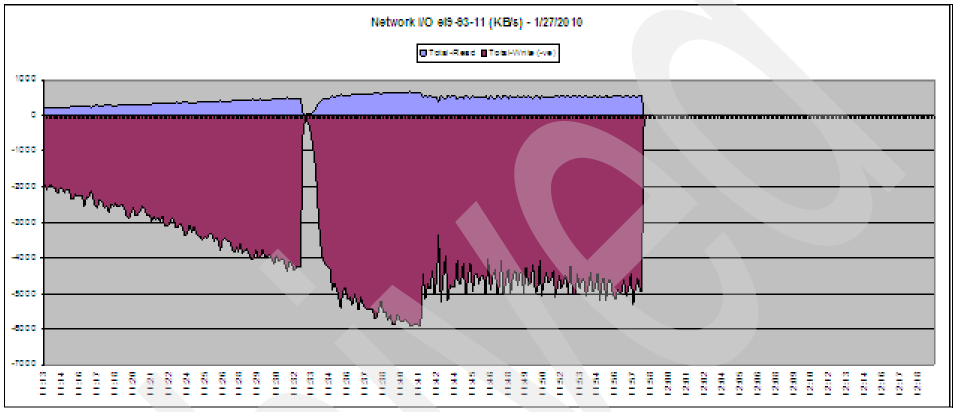
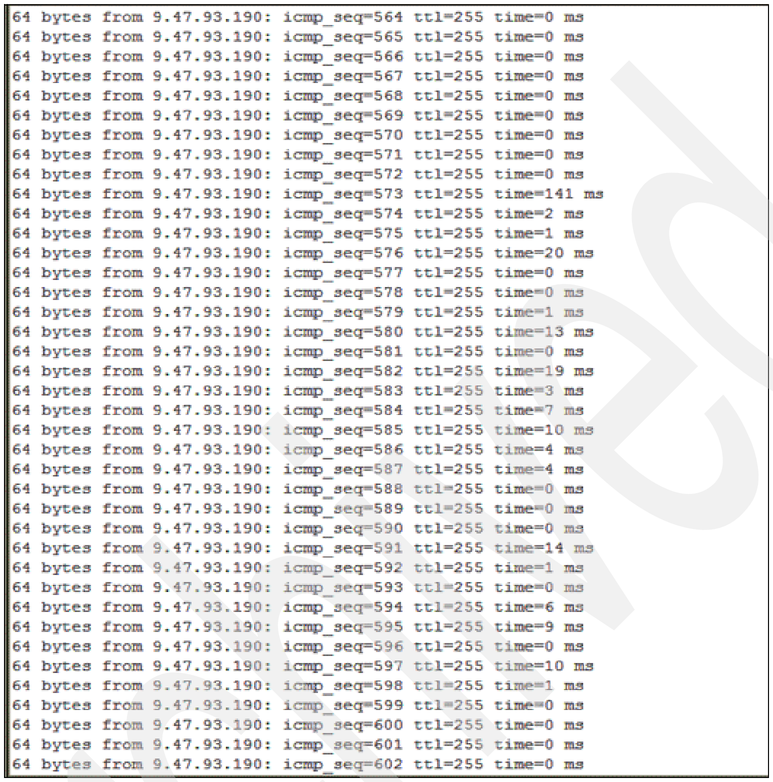


Figure 7-37 Mobile partition: Network I/O

Recommendation: Although performing a LPM, keep the network traffic in the network adapter as minimal as possible. This ensures the maximum bandwidth for the VIOS MSP to proceed with the migration.

We captured the results of a ping command that enables us to evaluate more precisely the down time of the network connectivity (Figure 7-38). In this example, the inaccessibility of the network was of about 141 ms.



```
64 bytes from 9.47.93.190: icmp_seq=564 ttl=255 time=0 ms
64 bytes from 9.47.93.190: icmp_seq=565 ttl=255 time=0 ms
64 bytes from 9.47.93.190: icmp_seq=566 ttl=255 time=0 ms
64 bytes from 9.47.93.190: icmp_seq=567 ttl=255 time=0 ms
64 bytes from 9.47.93.190: icmp_seq=568 ttl=255 time=0 ms
64 bytes from 9.47.93.190: icmp_seq=569 ttl=255 time=0 ms
64 bytes from 9.47.93.190: icmp_seq=570 ttl=255 time=0 ms
64 bytes from 9.47.93.190: icmp_seq=571 ttl=255 time=0 ms
64 bytes from 9.47.93.190: icmp_seq=572 ttl=255 time=0 ms
64 bytes from 9.47.93.190: icmp_seq=573 ttl=255 time=141 ms
64 bytes from 9.47.93.190: icmp_seq=574 ttl=255 time=2 ms
64 bytes from 9.47.93.190: icmp_seq=575 ttl=255 time=1 ms
64 bytes from 9.47.93.190: icmp_seq=576 ttl=255 time=20 ms
64 bytes from 9.47.93.190: icmp_seq=577 ttl=255 time=0 ms
64 bytes from 9.47.93.190: icmp_seq=578 ttl=255 time=0 ms
64 bytes from 9.47.93.190: icmp_seq=579 ttl=255 time=1 ms
64 bytes from 9.47.93.190: icmp_seq=580 ttl=255 time=13 ms
64 bytes from 9.47.93.190: icmp_seq=581 ttl=255 time=0 ms
64 bytes from 9.47.93.190: icmp_seq=582 ttl=255 time=19 ms
64 bytes from 9.47.93.190: icmp_seq=583 ttl=255 time=3 ms
64 bytes from 9.47.93.190: icmp_seq=584 ttl=255 time=7 ms
64 bytes from 9.47.93.190: icmp_seq=585 ttl=255 time=10 ms
64 bytes from 9.47.93.190: icmp_seq=586 ttl=255 time=4 ms
64 bytes from 9.47.93.190: icmp_seq=587 ttl=255 time=4 ms
64 bytes from 9.47.93.190: icmp_seq=588 ttl=255 time=0 ms
64 bytes from 9.47.93.190: icmp_seq=589 ttl=255 time=0 ms
64 bytes from 9.47.93.190: icmp_seq=590 ttl=255 time=0 ms
64 bytes from 9.47.93.190: icmp_seq=591 ttl=255 time=14 ms
64 bytes from 9.47.93.190: icmp_seq=592 ttl=255 time=1 ms
64 bytes from 9.47.93.190: icmp_seq=593 ttl=255 time=0 ms
64 bytes from 9.47.93.190: icmp_seq=594 ttl=255 time=6 ms
64 bytes from 9.47.93.190: icmp_seq=595 ttl=255 time=9 ms
64 bytes from 9.47.93.190: icmp_seq=596 ttl=255 time=0 ms
64 bytes from 9.47.93.190: icmp_seq=597 ttl=255 time=10 ms
64 bytes from 9.47.93.190: icmp_seq=598 ttl=255 time=1 ms
64 bytes from 9.47.93.190: icmp_seq=599 ttl=255 time=0 ms
64 bytes from 9.47.93.190: icmp_seq=600 ttl=255 time=0 ms
64 bytes from 9.47.93.190: icmp_seq=601 ttl=255 time=0 ms
64 bytes from 9.47.93.190: icmp_seq=602 ttl=255 time=0 ms
```

Figure 7-38 Snapshot of a ping command

On VIOS1 we see that it is writing away to the other system at a speed of about 75 MBps until 11:32, moment where the hand-over happened. See Figure 7-39.

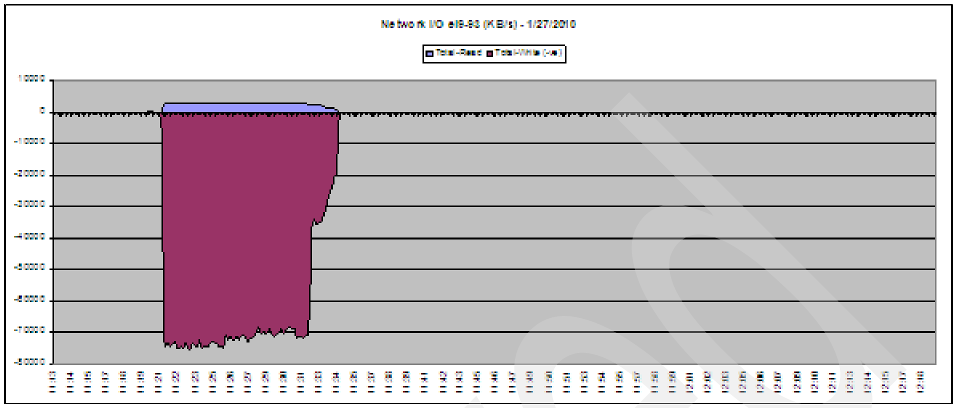


Figure 7-39 VIOS1: Network I/O

Similarly on VIOS2, we see a symmetrical read from the source system at exactly the same speed. See Figure 7-40.

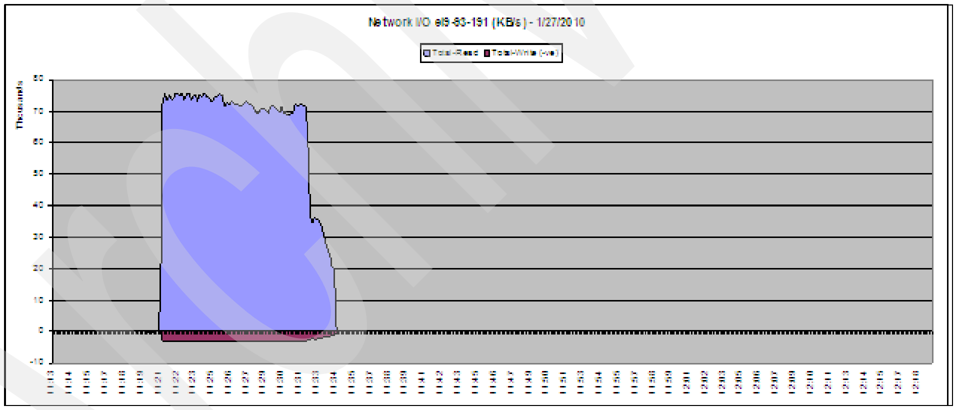


Figure 7-40 VIOS2: Network I/O

LPM: Dry run

To measure the impact of the VIOSs on the mobile partition during a LPM, we ran a dry run (migration without any particular load), while the partition was only running AIX. This demonstrates that the VIOSs do not bring any overhead. Moreover it shows the similarities in the graphs we got.

Compared to the “OLTP workload” on page 336, we observe a similar behavior, disregard the system is loaded or not.

CPU level

We observe in the graphs that the source VIOS, which is the initiator of the migration, has its CPU waving from about 0.1 to 0.25 CPU units. At the moment the migration process starts (02:46), the CPU of the VIOS initiator (VIOS1) increases instantly and even goes over its entitled capacity to attain about 125% of the EC, while the destination VIOS also increases its CPU capacity similarly. This lasts until the mobile partition is migrated (03:00). This behavior is similar to the testing we did with a loaded system with the exception of the time until the hand-over occurs, which is shorter for a non-loaded system than for a loaded one. However, the global migration time is slightly similar. For more information refer to “CPU level” on page 337.

Attention: From Figure 7-41 and Figure 7-42 on page 347 we notice the migration activity to migrate the mobile partition from the source to the destination server. The migration started at 02:46 and ended at 03:01. However, there are still tasks to be performed to finalize the migration, including the copying of remaining memory pages still on the source server. Refer to 7.5, “Migration process and flow” on page 316 for more information about the migration processes.

Note: The recommendation to configure the VIOSs is to uncap the CPU. The weight factor needs to be carefully set to allow the VIOSs to get CPU prior to any other type of partition.

Remember that the VIOSs are the main virtualization servers that cannot afford lack of resources, or your LPAR suffers or worse, shuts down.

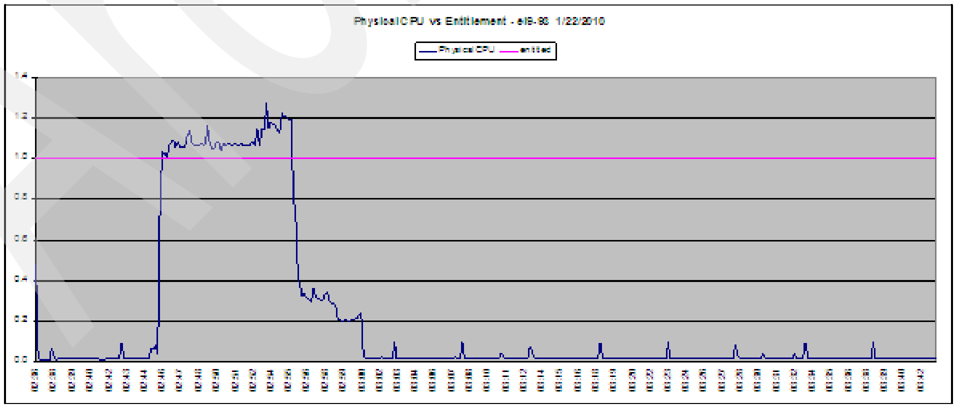


Figure 7-41 VIOS1: Physical CPU

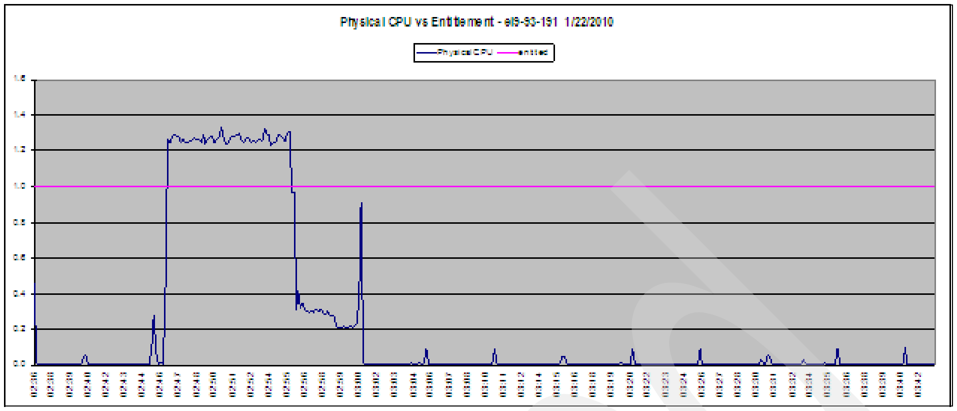


Figure 7-42 VIOS2: Physical CPU

The mobile partition is not influenced because it has nothing to do with the migration process. Moreover, during this test we did not run any load on it. System activity here is mainly related to running Java processes. See Figure 7-43.

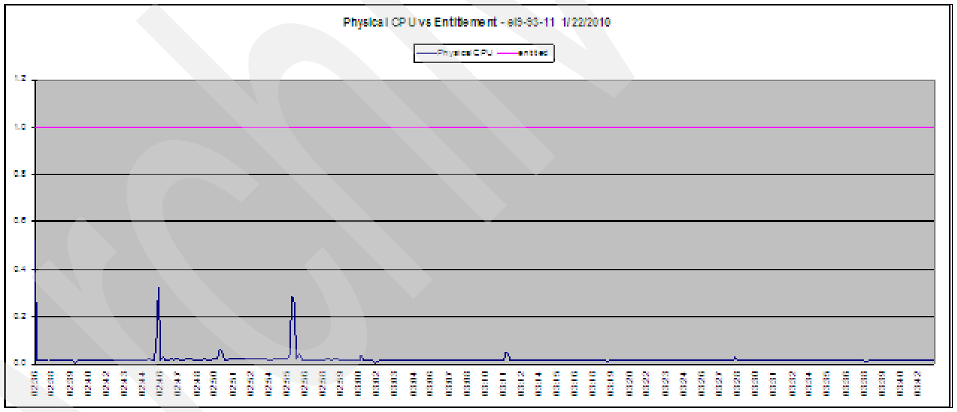


Figure 7-43 Mobile LPAR: Physical CPU

In addition, the average number of kernel threads in the run queues of both VIOSs show a similar significant increase, as shown in Figure 7-44.

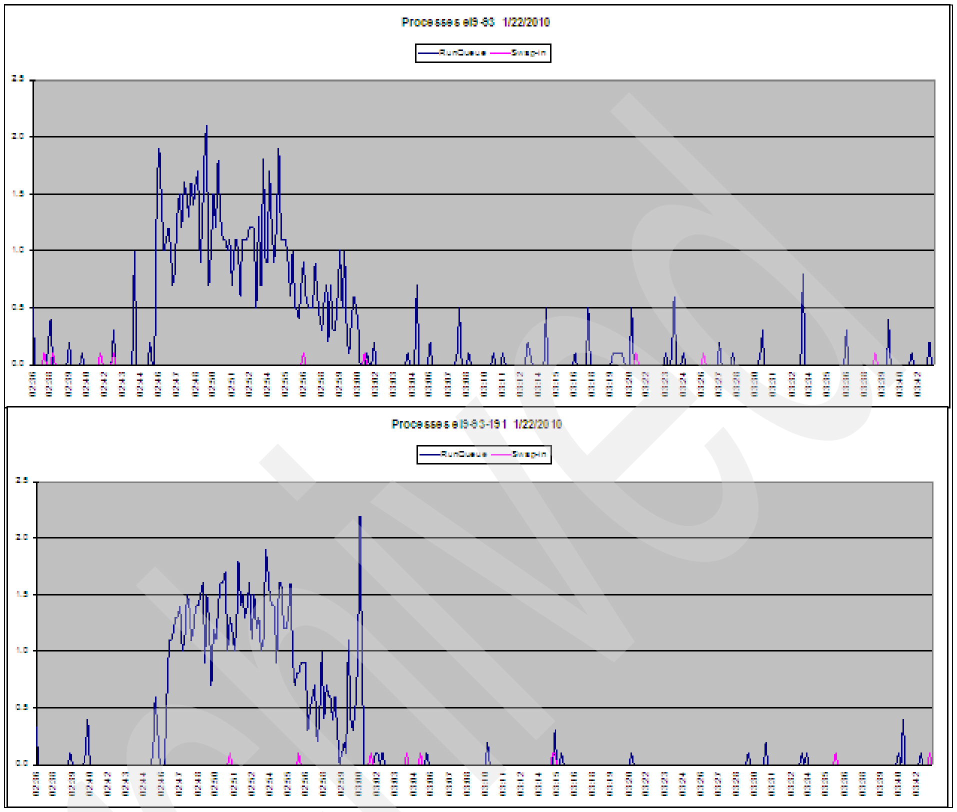


Figure 7-44 VIOSs run queue

When the migration completed at 03:01, we observed the run queue of the mobile partition attained a peak, as shown in Figure 7-45. This behavior is slightly similar to the one observed during our run with a heavy load on the system. In that situation, the run queue increased slowly until reaching a peak (at the migration hand-over time) to decrease in the same manner to its original value.

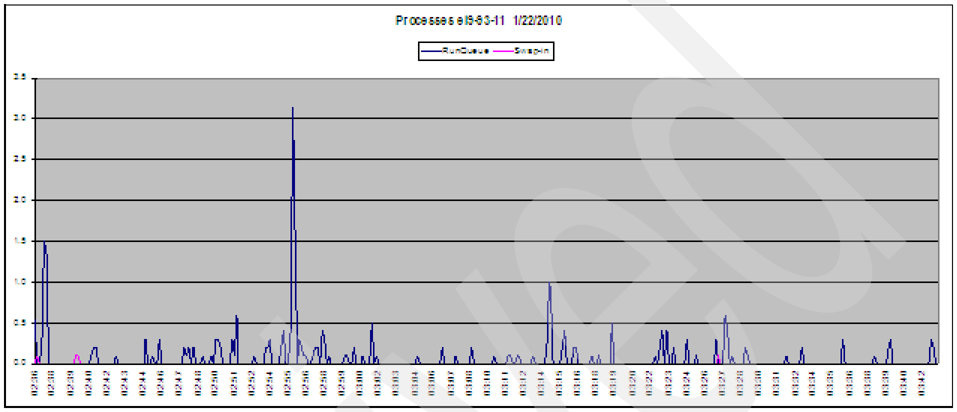


Figure 7-45 Mobile partition: Run queue

Disk level

We do not expect any particular disk activity on the mobile partition because there is no load on it. This is reflected in Figure 7-46.

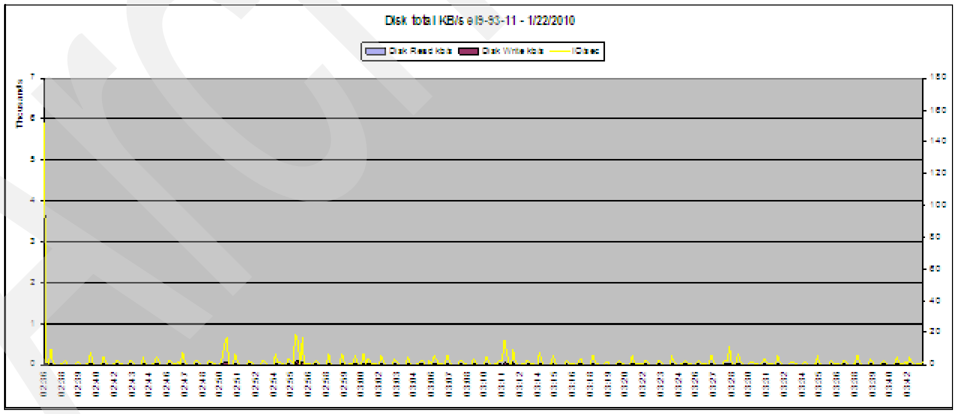


Figure 7-46 Mobile: Disk use

In the same way, no disk activity appears on the VIOSs, as the client does not generate any traffic. This is shown in Figure 7-47 and Figure 7-48.

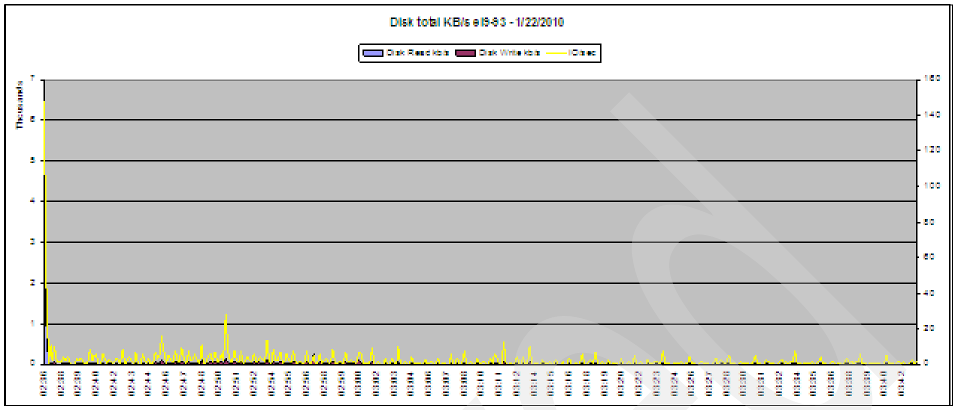


Figure 7-47 VIOS1: Disk use

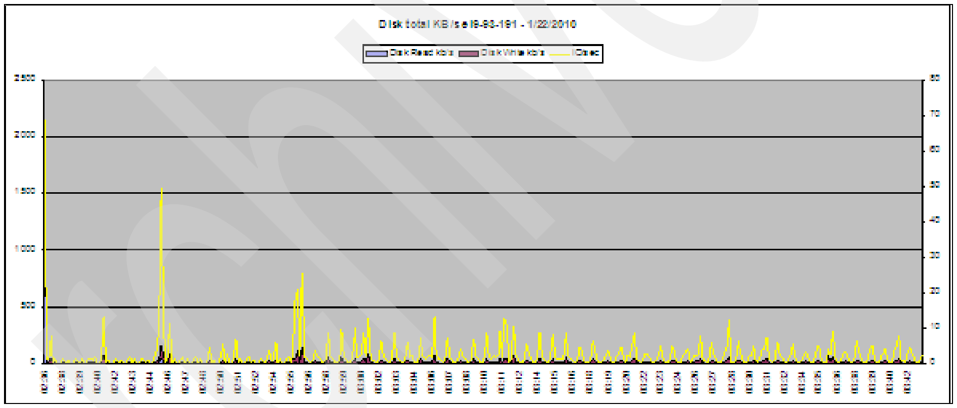


Figure 7-48 VIOS2: Disk use

Memory level

The observation shows that the VIOSs are only using about 1.6 GB of the total allocated memory, as shown in Figure 7-49 and Figure 7-50.

This is neither more nor less than the VIOS is actually using when doing nothing. In other words, the migration does not demand more memory during the transfer. However, this behavior can be influenced by a heavy network traffic, which demands memory to handle the network IP packets.

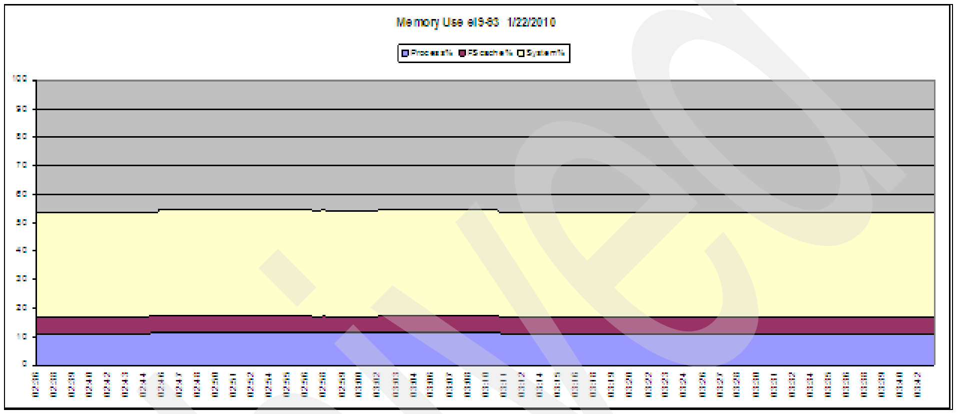


Figure 7-49 VIOS1: Memory use

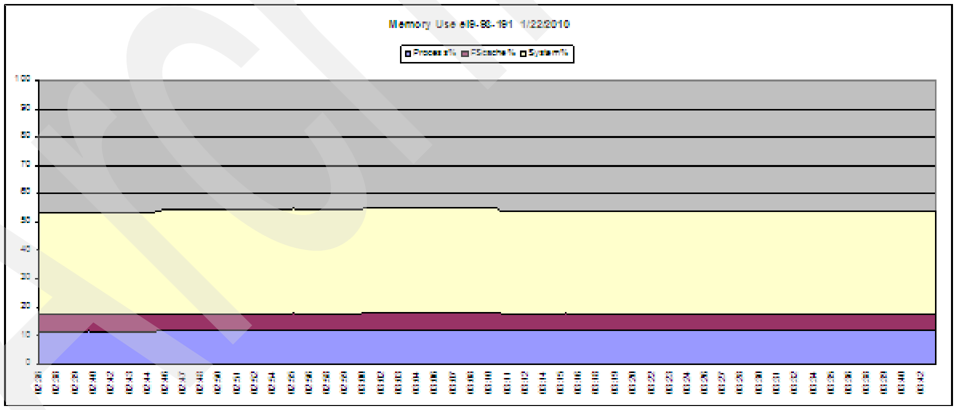


Figure 7-50 VIOS2: Memory use

The mobile partition is not affected, so it does not show any particular memory activity, as shown in Figure 7-51. This is the same result we observed while the system was loaded. Refer to “Memory level” on page 341 for more details.

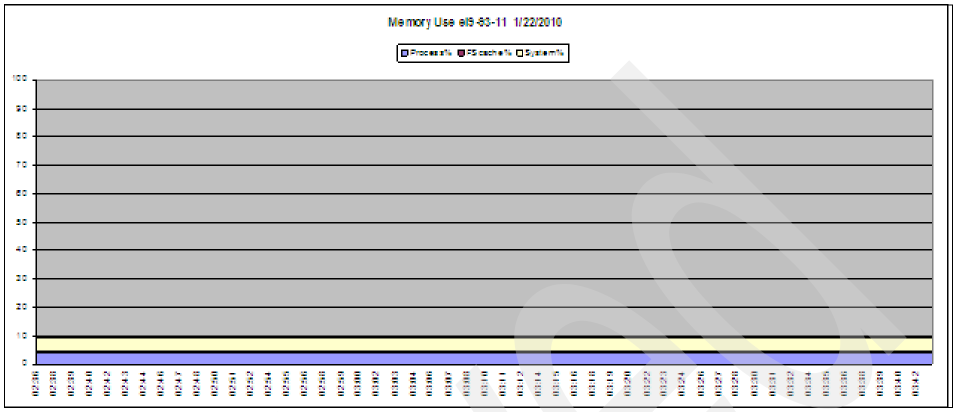


Figure 7-51 Mobile partition: Memory use

Network level

The network is obviously the part that reveals what is happening during a migration. As we have already seen in this chapter, the network is doing the job of migrating from the source to the destination server.

Figure 7-52 on page 353 and Figure 7-53 on page 353 show that in the window where the migration from the source to the destination server occurs, the network reaches 75 MBps. When VIOS1 is pushing information to VIOS2, VIOS2 is receiving that information. This is why both graphs are symmetrical.

Moreover, in Figure 7-52 we see little read activity. Similarly in Figure 7-53 we see write activity. That activity is like a handshaking between both MSPs in the migration process to transmit all necessary information to rebuild the LPAR's profile, and to copy the memory to the destination system.

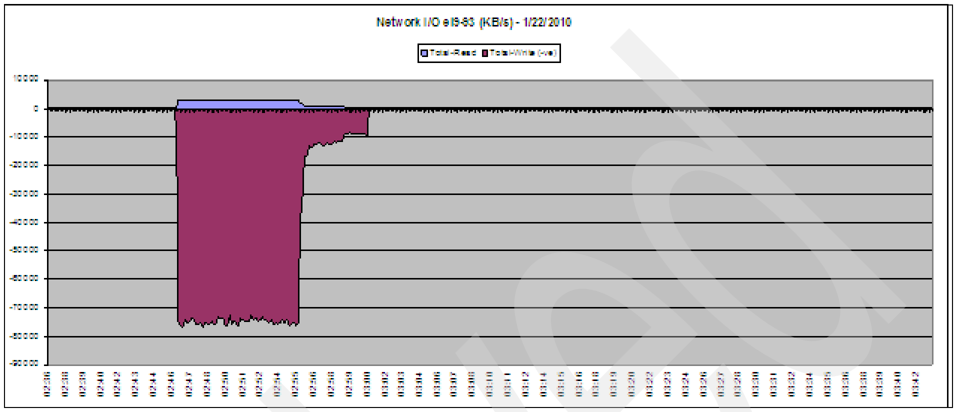


Figure 7-52 VIOS1: Network I/O

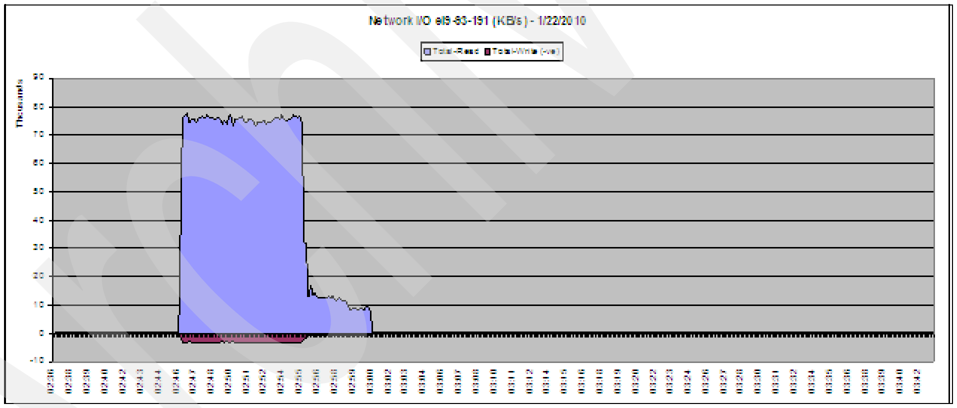


Figure 7-53 VIOS2: Network I/O

However, our mobile partition does not show any particular network activity as shown in Figure 7-54, because we did not load that system, and because it is not acting in the migration process itself.

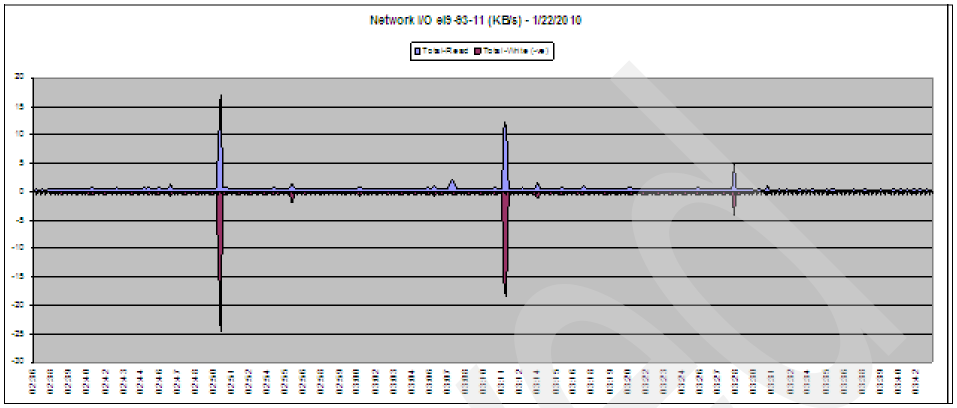


Figure 7-54 Mobile partition: Network I/O

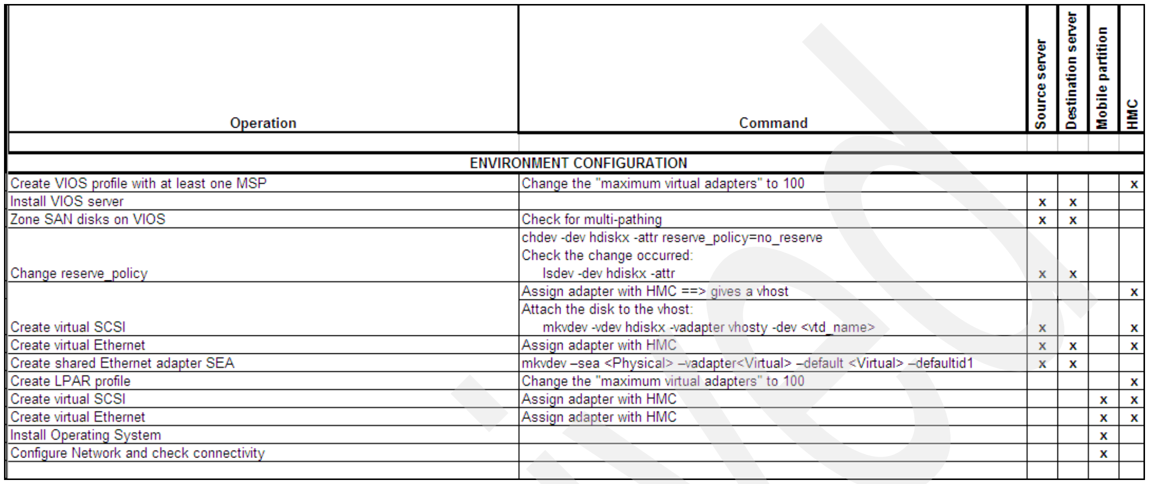
Note: If multiple network interfaces are available on a MSP, it is possible through the HMC command line to select which IP address the mover uses to transport client partition data during a LPM event. HMC level V7r3.5.0 or later is required for this function.

7.7 LPM: Summary of actions

The next figures summarize the necessary actions to undertake, before, during and after the migration process. These figures are followed by two figures outlining the major errors you can encounter when using the LPM capabilities of your servers.

7.7.1 LPM: Environment configuration

Figure 7-57 on page 356 shows the operations related with environment configuration.

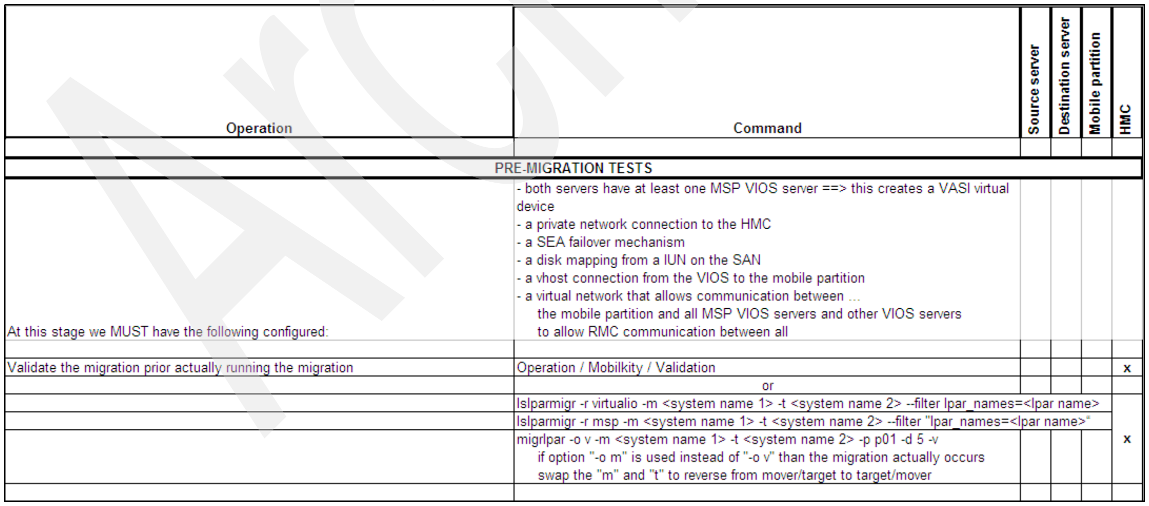


Operation	Command	Source server	Destination server	Mobile partition	HMC
ENVIRONMENT CONFIGURATION					
Create VIOS profile with at least one MSP	Change the "maximum virtual adapters" to 100				x
Install VIOS server		x	x		
Zone SAN disks on VIOS	Check for multi-pathing	x	x		
Change reserve_policy	chdev -dev hdiskx -attr reserve_policy=no_reserve Check the change occurred: lsdev -dev hdiskx -attr	x	x		
	Assign adapter with HMC ==> gives a vhost				x
Create virtual SCSI	Attach the disk to the vhost: mkvdev -vdev hdiskx -vadapter vhosty -dev <vtd_name>	x			x
Create virtual Ethernet	Assign adapter with HMC	x	x		x
Create shared Ethernet adapter SEA	mkvdev –sea <Physical> –vadapter<Virtual> –default <Virtual> –defaultid1	x	x		
Create LPAR profile	Change the "maximum virtual adapters" to 100				x
Create virtual SCSI	Assign adapter with HMC			x	x
Create virtual Ethernet	Assign adapter with HMC			x	x
Install Operating System				x	
Configure Network and check connectivity				x	

Figure 7-55 LPM: Environment configuration

7.7.2 LPM: Pre-migration tests

Figure 7-56 shows the pre migrations tests (validation).

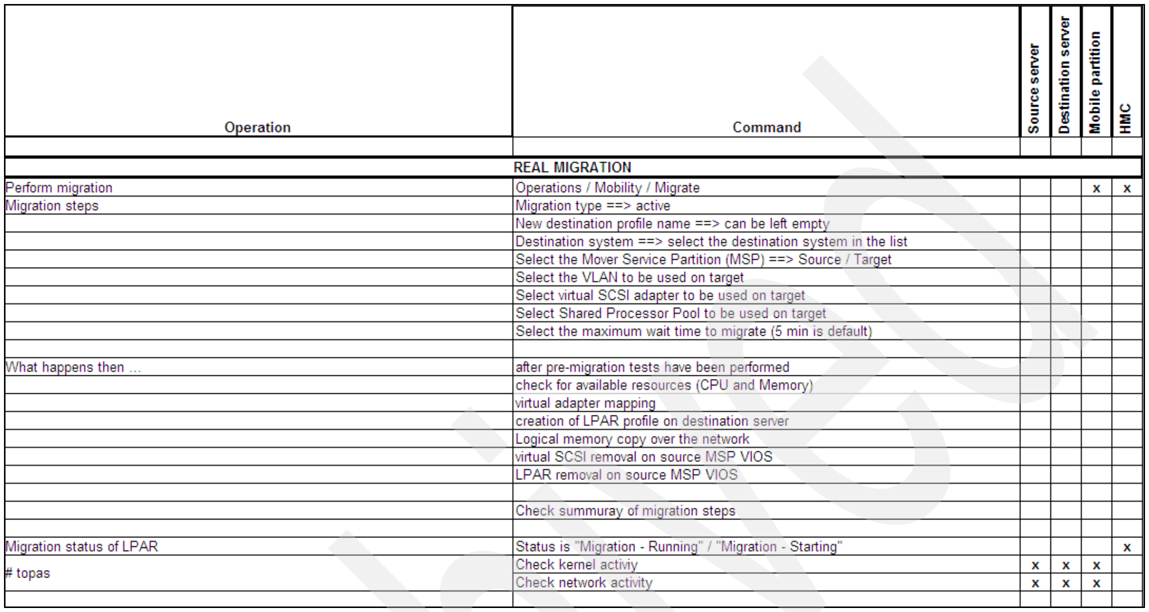


Operation	Command	Source server	Destination server	Mobile partition	HMC
PRE-MIGRATION TESTS					
At this stage we MUST have the following configured:	- both servers have at least one MSP VIOS server ==> this creates a VASI virtual device - a private network connection to the HMC - a SEA failover mechanism - a disk mapping from a IUN on the SAN - a vhost connection from the VIOS to the mobile partition - a virtual network that allows communication between ... the mobile partition and all MSP VIOS servers and other VIOS servers to allow RMC communication between all				
Validate the migration prior actually running the migration	Operation / Mobilkity / Validation				x
	or				
	lslparmigr -r virtualio -m <system name 1> -t <system name 2> --filter lpar_names=<lpar name>				
	lslparmigr -r msp -m <system name 1> -t <system name 2> --filter "lpar_names=<lpar name>"				
	migrlpar -o v -m <system name 1> -t <system name 2> -p p01 -d 5 -v if option "-o m" is used instead of "-o v" than the migration actually occurs swap the "m" and "t" to reverse from mover/target to target/mover				x

Figure 7-56 LPM: Pre-migration tests (validation)

7.7.3 LPM: Real migration

Figure 7-57 shows the real migration operations.

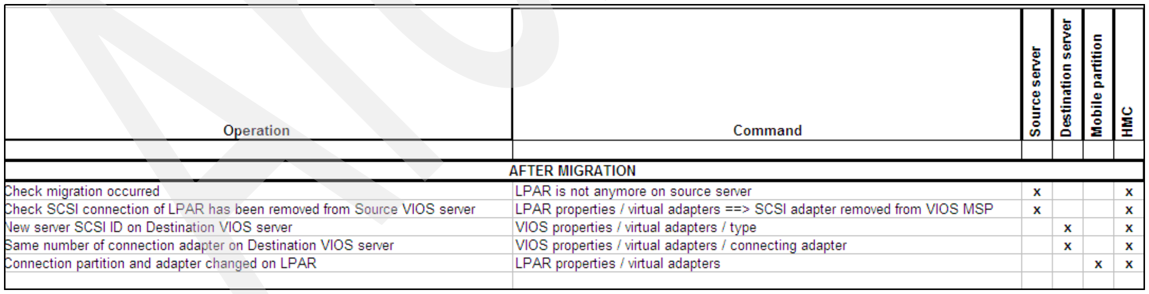


Operation	Command	Source server	Destination server	Mobile partition	HMC
	REAL MIGRATION				
Perform migration	Operations / Mobility / Migrate			x	x
Migration steps	Migration type ==> active				
	New destination profile name ==> can be left empty				
	Destination system ==> select the destination system in the list				
	Select the Mover Service Partition (MSP) ==> Source / Target				
	Select the VLAN to be used on target				
	Select virtual SCSI adapter to be used on target				
	Select Shared Processor Pool to be used on target				
	Select the maximum wait time to migrate (5 min is default)				
What happens then ...	after pre-migration tests have been performed				
	check for available resources (CPU and Memory)				
	virtual adapter mapping				
	creation of LPAR profile on destination server				
	Logical memory copy over the network				
	virtual SCSI removal on source MSP VIOS				
	LPAR removal on source MSP VIOS				
	Check summuray of migration steps				
Migration status of LPAR	Status is "Migration - Running" / "Migration - Starting"				x
# topas	Check kernel activiy	x	x	x	
	Check network activity	x	x	x	

Figure 7-57 LPM: Real migration

7.7.4 Post migration

Figure 7-58 shows the post-migration operations.

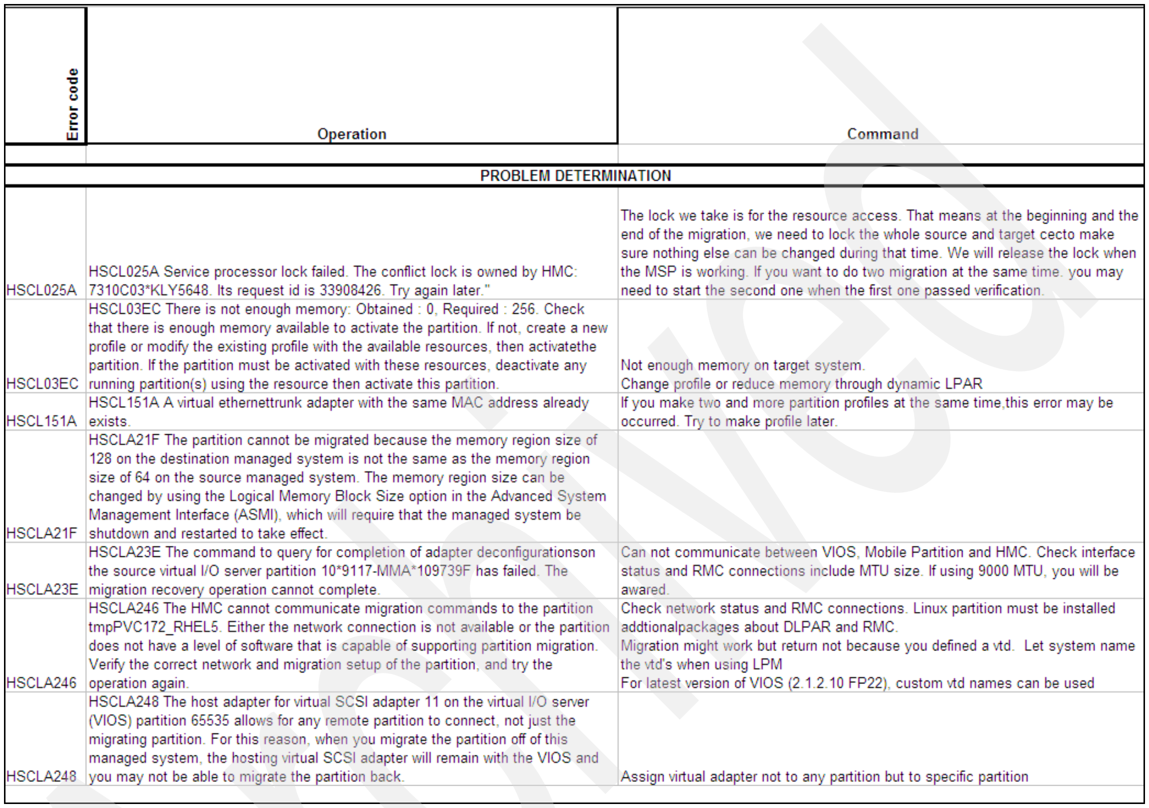


Operation	Command	Source server	Destination server	Mobile partition	HMC
	AFTER MIGRATION				
Check migration occurred	LPAR is not anymore on source server	x			x
Check SCSI connection of LPAR has been removed from Source VIOS server	LPAR properties / virtual adapters ==> SCSI adapter removed from VIOS MSP	x			x
New server SCSI ID on Destination VIOS server	VIOS properties / virtual adapters / type		x		x
Same number of connection adapter on Destination VIOS server	VIOS properties / virtual adapters / connecting adapter		x		x
Connection partition and adapter changed on LPAR	LPAR properties / virtual adapters			x	x

Figure 7-58 LPM: After migration

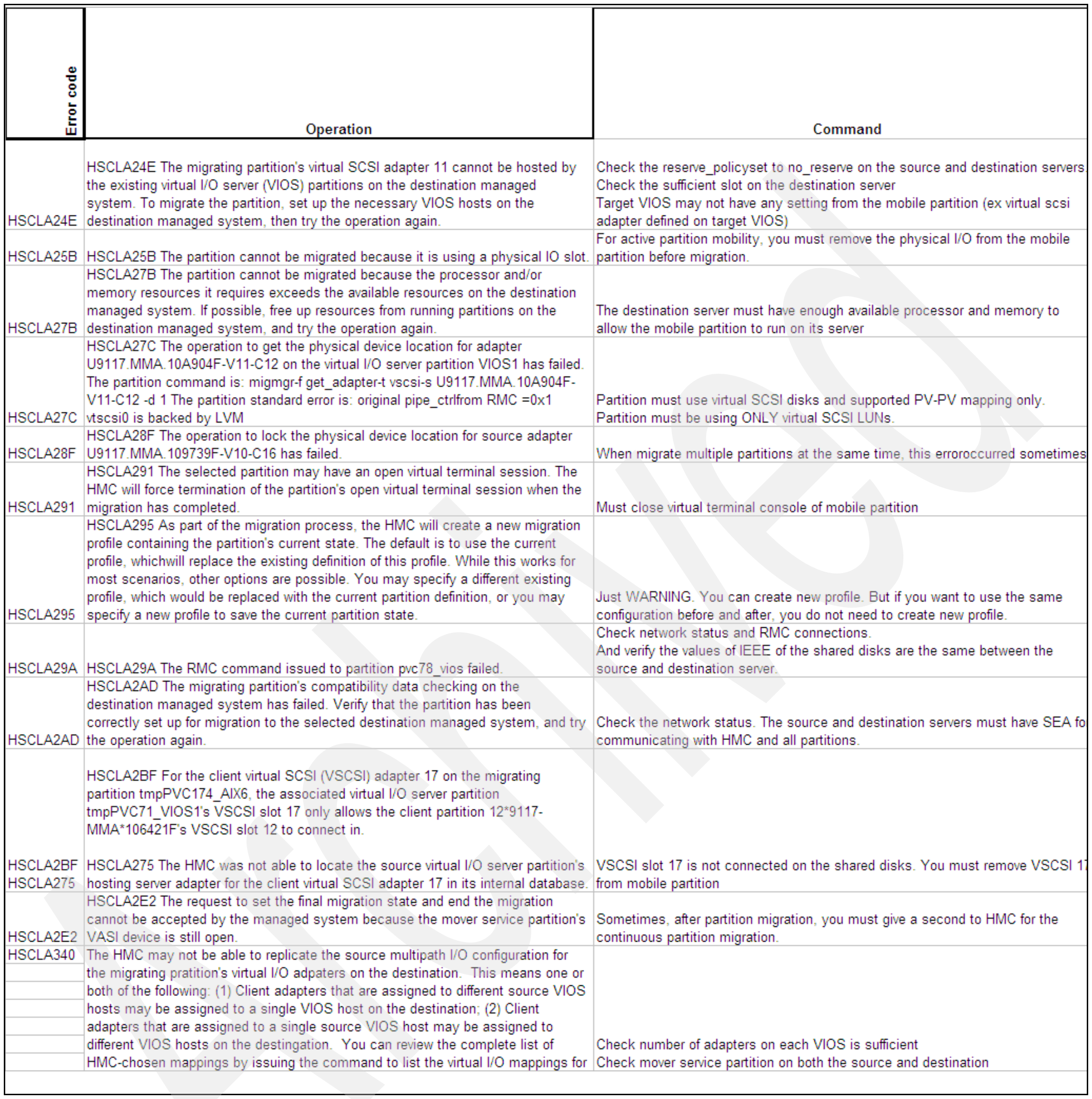
7.7.5 Problem determination

Figure 7-59 and Figure 7-60 on page 358 show the operations related with problem determination.



Error code	Operation	Command
PROBLEM DETERMINATION		
HSCL025A	HSCL025A Service processor lock failed. The conflict lock is owned by HMC: 7310C03*KLY5648. Its request id is 33908426. Try again later."	The lock we take is for the resource access. That means at the beginning and the end of the migration, we need to lock the whole source and target cecto make sure nothing else can be changed during that time. We will release the lock when the MSP is working. If you want to do two migration at the same time. you may need to start the second one when the first one passed verification.
HSCL03EC	HSCL03EC There is not enough memory: Obtained : 0, Required : 256. Check that there is enough memory available to activate the partition. If not, create a new profile or modify the existing profile with the available resources, then activatethe partition. If the partition must be activated with these resources, deactivate any running partition(s) using the resource then activate this partition.	Not enough memory on target system. Change profile or reduce memory through dynamic LPAR
HSCL151A	HSCL151A A virtual ethernettrunk adapter with the same MAC address already exists.	If you make two and more partition profiles at the same time,this error may be occurred. Try to make profile later.
HSCLA21F	HSCLA21F The partition cannot be migrated because the memory region size of 128 on the destination managed system is not the same as the memory region size of 64 on the source managed system. The memory region size can be changed by using the Logical Memory Block Size option in the Advanced System Management Interface (ASMI), which will require that the managed system be shutdown and restarted to take effect.	
HSCLA23E	HSCLA23E The command to query for completion of adapter deconfigurationson the source virtual I/O server partition 10*9117-MMA*109739F has failed. The migration recovery operation cannot complete.	Can not communicate between VIOS, Mobile Partition and HMC. Check interface status and RMC connections include MTU size. If using 9000 MTU, you will be awared.
HSCLA246	HSCLA246 The HMC cannot communicate migration commands to the partition tmpPVC172_RHEL5. Either the network connection is not available or the partition does not have a level of software that is capable of supporting partition migration. Verify the correct network and migration setup of the partition, and try the operation again.	Check network status and RMC connections. Linux partition must be installed addtionalpackages about DLPAR and RMC. Migration might work but return not because you defined a vtd. Let system name the vtd's when using LPM For latest version of VIOS (2.1.2.10 FP22), custom vtd names can be used
HSCLA248	HSCLA248 The host adapter for virtual SCSI adapter 11 on the virtual I/O server (VIOS) partition 65535 allows for any remote partition to connect, not just the migrating partition. For this reason, when you migrate the partition off of this managed system, the hosting virtual SCSI adapter will remain with the VIOS and you may not be able to migrate the partition back.	Assign virtual adapter not to any partition but to specific partition

Figure 7-59 LPM: Problem determination, part 1 of 2



Error code	Operation	Command
HSCLA24E	HSCLA24E The migrating partition's virtual SCSI adapter 11 cannot be hosted by the existing virtual I/O server (VIOS) partitions on the destination managed system. To migrate the partition, set up the necessary VIOS hosts on the destination managed system, then try the operation again.	Check the reserve_policyset to no_reserve on the source and destination servers Check the sufficient slot on the destination server Target VIOS may not have any setting from the mobile partition (ex virtual scsi adapter defined on target VIOS)
HSCLA25B	HSCLA25B The partition cannot be migrated because it is using a physical IO slot.	For active partition mobility, you must remove the physical I/O from the mobile partition before migration.
HSCLA27B	HSCLA27B The partition cannot be migrated because the processor and/or memory resources it requires exceeds the available resources on the destination managed system. If possible, free up resources from running partitions on the destination managed system, and try the operation again.	The destination server must have enough available processor and memory to allow the mobile partition to run on its server
HSCLA27C	HSCLA27C The operation to get the physical device location for adapter U9117.MMA.10A904F-V11-C12 on the virtual I/O server partition VIOS1 has failed. The partition command is: migmgr-f get_adapter-t vscsi-s U9117.MMA.10A904F-V11-C12 -d 1 The partition standard error is: original pipe_ctrlfrom RMC =0x1 vtscsi0 is backed by LVM	Partition must use virtual SCSI disks and supported PV-PV mapping only. Partition must be using ONLY virtual SCSI LUNs.
HSCLA28F	HSCLA28F The operation to lock the physical device location for source adapter U9117.MMA.109739F-V10-C16 has failed.	When migrate multiple partitions at the same time, this erroroccurred sometimes
HSCLA291	HSCLA291 The selected partition may have an open virtual terminal session. The HMC will force termination of the partition's open virtual terminal session when the migration has completed.	Must close virtual terminal console of mobile partition
HSCLA295	HSCLA295 As part of the migration process, the HMC will create a new migration profile containing the partition's current state. The default is to use the current profile, whichwill replace the existing definition of this profile. While this works for most scenarios, other options are possible. You may specify a different existing profile, which would be replaced with the current partition definition, or you may specify a new profile to save the current partition state.	Just WARNING. You can create new profile. But if you want to use the same configuration before and after, you do not need to create new profile.
HSCLA29A	HSCLA29A The RMC command issued to partition pvc78_vios failed.	Check network status and RMC connections. And verify the values of IEEE of the shared disks are the same between the source and destination server.
HSCLA2AD	HSCLA2AD The migrating partition's compatibility data checking on the destination managed system has failed. Verify that the partition has been correctly set up for migration to the selected destination managed system, and try the operation again.	Check the network status. The source and destination servers must have SEA fo communicating with HMC and all partitions.
HSCLA2BF HSCLA275	HSCLA2BF For the client virtual SCSI (VSCSI) adapter 17 on the migrating partition tmpPVC174_AIX6, the associated virtual I/O server partition tmpPVC71_VIOS1's VSCSI slot 17 only allows the client partition 12*9117-MMA*106421F's VSCSI slot 12 to connect in. HSCLA275 The HMC was not able to locate the source virtual I/O server partition's hosting server adapter for the client virtual SCSI adapter 17 in its internal database.	VSCSI slot 17 is not connected on the shared disks. You must remove VSCSI 17 from mobile partition
HSCLA2E2	HSCLA2E2 The request to set the final migration state and end the migration cannot be accepted by the managed system because the mover service partition's VASI device is still open.	Sometimes, after partition migration, you must give a second to HMC for the continuous partition migration.
HSCLA340	The HMC may not be able to replicate the source multipath I/O configuration for the migrating pratition's virtual I/O adpaters on the destination. This means one or both of the following: (1) Client adapters that are assigned to different source VIOS hosts may be assigned to a single VIOS host on the destination; (2) Client adapters that are assigned to a single source VIOS host may be assigned to different VIOS hosts on the destingation. You can review the complete list of HMC-chosen mappings by issuing the command to list the virtual I/O mappings for	Check number of adapters on each VIOS is sufficient Check mover service partition on both the source and destination

Figure 7-60 LPM: Problem determination, part 2 of 2

8

Workload partitioning

A Workload Partition (WPAR) is an isolated execution environment with its own init process inside the AIX environment. It creates a virtualized operating system environment for managing multiple workloads. To the user, WPAR appears as a standalone AIX system.

In this chapter we discuss WPAR concepts and the how DB2 can be configured under a WPAR environment.

This chapter has the following sections:

8.1 Overview of WPAR

WPAR is the software-based partitioning provided by the operating system.One AIX operating instance is partitioned into multiple instances, each called a *workload partition*. Each workload partition can host applications and isolate them from applications executing within other WPARS. Figure 8-1 shows a sample LPAR and WPAR configuration.

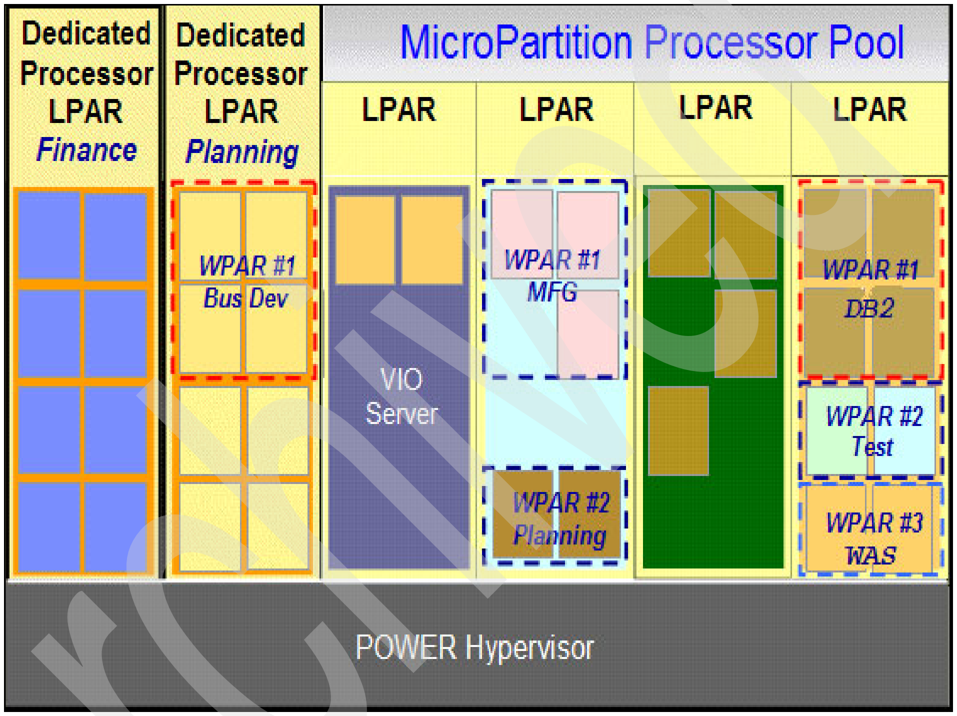


Figure 8-1 LPAR and WPAR

Prior to WPARs, it was required to create a new logical partition (LPAR) for each new isolated environment. With AIX 6.1, this is no longer necessary, as there are many circumstances when one can get along fine with multiple WPARs within one LPAR. Why is this important? Every LPAR requires its own operating system image and a certain number of physical resources. Although you can virtualize many of these resources, there are still physical resources that must be allocated to the system. Furthermore, the need to install patches and technology upgrades to each LPAR is cumbersome. Each LPAR requires its own archiving strategy and DR strategy. It also takes time to create an LPAR through a Hardware Management Console (HMC) or the Integrated Virtualization Manager (IVM).

Table 8-1 compares LPARs and WPARs.

Table 8-1 LPAR versus WPAR

LPAR	**WPAR**
Logical partition in a server.	Workload partition within LPAR.
Difficult to create and manage (needs HMC, IVM).	Easy to create and manage.
Helps to consolidate and virtualize hardware resources in a single server.	Helps in resource management by sharing AIX images and using the hardware resources.
LPAR is the single point of failure for WPAR.If LPAR fails then WPAR fails.	if WPAR fails then only that instance which was partitioned gets affected.

8.2 Other types of WPARs

As mentioned, workload partitions are created within AIX 6.1 instances. The instance that does not belong to any workload partition is called a *global environment*. A system administrator must be logged into the global environment to create, activate, and manage the workload partitions.

Important: Most performance monitoring and tuning activities are performed from the global environment.

There are two types of workload partitions that can reside in a global environment.

- System WPAR: Almost a full AIX environment.
- Application WPAR: Light environment suitable for execution of one or more processes.

Figure 8-2 shows the global environment and WPAR.

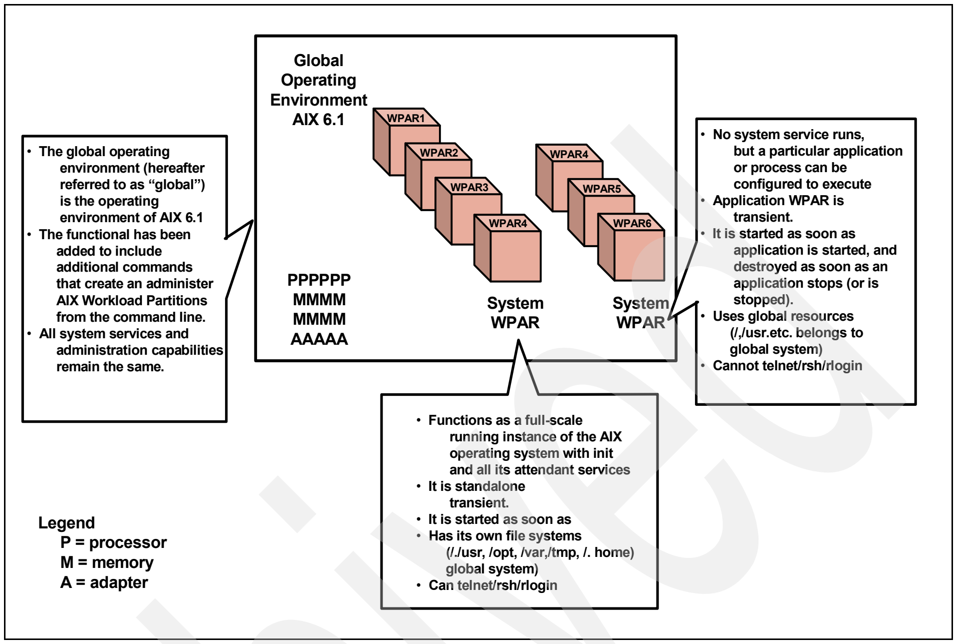

Global Operating Environment AIX 6.1
WPAR1
WPAR2
WPAR3
WPAR4
WPAR4
WPAR5
WPAR6
PPPPPP MMMM MMMM AAAAA
System WPAR
System WPAR
• The global operating environment (hereafter referred to as "global") is the operating environment of AIX 6.1 • The functional has been added to include additional commands that create an administer AIX Workload Partitions from the command line. • All system services and administration capabilities remain the same.
• Functions as a full-scale running instance of the AIX operating system with init and all its attendant services • It is standalone transient. • It is started as soon as • Has its own file systems (/./usr, /opt, /var,/tmp, /. home) global system) • Can telnet/rsh/rlogin
• No system service runs, but a particular application or process can be configured to execute • Application WPAR is transient. • It is started as soon as application is started, and destroyed as soon as an application stops (or is stopped). • Uses global resources (/,/usr.etc. belongs to global system) • Cannot telnet/rsh/rlogin
Legend
P = processor
M = memory
A = adapter

Figure 8-2 Global environment and WPAR

8.2.1 System WPAR

System WPAR has the following characteristics:

- ► Dedicated writable file system. It can share the global environments /usr and /opt as read only mode or it can has its own private /usr, /opt as well.
- ► Its own init process
- ► inetd daemon allows complete networking capacity, so remote login to the system LPAR is possible
- ► Runs cron daemon, so execution of process can be scheduled

Note: At the time of writing this IBM Redbooks publication, running NFS server inside the WPAR was not supported.

Creating a system WPAR

The following sections discuss the creation of a system WPAR with shared /usr and private /usr.

System WPAR with shared /usr

In this WPAR, /usr and /opt of global environment are shared by the system WPAR. The following command creates the shared system WPAR:

```
mkwpar -n <system wpar name>
```

For example: **`mkwpar -n sys_wpar_shared_usr`**.

System WPAR with private /usr

In this WPAR, /usr and /opt are created separately for the system WPAR. The following command creates the private system WPAR:

```
mkwpar -l -n “wpar name”
```

If the network configuration needs to be done while creating the workload, run the following command:

```
mkwpar -l -N interface=en0 address="IP" netmask=255.255.255.192
broadcast=9.2.60.255 -n "wpar name having DNS entry"
```

8.2.2 Application WPAR

An application WPAR has the following characteristics:

- Any application that can be started as a one command in the AIX command line interface is the candidate for application WPAR.
- It does not own any dedicated storage and it shares the file system of the global environment.
- It can run daemons but it cannot run any system service daemons such as inetd, srcmstr, and so forth.
- Remote login is not possible

Creating an application WPAR

The **`wparexec`** command is used to create the wpar application as shown:

```
wparexec -n <“Application wpar name” > <path of the application with
arguments,if any>
```

For example:

```
wparexec -n appwpar /tmp/myApp
```

Example 8-1 shows a sample output of the wparexec command.

Example 8-1 WPAR command

```
# wparexec /usr/bin/ls
Starting workload partition ls.
Mounting all workload partition file systems.
Loading workload partition.
.com.zerog.registry.xml  ibm                      perf
LicenseUseManagement     ifor                     preserve
aacct                    lib                      run
adm                      locks                    security
ct                       log                      snapp
db2                      lost+found               spool
ecc                      msgs                     ssl
empty                    ncs                      statmon
esa                      news                     tmp
ha                       opt                      websm
hacmp                    pconsole                 yp
Shutting down all workload partition processes.
```

8.3 Installing and configuring DB2 on a WPAR

DB2 installation is supported only on a system WPAR. System WPARs either share the /usr and /opt directories with the global environment, or have a local copy of the /usr and /opt directories.

- A DB2 product can be installed in a global environment with a DB2 copy shared with other system WPARs.

  When a DB2 copy is installed in a global environment under either the /usr or /opt directory, which are shared with system WPARs, those system WPARs use the shared DB2 copy to set up DB2 instances.
- A DB2 product can be installed in a local file system on a system WPAR.
- Each system WPAR manages its own DB2 instances.
- DB2 instances created on one WPAR, or in a global environment, are not visible from any other system (system WPAR or global environment).

8.3.1 Installing the DB2 copy

Installing a DB2 copy on a system WPAR is similar to any other DB2 product installation, with the following exceptions. The following items cannot be installed on a system WPAR:

- IBM Tivoli System Automation for Multiplatforms (SAMP)
- IBM Data Studio Administration Console

Note: UnInstalling the DB2 copy considerations:

- System WPAR is active
- All instances within the system WPAR is dropped or update it to other DB2 copy (using "db2iupdt")

DB2 instance creation under system WPAR

Perform the following steps to create a DB2 instance under system WPAR.

1. Run the following command:

   ```
   mkwpar -l -n sys_wpar_db2_private
   ```

   In this command, `sys_wpar_db2_private` is the WPAR name.

2. To start the workload partition, execute the following as root:

   ```
   startwpar [-v] sys_wpar_db2_private
   ```

3. Verify the status of the WPAR using the **`lswpar`** command, as shown in Example 8-2.

Example 8-2 Listing the characteristics of WPAR

```
# lswpar
Name          State  Type  Hostname      Directory
-------------------------------------------------------------
sys_wpar_db2_private  A      S     sys_wpar_db2_private
/wpars/sys_wpar_db2_private
```

4. Log in to the newly created WPAR using the **clogin** command. See Example 8-3.

```
clogin sys_wpar_db2_private
```

Example 8-3 Log in to system WPAR using “clogin”

```
# clogin sys_wpar_db2_private
*******************************************************************
*
*  Welcome to AIX Version 6.1!
*
*  Please see the README file in /usr/lpp/bos for information pertinent
to this release of the AIX Operating System.
*
********************************************************************


# hostname
sys_wpar_db2_private
# df
Filesystem    512-blocks      Free %Used    Iused %Iused Mounted on
Global            196608    141152   29%     1999    12% /
Global             65536     63776    3%        5     1% /home
Global           4653056   2096048   55%    27254    11% /opt
Global                 -         -    -         -     -  /proc
Global            196608    193376    2%       11     1% /tmp
Global          10027008   5658984   44%    42442     7% /usr
Global            262144    123080   54%     4371    24% /var
```

5. Create groups and users using the commands shown in Example 8-4.

Example 8-4 Create groups and users

```
mkgroup id=998 db2grp
mkgroup id=999 db2iadm1
useradd -g db2grp -u 200 -md /home/db2inst1 db2inst1
useradd -g db2grp -u 201 -md /home/db2fenc1 db2fenc1
passwd db2inst1 (to set the password)
passwd db2fenc1 (to set the password)
```

6. Update the `/etc/hosts` file with the new host name, as shown in Example 8-5.

Example 8-5 Update the /etc/hosts file

```
#127.0.0.1              loopback localhost      # loopback (lo0) 
name/address (comment existing entry)
127.0.0.1   sys_wpar_db2_private.xx.xx.com  sys_wpar_db2_private (new 
entry)
```

7. Create the DB2 Instance. See Example 8-6.

   `db2icrt -p 50000 -s ese -u db2fenc1 db2inst1`

Example 8-6 Create the DB2 Instance

```
# ./db2icrt -p 50000 -s ese -u db2fenc1 db2inst1
DBI1070I Program db2icrt completed successfully.

# su - db2inst1
$ db2start
12/02/2009 12:18:51     0   0   SQL1063N  DB2START processing was 
successful.
SQL1063N  DB2START processing was successful.
```

Note: DB2 cannot be installed in system WPAR that does not have write permission to global environment's /usr and /opt.

8.4 Benefits of WPAR

WPAR offers customers the following benefits:

- Hundreds of WPARS can be created, far exceeding the capability of other partitioning technologies.
- WPAR resource controls enable the over-provisioning of resources. If a WPAR is below allocated levels, the unused allocation is automatically available to other WPARS.
- Because there are fewer operating system images when WPARs are used, there is a reduction in the total amount of AIX system administration and maintenance tasks.
- There is a reduction in the total amount of system resources needed because you do not need as much CPU and memory capacity when you have fewer instances of AIX.

- Even though the OS image used for all WPARs in one LPAR is the same, it is possible to install versions of the applications used in system WPARs.
- When you run an application in a WPAR, your application is isolated from other applications, making it easier to monitor and control resources for that application.
- With WPARs, you can configure unique user and security access roles for system WPARs.
- No performance cost for using virtual devices.
- WPARS support fast provisioning and fast resource adjustments in response to normal/unexpected demands. WPARS can be created and resource controls modified in seconds.
- Enablement of Live Application Mobility for planned outages and workload distribution.

8.4.1 When to use WPARs

As discussed earlier, WPARs are virtualized operating system environments that are created within a single AIX image. Although they might be self-contained in the sense that each WPAR has its own private execution environment with its own file systems and network addresses, they still run inside the global environment. The global environment, the actual LPAR, owns all the physical resources of the logical partition. It is important to note that the global environment can see all the processes running inside the specific WPARs. The following list details scenarios that call for the use of WPARs.

- Application and workload isolation

  WPARs enable development, test, and production cycles of one workload to be placed on a single system. Every organization has at least these three environments for their applications. As a common practice, for each of these environments you need to create a LPAR because in today’s world of high availability, it is natural to give each application environment its own home. It can be a difficult task for a system administrator to maintain all these environments. This is where the WPAR has the most value. Each of the environments can be deployed on a separate WPAR within a LPAR, reducing costs through sharing of hardware resources and software resources (such as operating systems, databases and other tools).

- An ideal sandbox environment

  Creating a separate test environment for system administrators has never been easier. With WPARs, administrators now have the opportunity to install new software, test out new patches, install new technology levels and generally have the flexibility to experiment on the system without any impact to the other users.

- Troubleshooting

  It takes a minimal amount of time to create application WPARs, making it ideal for quickly troubleshooting or reproducing a customer scenario. As these are temporary resources, they are destroyed as soon as they end, which simplifies the manageability of these partitions.

8.4.2 When WAPR might not be the best choice

Here are a few cases when using WAPR might not be the best choice:

- Security

  As stated previously, WPAR processes can be seen by the global environment from the central LPAR. If you cannot compromise on application security, do not use a WPAR.

- Performance

  When doing a performance benchmark on an application running on a WPAR, remember that the WPARs within the LPAR are sharing system resources, so race conditions are bound to occur that might change benchmark results.

- Availability

  If you are in an environment where it is difficult to bring a system down, it is important to note that when performing maintenance on an LPAR that every WPAR defined is affected. At the same time, if there is a system panic and AIX crashes, every WPAR has now been brought down. From this standpoint, LPARs without WPARs can provide increased availability across your environment, albeit at a cost that might be prohibitive.

- Production

  If your application demands the granularity and complete OS isolation that LPARs provide, without having multiple environments to worry about, you must go in for a LPAR as against a WPAR.

- Physical devices

  Physical devices are not supported within a WPAR. Although there is a way to export devices, this can be a problem for applications that require non-exportable devices. In this case, they are restricted to running in the global environment.

8.5 Live Application Mobility versus Live Partition Mobility

Live Application Mobility (LAM) provides the ability to move a running WPAR and its applications, without moving the AIX instance that contains the WPAR. It is implemented in software and does not have any dependency on a POWER processor chip. Any POWER processor-based server running AIX 6.1 supports workload partitions, whereas Live Partition Mobility provides the ability to move a running LPAR and all of its running applications from one physical server to another physical server without disrupting operation of the LPAR. It is only available on POWER6-processor-based servers. The AIX versions supported are 5.3 TL7 and 6.1.

You achieve near continuous availability because Live Partition Mobility and LAM move running workloads between Power Systems to eliminate the need for planned system downtime. Both LPM and LAM eliminate downtime during planned outages but cannot increase system availability during unplanned outages such as a hardware crash. Therefore, it is not a replacement for HACMP.

With Live Partition Mobility, you can move an active or inactive partition from one physical server to another without the user aware of the change happening. With LAM, however, you have to check-stop your WPAR so that it restarts on the destination machine. This might result in brief periods of unresponsiveness.

Table 8-2 lists the differences between Partition Mobility and Application Mobility.

Table 8-2 Basic differences between Partition Mobility and Application Mobility

Type	Partition Mobility	Application Mobility
OS	Linux, AIX 5.3, AIX 6.1	AIX6.1
Hardware	POWER6	PowerPC 970, POWER4, 5, 6.
Functionality	Server to Server	Within the same AIX instance.
	Active and inactive partition can be moved	Stop the application and restart on the destination machine.This results in brief periods of unresponsiveness.

Note: Both application mobility and partition mobility can eliminate downtime during planned outages but does not provide high availability. They are not a replacement for HACMP.

A

System health check

This appendix describes the tools used for the DB2 and AIX health check.

Why collect ongoing data and data before any major change?

Over time and through planned and unplanned outages, the system can start behaving differently. This difference can be a positive or it can have adverse effects (for example, an increase in memory/paging space, queries that used to complete in a few seconds taking minutes, increase in I/O on certain disks and so forth). If information is collected at a point in time before such adverse effects are noticed and the same information is collected after the adverse effects or while they are occurring, it makes diagnosing the location of the problem easier. If, for example you collect access plans of a certain query, before the query used to take a greater amount of time, you can use theseaccess plans to diagnose why the query is experiencing a performance degradation.

The same applies for AIX-related information. For example, if information regarding the system parameters and information about the Logical Volume Manager (such as mount options and file system layout) is collected before I/O contention, comparing two sets of relevant identical data aids the problem determination process.

Why perform ongoing health checks and act on these?

Collecting data is useful for problem determination. However, there are ongoing health checks that assist in ensuring that the system is performing at an optimal level. Without regular health checks, and given that business applications seldom remain constant, it is almost certain that the systems overall performance can be affected. An example of this at the AIX level can monitoring any hot disks and time spent in waiting for I/O to complete. Regular checks can allow system administrators and database administrators to avoid future problems but also allow the system to perform optimally.

DB2 health check

Table A-1 and Table A-2 on page 374 summarize the tools you can use for DB2 health check.

Table A-1 Ongoing data and data before any major change

Task	Date Completed	Location of Output
db2support An all-purpose utility that collects data ranging from ICBM / db cfg, table space layout to contents of history file information. Additional information can be found at the following Web page: `http://publib.boulder.ibm.com/infocenter/db2luw/v9r5/index.jsp?topic=/com.ibm.db2.luw.admin.trb.doc/doc/t0020808.html`		
db2support to gather optimizer data If, for example you collect access plans of a certain query, before the query used to take a greater amount of time, you can use these access plans to diagnose why the query is experiencing a performance degradation. db2support utility invoked in the optimizer mode can be to collect such data. Additional information can be found at the following Web page: `http://publib.boulder.ibm.com/infocenter/db2luw/v9r5/index.jsp?topic=/com.ibm.db2.luw.admin.cmd.doc/doc/r0004503.html`		
db2fodc - perf db2fodc is a utility that can be invoked to gather performance-related data. By saving the DB2 performance-related data when the system is performing well you can establish a baseline to compare performance issues if the DB2 performance degrades. Additional information can be found at the following Web page: `http://publib.boulder.ibm.com/infocenter/db2luw/v9r5/index.jsp?topic=/com.ibm.db2.luw.admin.cmd.doc/doc/r0051934.html`		
db2cfexp / db2cfimp These utilities are used to export connectivity configuration information to an export profile, which can later be imported. Additional information can be found at the following Web page: `http://publib.boulder.ibm.com/infocenter/db2luw/v9r5/index.jsp?topic=/com.ibm.db2.luw.qb.migration.doc/doc/t0023398.html`		

Table A-2 Ongoing health checks

Task	Date Completed	Location of Output
Taking regular backups This is vital to any disaster recovery strategy. Whether it be taking db2 backups or other forms of ensuring that the database can be recovered, this task is of utmost importance. Additional information can be found at the following Web page: `http://publib.boulder.ibm.com/infocenter/db2luw/v9r5/index.jsp?topic=/com.ibm.db2.luw.admin.ha.doc/doc/c0005945.html`		
Ensuring logs are archived successfully If the database is enabled for rollforward recovery, it is crucial to ensure that the logs have been archived to a safe location in the event of a failure. Additional information can be found at the following Web page: `http://publib.boulder.ibm.com/infocenter/db2luw/v9r5/index.jsp?topic=/com.ibm.db2.luw.admin.ha.doc/doc/c0006086.html`		
Space usage in table spaces This task has been made easier by the introduction of automatic storage. There are various health monitoring options that can even pages out if space available falls below defined thresholds. Additional information can be found at the following Web pages: ► `http://publib.boulder.ibm.com/infocenter/db2luw/v9r5/index.jsp?topic=/com.ibm.db2.luw.admin.dbobj.doc/doc/c0052484.html` ► `http://publib.boulder.ibm.com/infocenter/db2luw/v9r5/index.jsp?topic=/com.ibm.db2.luw.admin.mon.doc/doc/c0011705.html`		
reorgchk / reorg Over time data and indexes in table spaces can become fragmented as they are inserted, updated, and deleted. To check for this and correct it use the **`reorgchk`** and **`reorg`** commands. Additional information can be found at the following Web pages: ► `http://publib.boulder.ibm.com/infocenter/db2luw/v9r5/index.jsp?topic=/com.ibm.db2.luw.admin.mon.doc/doc/c0011705.html` ► `http://publib.boulder.ibm.com/infocenter/db2luw/v9r5/index.jsp?topic=/com.ibm.db2.luw.admin.perf.doc/doc/c0005406.html`		
runstats The optimizer needs to use accurate statistics to ensure that the access plan chosen for any given query is optimal. It relies on the information gathered as part of the collection of statistics that is performed through the **`runstats`** command. Additional information can be found at the following Web page: `http://publib.boulder.ibm.com/infocenter/db2luw/v9r5/index.jsp?topic=/com.ibm.db2.luw.admin.cmd.doc/doc/r0001980.html`		

Task	Date Completed	Location of Output
Rebinding packages Applications that issue static SQL statements have their access plan stored in the system catalogs. If there are changes in data and new statistics have been gathered to reflect these changes, these packages that contain the data access methods need to be rebound. Additional information can be found at the following Web page: `http://publib.boulder.ibm.com/infocenter/db2luw/v9r5/index.jsp?topic=/com.ibm.db2.luw.qb.migration.doc/doc/t0022384.html`		
Dynamic SQL Without delving much into tasks involved with monitoring, one monitoring check is included here. That being the time taken by individual dynamic SQL statements to identify which SQL statements are consuming the most CPU and are I/O intensive. Additional information can be found at the following Web page: `http://publib.boulder.ibm.com/infocenter/db2luw/v9r5/index.jsp?topic=/com.ibm.db2.luw.admin.mon.doc/doc/r0007635.html`		
db2advis The design advisor can assist in the creation of indexes or even MQT's for given workloads and individual SQL statements. After identifying a particular SQL statement, the db2advis utility can be used to recommend various indexes and the performance gain that might result as a result of creating such recommendations. Additional information can be found at the following Web page: `http://publib.boulder.ibm.com/infocenter/db2luw/v9r5/index.jsp?topic=/com.ibm.db2.luw.admin.cmd.doc/doc/r0002452.html`		
Diagnostic logs Prudent DBAs not only monitoring the state of their database but also check their diagnostic logs to find errors that might not be apparent on the surface. Two such logs to keep checking are the db2diag.log and the Administration notification log. Additional information can be found at the following Web pages: ► `http://publib.boulder.ibm.com/infocenter/db2luw/v9r5/index.jsp?topic=/com.ibm.db2.luw.admin.ha.doc/doc/c0023140.html` ► `http://publib.boulder.ibm.com/infocenter/db2luw/v9r5/index.jsp?topic=/com.ibm.db2.luw.admin.trb.doc/doc/c0020815.html`		
db2dart / db2 inspect To check the structural integrity of the underlying database table space and containers in which the data, indexes, lobs and so forth reside, use the db2dart and or the db2 inspect utilities. Additional information can be found at the following Web page: `http://publib.boulder.ibm.com/infocenter/db2luw/v9r5/index.jsp?topic=/com.ibm.db2.luw.admin.trb.doc/doc/c0020763.html`		

AIX health check

Table A-3 summarizes the tools you can use for AIX health check.

Table A-3 AIX health check tools

Task	Date Completed	Location of Output
NMON nmon (short for Nigel's Monitor) is a popular system monitor tool for the AIX and Linux operating systems. It provides monitoring information about the overall health of these operating systems. Information about nmon can be found from the following Web page: `http://www.ibm.com/developerworks/aix/library/au-analyze_aix/`		
VMSTAT This tool is useful for reporting statistics about kernel threads, virtual memory, disks, and CPU activity. Information about VMSTAT usage can be found in following Web page: `http://publib.boulder.ibm.com/infocenter/aix/v6r1/topic/com.ibm.aix.cmds/doc/aixcmds6/vmstat.htm`		
IOSTAT The iostat tool reports CPU statistics and input/output statistics for TTY =devices, disks, and CD-ROMs. See the following Web page for details on using IOSTAT: `http://publib.boulder.ibm.com/infocenter/aix/v6r1/index.jsp?topic=/com.ibm.aix.cmds/doc/aixcmds3/iostat.htm`		
LPARSTAT Use this tool to check the resources for the LPAR on AIX. It can be used to see the overall CPU usage relative to the shared pool and to get statistics with regard to the power hypervisor. Information about LPARSTAT usage can be found in the following Web page: `http://publib.boulder.ibm.com/infocenter/aix/v6r1/topic/com.ibm.aix.cmds/doc/aixcmds3/lparstat.htm`		
PM Services PM Services can be used to moniitor the overall system vitals of Power on AIX. It can be used as a useful utility for monitoring the overall health of a system or multiple systems. Information about PM can be found in following Web page: `http://www-03.ibm.com/systems/power/support/pm/news.html`		

Task	Date Completed	Location of Output
PS The **PS** command displays statistics and status information about processes in the system, including process or thread ID, I/O activity, and CPU and memory use.The **PS** command can be used to monitor memory use by an individual process. For more information, see the following Web page: http://publib.boulder.ibm.com/infocenter/aix/v6r1/topic/com.ibm.aix.cmds/doc/aixcmds4/ps.htm		
NETSTAT The **netstat** command displays information regarding traffic on the configured network interfaces. For more information, see the following Web page: http://publib.boulder.ibm.com/infocenter/aix/v6r1/topic/com.ibm.aix.cmds/doc/aixcmds4/netstat.htm		
SVMON The svmon can be used for in-depth analysis of memory usage. It displays information about the current state of memory. For more information, see the following Web page: http://publib.boulder.ibm.com/infocenter/aix/v6r1/topic/com.ibm.aix.cmds/doc/aixcmds5/svmon.htm		
PRTCONF Get the basic system configuration. For more information see the following Web page: http://publib.boulder.ibm.com/infocenter/aix/v6r1/topic/com.ibm.aix.cmds/doc/aixcmds4/prtconf.htm		

Abbreviations and acronyms

AIO	asynchronous I/O
AMS	Active Memory Sharing
Ack	acknowledgement
CA	Configuration Advisor
CGTT	created global temporary tables
COD	Capacity On Demand
CTQ	Command Tagged Queueing
DBA	database administrator
DFU	Decimal Floating point unit
DGTT	declared global temporary table
DLPAR	dynamic logical partitioning
DMS	Database managed space
DW	data warehouse
FC	Fibre Channel
GPFS	General Parallel File System
HA	high availability
HADR	High Availability Disaster Recovery
HBA	Host Based Adapter
HMC	Hardware Management Console
IBM	International Business Machines Corporation
IOCP	I/O completion ports
ITSO	International Technical Support Organization
IVE	integrated virtual Ethernet
IVM	Integrated Virtualization Manager
JFS2	Journaled Filesystem Extended
LBAC	Label Based Access Control
LPAR	logical partition
LRU	Least Recently Used
LTG	logical track group
LUN	logical unit number
LVM	Logical Volume Manager
MLS	multilevel security
MPIO	multiple path I/O
MTU	maximum transmission unit
NIM	Network Installation Manager
NUMA	non-uniform memory access
OLTP	online transaction processing
PV	physical volume
RAID	Redundant Array of Independent Disks
RBAC	Role Based Access Control
RID	row identifier
SAMP	System Automation for Multiplatforms
SEA	Shared Ethernet Adapter
SMS	system managed space
SMT	Simultaneous Multi Threading
STMM	Self Tuning Memory Manager
TCO	total cost of ownership
VG	volume groups
VMM	Virtual Memory Manager
VMX	Vector Multimedia extension
WPAR	Workload Partition

Related publications

The publications listed in this section are considered particularly suitable for a more detailed discussion of the topics covered in this book.

IBM Redbooks

For information about ordering these publications, see "How to get Redbooks" on page 382. Note that a few of the documents referenced here might be available in softcopy only.

- *IBM PowerVM Live Partition Mobility*, SG24-7460
- *PowerVM Virtualization on IBM System p: Introduction and Configuration Fourth Edition*, SG24-7940
- *Introduction to the IBM System Storage DS5000 Series*, SG24-7676
- *DB2 UDB V7.1 Performance Tuning Guide*, SG24-6012
- *Integrated Virtualization Manager on IBM System p5*, REDP-4061
- *A Practical Guide for Resource Monitoring and Control (RMC)*, SG24-6615
- *Integrated Virtual Ethernet Adapter Technical Overview and Introduction*, REDP-4340
- *WebSphere Application Server V6.1: JMS Problem Determination*, REDP-4330
- *IBM DB2 9 on AIX 5L with NFS, iSCSI, and FCP using IBM System Storage N series*, REDP-4250
- *IBM System p Advanced POWER Virtualization (PowerVM) Best Practices*, REDP-4194

Online resources

These Web sites are also relevant as further information sources:

- AIX 6.1 information center:

  http://publib.boulder.ibm.com/infocenter/aix/v6r1/index.jsp
- IBM System p information center:

  http://publib16.boulder.ibm.com/pseries/index.htm
- PowerVM information center:

  http://www-03.ibm.com/systems/power/software/virtualization/index.html
- Power Systems information center:

  http://publib.boulder.ibm.com/infocenter/powersys/v3r1m5/index.jsp
- DB2 information center:

  http://publib.boulder.ibm.com/infocenter/db2luw/v9r7/index.jsp
- AIX Commands Reference:

  http://publib.boulder.ibm.com/infocenter/pseries/v5r3/index.jsp?topic=/com.ibm.aix.doc/doc/base/commandsreference.htm
- Information about HMC upgrade:

  http://www-933.ibm.com/support/fixcentral/
- Information about how to set up SSH keys authentication for HMC setup:

  http://publib.boulder.ibm.com/infocenter/powersys/v3r1m5/index.jsp
- Information about how to change your current VIOS level:

  http://www14.software.ibm.com/webapp/set2/sas/f/vios/download/home.html
- Location for DB2 and NAS over iSCSI demo:

  http://www.ibm.com/partnerworld/wps/servlet/ContentHandler/VPAA-7M59ZR

How to get Redbooks

You can search for, view, or download Redbooks, Redpapers, Technotes, draft publications and Additional materials, as well as order hardcopy Redbooks publications, at this Web site:

ibm.com/redbooks

Help from IBM

IBM Support and downloads

ibm.com/support

IBM Global Services

ibm.com/services

Index

IBM

Redbooks

Best Practices for DB2 on AIX 6.1 for POWER Systems

(0.5" spine)
0.475"<->0.875"
250 <-> 459 pages

Best Practices for DB2 on AIX 6.1 for POWER Systems

Explains partitioning and virtualization technologies for Power Systems

Discusses DB2 performance optimization on System p

Covers OLTP and data warehouse workloads

This IBM Redbooks publication presents a best practices guide for DB2 and InfoSphere Warehouse performance on a AIX 6L with Power Systems virtualization environment. It covers Power hardware features such as PowerVM, multi-page support, Reliability, Availability, and Serviceability (RAS) and how to best exploit them with DB2 LUW workloads for both transactional and data warehousing systems.

The popularity and reach of DB2 and InfoSphere Warehouse has grown in recent years. Enterprises are relying more on these products for their mission-critical transactional and data warehousing workloads. It is critical that these products be supported by an adequately planned infrastructure. This publication offers a reference architecture to build a DB2 solution for transactional and data warehousing workloads using the rich features offered by Power systems.

IBM Power Systems have been leading players in the server industry for decades. Power Systems provide great performance while delivering reliability and flexibility to the infrastructure.

This book presents a reference architecture to build a DB2 solution for transactional and data warehousing workloads using the rich features offered by Power systems. It aims to demonstrate the benefits DB2 and InfoSphere Warehouse can derive from a Power Systems infrastructure and how Power Systems support these products.

The book is intended as a guide for a Power Systems specialist to understand the DB2 and InfoSphere Warehouse environment and for a DB2 and InfoSphere Warehouse specialist to understand the facilities available for Power Systems supporting these products.

SG24-7821-00 0738434191